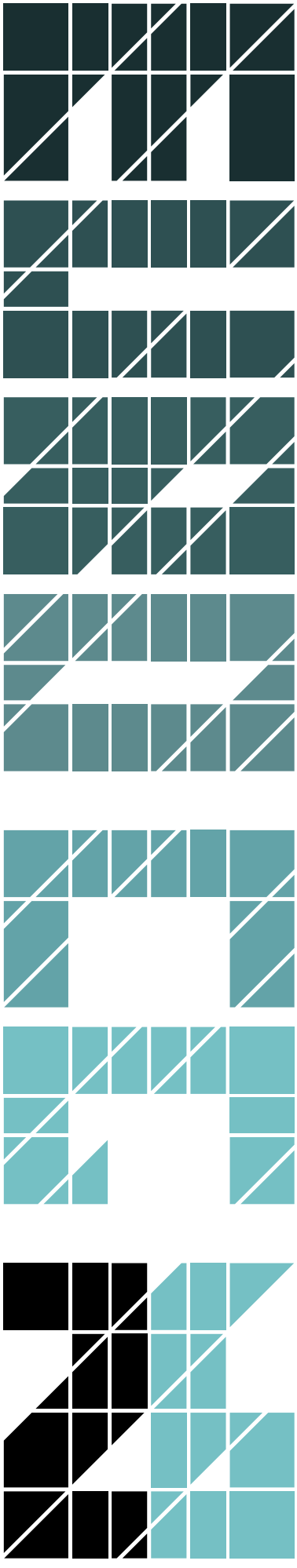


42nd European Workshop on Computational Geometry

Booklet of Extended Abstracts

March 25 – 27

Hagen, Germany

1 Preface

The 42nd European Workshop on Computational Geometry (EuroCG 2026) took place from March 25 to 27, 2026, in Hagen, Germany. As an annual, informal workshop, EuroCG provides a forum for scientists to present their work, interact, and establish collaborations, fostering research in Computational Geometry throughout Europe and beyond.

The scientific program consisted of 94 contributed talks selected from 104 submissions following a review process by the program committee. Since EuroCG does not have formally published proceedings, we expect most of the results outlined here to be submitted to peer-reviewed conferences or journals in the future. This collection of short abstracts, available via the EuroCG 2026 website, should therefore be regarded as a series of preprints. In addition to the contributed presentations, this volume contains abstracts from our invited speakers: Marcus Schaefer (DePaul University Chicago, USA), Maike Buchin (Ruhr University Bochum, Germany), and Jean Cardinal (Université libre de Bruxelles, Belgium).

The workshop hosted 154 registered participants. We would like to thank all authors for their contributions and express our gratitude to the members of the program committee and the external reviewers for their valuable feedback. We also thank the local organizing committee from the FernUniversität in Hagen: Jonathan Rollin, Alexandra Weinberger, Jona Handszuj, and Julia Oppermann, for their dedicated work. Furthermore, we are grateful for the generous financial support provided by the German Research Foundation (DFG), the "Freunde der FernUniversität" society, and the Faculty of Mathematics and Computer Science at the FernUniversität in Hagen, which also provided the necessary rooms and services.

During the workshop, attendees voted for the Best Student Presentation Award. The winner was Chek-Manh Loi for his outstanding presentation, "Drone Air Traffic Control: Tracking a Set of Moving Objects with Minimal Power." During the business meeting, Christian Rieck (University of Kassel) presented the 2027 edition of EuroCG, which will be held in Norrköping, Sweden. Ignaz Rutter (University of Passau) presented a bid to host EuroCG 2028 in Passau, which was formally accepted.

We look forward to seeing you all next year in Norrköping.

April 2026

Lena Schlipf and André Schulz

2 Program Committee

Mikkel Abrahamsen	University of Copenhagen
Eyal Ackerman	University of Haifa at Oranim
Hugo A. Akitaya	University of Massachusetts Lowell
Erin Chambers	University of Notre Dame
Simon D. Fink	Technische Universität Wien
Henry Förster	John Cabot University, Rome
Thekla Hamm	TU Eindhoven
Katharina Klost	Freie Universität Berlin
Jan Kratochvil	Charles University
Giuseppe Liotta	University of Perugia
André Nusser	CNRS, France
Eunjin Oh	POSTECH
Evanthia Papadopoulou	Università della Svizzera italiana
Irene Parada	Universitat Politècnica de Catalunya
Carolin Rehs	TU Dortmund
Christian Rieck	University of Kassel
Jonathan Rollin	FernUniversität in Hagen
Maria Saumell	Czech Technical University in Prague
Christian Scheffer	Bochum University of Applied Sciences
Lena Schlipf	Universität Tübingen
Christiane Schmidt	Linköping University
Patrick Schnider	ETH Zürich and University of Basel
André Schulz	FernUniversität in Hagen
Michiel Smid	Carleton University
Frank Staals	Utrecht University
Alexandra Weinberger	FernUniversität in Hagen
Carola Wenk	Tulane University
Alexander Wolff	Universität Würzburg
Yelena Yuditsky	Université libre de Bruxelles
Ji Zeng	Rényi Institute

3 Organizing Committee

Jona Handzsuj
Julia Oppermann
Jonathan Rollin (co-chair)
André Schulz
Alexandra Weinberger (co-chair)

4 External Reviewers

Ágoston, Péter
Aronov, Boris

Balko, Martin
Barequet, Gill
Baumann, Alexander
Bekos, Michael A.
Binucci, Carla
Blank, Lotte
Brandenburg, Franz
Brass, Peter
Brunck, Florestan
Brötzner, Anna
Büngener, Aaron

Červenková, Eliška

Da Lozzo, Giordano
Damásdi, Gábor
Depian, Thomas
Dorfer, Joseph
Džuklevski, Aleksa

Fan, Chenglin
Fekete, Sándor
Fiala, Jiri
Filtser, Omrit
Firbas, Alexander
Franco Sánchez, Víctor
Friemel, Jonas

Gregor, Petr
Grilli, Luca
Grosso, Fabrizio
Gärtner, Bernd

Hegemann, Tim
Hoang, Hung
Hofer, Anna
Huber, Stefan
Huemer, Clemens

Jilg, Dominik

Kalb, Antonia
Keldenich, Phillip
Keszegh, Balázs
Kleist, Linda
Klemz, Boris
Knorr, Kristin
Kolbe, Benedikt
Kosfeld, Ramin
Kramer, Peter
Ködmön, Lili

Lauerbach, Antonio
Lim, Jeck
Loi, Chek-Manh

Maheshwari, Anil
Majhi, Sushovan
Meijer, Lucas
Miller, Jacob
Moore, Kenneth
Mulzer, Wolfgang

Okada, Yuto
Oliver, Nico
Oppermann, Julia
Orthaber, Joachim

Paul, Rosna
Perk, Michael
Perz, Daniel
Polishchuk, Valentin

Qin, Demi

Reinhardt, Jan-Marc
Richter, Marena
Rote, Günter

Salazar, Gelasio
Scheele, Torben
Scheffler, Robert
Scherzer, Lena
Seybold, Martin P.
Sieper, Marie Diana
Silveira, Rodrigo
Suk, Andrew

Tang, Hadyn
Tappini, Alessandra
Tejel, Javier
Terziadis, Soeren
Tiwary, Hans Raj
Tkadlec, Josef
Tóth, Csaba
Tóth, Géza

Ueckerdt, Torsten

Vahrenhold, Jan
van der Horst, Thijs
van Wordragen, Geert
Verciani, Francesco

Wang, Elena
Wang, Zeyu
Wiese, Andreas
Willert, Max
Wintraecken, Mathijs
Wolf, Samuel
Wulf, Lasse

Zhang, Hanwen
Zink, Johannes

5 Table of Contributions

Invited talks

Calculating with Pennies and Marbles 11
Marcus Schaefer

A natural metric for curves - 35 years of Fréchet distance computation 12
Maike Buchin

Compact Representations 13
Jean Cardinal

Contributed talks

Tilt Automata: Gathering Particles With Uniform External Control 14
Sándor P. Fekete, Jonas Friemel, Peter Kramer, Jan-Marc Reinhardt, Christian Rieck and Christian Scheffer

Locally Correct Interleavings between Merge Trees 23
Thijs Beurskens, Tim Ophelders, Bettina Speckmann and Kevin Verbeek

Multi-Block Grids via Polycubes 30
Maxim Snoep, Stevie-Ray Janssen, Bettina Speckmann and Kevin Verbeek

Counting d-Dimensional Polycubes, Revisited 35
Gill Barequet and Tom Feldman

A 44-Point Configuration Not Coverable by Disjoint Unit Disks 42
Takuto Nakai and Shuya Bundo

Grounded String Representations of Series-Parallel Graphs without Transitive Edges .. 48
Sabine Cornelsen, Jan Kratochvíl, Miriam Münch, Giacomo Ortali, Alexandra Weinberger and Alexander Wolff

Planar Convex Obstacle Number of Trees 56
Emilio Di Giacomo, Carolina Haase, Philipp Kindermann and Giuseppe Liotta

General Visibility Graphy 64
Franz Brandenburg

Fréchet Distance in the Imbalanced Case 70
Lotte Blank

Solving the Contiguous Art Gallery Problem using Few Starting Points 77
Sarita de Berg, Jacobus Conradi, Ivor van der Hoog and Eva Rotenberg

Garment numbers of bi-colored point sets in the plane 83
Oswin Aichholzer, Helena Bergold, Simon D. Fink, Maarten Löffler, Patrick Schnider and Josef Tkadlec

Devil's Games and QR: Continuous Games complete for the First-Order Theory of the Reals 90
Lucas Meijer, Arnaud de Mesmay, Till Miltzow, Marcus Schaefer and Jack Stade

Signotopes Induce Unique Source Orientations on Grids 96
Sandro M. Roch

Structural Properties of Shortest Flip Sequences Between Plane Spanning Trees 103
Oswin Aichholzer, Joseph Dorfer, Peter Kramer, Christian Rieck and Birgit Vogtenhuber

On t-colorable k-plane drawings 111
Miriam Goetze, Michael Kaufmann and Soeren Terziadis

On the Computation of Schrijver's Kernels 120
Vincent Delecroix, Oscar Fontaine and Francis Lazarus

The Witness Unit Disk Representability Problem 126
Carolina Haase, Giuseppe Liotta, Maarten Löffler, Fabrizio Montecchiani, Alessandra Tappini and Soeren Terziadis

Guarding Offices with Maximum Dispersion 133
Sándor Fekete, Kai Kobbe, Dominik Krupke, Joseph Mitchell, Christian Rieck and Christian Scheffer

Sliding Cubes in Parallel 140
Hugo A. Akitaya, Joseph Dorfer, Peter Kramer, Christian Rieck, Gabriel Shahrouzi and Frederick Stock

Undirected TSP as a Constrained GSTP Variant 149
Yilmaz Arslanoğlu

Small Empty Cycles in Simple Drawings of K_n 157
Anna Hofer, Joachim Orthaber, Birgit Vogtenhuber and Alexandra Weinberger

Recognizing Subgraphs of Regular Tilings 165
Eliel Ingervo and Sándor Kisfaludi-Bak

Towards Computing Average Merge Tree Based on the Interleaving Distance 173
Elena Farahbakhsh Touli, Ingrid Hotz and Talha Bin Masood

On quadrilaterals in higher order Voronoi diagrams 182
Andrea de Las Heras-Parrilla, Clemens Huemer and Javier Tejel

Disproving two conjectures on the Hamiltonicity of Venn diagrams 191
Sofia Brenner, Linda Kleist, Torsten Mütze, Christian Rieck and Francesco Verciani

Bouquet: A Visualization Tool for Symmetry Sets and Vineyards 200
Erin Chambers, Christopher Fillmore, Shankha Shubhra Mukherjee, Rohit Roy, Elizabeth Stephenson and Mathijs Wintraecken

The Domino Problem is Decidable for Robust Tilesets 213
Nathalie Aubrun, Manon Blanc and Olivier Bournez

Drone Air Traffic Control: Tracking a Set of Moving Objects with Minimal Power 219
Sándor Fekete, Malte Hoffmann, Chek-Manh Loi and Michael Perk

Line Segment Visibility in Simple Polygons: Exact, Robust, Scalable Computation and Applications 227
Sándor Fekete, Prahlad Narasimhan Kasthurirangan, Phillip Keldenich, Fabian Kolhoff, Chek-Manh Loi and Michael Perk

Ordinal Geometric Thickness of Complete and Complete Bipartite Graphs 237
Patrizio Angelini, Michael Bekos, Luca Grilli and Aikaterini Maria Ntasiou

Oracle Separations for RPH 245
Lucas Meijer, Till Miltzow, Subhasree Patro and Thekla Hamm

Dynamic Level Planarity Testing 250
Jonathan Højlev, Simon D. Fink and Eva Rotenberg

Partitioning a Tile Arrangement for Construction by a Team of Robots 257
Sándor P. Fekete, Jonas Friemel, Prahlad Narasimhan Kasthurirangan, Ramin Kosfeld, Christian Scheffer and Arne Schmidt

Flipping Non-crossing Spanning Trees is NP-Hard 266
Havard Bjerkevik, Joseph Dorfer, Linda Kleist, Torsten Ueckerdt and Birgit Vogtenhuber

The Chromatic Dispersive Art Gallery Problem in Polyominoes 275
Anna Brötzner, Kien C. Huynh, Christiane Schmidt and Frederick Stock

Online Packing of Orthogonal Polygons 284
Tim Gerlach, Benjamin Hennies and Linda Kleist

Approximating Triangle Covers of Polygons 291
Linda Kleist and Lena Scherzer

A Framework for Dimension Reduction for Curves 300
Matthijs Ebbens, Jie Lu and Alexander Munteanu

Improved Approximation of Two Watchmen's Routes in Simple Polygons 306
Anna Brötzner, Bengt J. Nilsson and Christiane Schmidt

Simplex volumes in hyperplane arrangements 312
Koki Furukawa

What induces plane structures in complete graph drawings? 319
Alexandra Weinberger and Ji Zeng

Antipodal Pairs and Crossing Numbers of Complete Graphs 329
Stefan Felsner

Fréchet Distance for paths in a d-dimensional grid graphs 336
Ivor van der Hoog, Eva Rotenberg and Frederikke Uldahl

The Euclidean Minimum Spanning Tree Extension Problem (and its Approximation) .. 343
Emilio Di Giacomo, Giuseppe Liotta, Daniel Perz and Morteza Saghafian

Graph Tile Connectability with Turn Tiles 351
Maarten Löffler and Ids de Vlas

Dynamic $(1 + \epsilon)$-Spanner in Disk Intersection Graphs 358
Sarita de Berg, Ivor van der Hoog, Eva Rotenberg, Johanne M. Vistisen and Sampson Wong

Computing L_∞ Hausdorff Distances Under Translations: The Interplay of Dimensionality, Symmetry and Discreteness 365
Sebastian Angrick, Kevin Buchin, Geri Gokaj and Marvin Künnemann

On Small Pair Decompositions for Point Sets: Hardness and the 1D case 371
Kevin Buchin, Jacobus Conradi, Sariel Har-Peled, Antonia Kalb, Abhiruk Lahiri, Lukas Plätz, Carolin Rehs and Sampson Wong

On minimum Venn diagrams 378
Sofia Brenner, Petr Gregor, Torsten Mütze and Francesco Verciani

Edge Densities of Drawings of Graphs with One Forbidden Cell 387
Benedikt Hahn, Torsten Ueckerdt and Birgit Vogtenhuber

Fully Scalable MPC Algorithms for WSPD in Euclidean Spaces 396
Eunjin Oh and Hyeonjun Shin

Computing the Fréchet Distance When Just One Curve is c-Packed: A Simple Almost-Tight Algorithm 404
Jacobus Conradi, Ivor van der Hoog, Thijs van der Horst and Tim Ophelders

The number of occurrences of the two smallest distances 411
Cameron Strachan and Konrad Swanepoel

Approximating the Minmax Three-Visiting Routes for m Treasures in a Simple Polygon 418
Anna Brötzner, Bengt J. Nilsson and Christiane Schmidt

Point Set Transformations using Given Groups 425
Thomas C. Van Dijk, Erwin Glazenburg, Wouter Meulemans, Anna Schenfisch and Arjen Simons

On (Directed) Width-Parameters of Geometric Spanners 432
Kevin Buchin, Carolin Rehs and Torben Scheele

Explicit High-Chromatic Hypergraphs Realized by Axis-Parallel Rectangles 439
Gábor Damásdi

Star-Based Separators for Intersection Graphs of c-Colored Pseudo-Segments 445
Mark de Berg, Bart M. P. Jansen and Jeroen S.K. Lamme

Rupture-Isolation for the Weak Graph Distance 451
Maike Buchin, Wolf Kißler and Fabian Kubon

Reconfiguration of Squares Using a Constant Number of Moves Each 457
Thijs van der Horst, Maarten Löffler, Tim Ophelders and Tom Peters

Revisiting Graph Modification via Disk Scaling: From One Radius to Interval-Based Radii 465
Thomas Depian and Frank Sommer

The Parameterized Complexity of Geometric 1-Planarity 473
Alexander Firbas

Semi-discrete convex order and Laguerre tessellation fitting 479
David P. Bourne, Thomas O. Gallouët, Quentin Mérigot and Andrea Natale

Simultaneous Embedding of Two Paths on the Grid 488
Stephen Kobourov, William Lenhart, Guiseppe Liotta, Daniel Perz, Pavel Valtr and Johannes Zink

Online searching for rays in the half-plane 496
Florian Gans and Elmar Langetepe

Wataridori is NP-Complete 504
Suthee Ruangwises

Simplicial Approximation to CW Complexes with Spherical Delaunay Triangulations . . 513
Raphaël Tinarrage

Which Vertical Graphs are Non VPHT Reconstructible? 523
Jette Gutzeit, Kalani Kistler, Tim Ophelders and Anna Schenfisch

Shortest Paths, Convexity, and Treewidth in Regular Hyperbolic Tilings 531
Sándor Kisfaludi-Bak, Tze-Yang Poon and Geert van Wordragen

Topologically Stable Hough Transform 538
Stefan Huber, Kristóf Huszár, Michael Kerber and Martin Uray

A set of convex prototiles that allows only countably many different tilings 545
Aleksa Džuklevski

Lower Bounding the Number of Triangulations as a Function of the Convex Hull Size . 551
Justin Dallant

Geometric Give and Take 558
Oswin Aichholzer, Katharina Klost, Kristin Knorr, Viola Mészáros and Josef Tkadlec

Compatible triangulations of simple polygons 564
Peyman Afshani, Boris Aronov, Kevin Buchin, Maike Buchin, Otfried Cheong, Katharina Klost, Carolin Rehs and Günter Rote

Simple Topological Thickness 571
Michael Hoffmann, Julia Oppermann, Rosna Paul, Jonathan Rollin and Alexandra Weinberger

On the Pathwidth of 2-Layer k-Matching-Planar Graphs 578
Saeed Odak, Jonathan Rollin and Torben Scheele

High Beer Index Implies Big Hollow Triangles 585
Arun Kumar Das, Vít Jelínek, Jan Kynčl, Martin Pergel, Felix Schröder, Peter Stumpf and Pavel Valtr

On the Surjectivity of a Map by Kapranov and Voevodsky in four Dimensions 591
Yan Alves Radtke

Partitioning the boundary of an art gallery with visibility polygons 598
Robert Barish and Tetsuo Shibuya

Fault-Tolerance and Oriented Dilation of the Greedy Triangulation 605
Antonia Kalb, Kevin Buchin and Prosenjit Bose

Approximation Depth of Convex Polytopes 612
Egor Bakaev, Florestan Brunck and Amir Yehudayoff

Range Counting Oracles for Extent problems 619
Sanghwa Han and Eunjin Oh

How many times can two minimum spanning trees cross? 626
Todor Antić, Morteza Saghafian, Maria Saumell, Felix Schröder, Josef Tkadlec and Pavel Valtr

On geodesic disks enclosing many points 633
Prosenjit Bose, Guillermo Esteban, David Orden, Rodrigo Silveira and Tyler Tuttle

Centered flips of non-crossing, perfect matchings 640
Alexandra Wesolek and Lisa Kudlik

The Complexity of Extending Storylines with Minimum Local Crossing Number 648
Alexander Dobler, Siddharth Gupta, Philipp Kindermann, Fabrizio Montecchiani and Martin Nöllenburg

On the edge surplus of 1-planar unit distance graphs over matchstick graphs 656
Eliška Červenková and Jan Kratochvil

Finding Patient Zero via Low-Dimensional Geometric Embeddings 663
Stefan Huber and Dominik Kaaser

On the solvability of Shortest Descending Paths 672
Víctor Franco-Sánchez, Alex Herrero and Rodrigo I. Silveira

An FPT Algorithm for Maximum $k \times k$ Square Packing Parameterized by Remaining Space 679
Maarten Dankers, Thomas C. Van Dijk and Kevin Verbeek

Exploring Mixedness of Bichromatic Point Sets 650
Thijs Beurskens, Marc van Kreveld, Frank Staals and Jules Wulms

NP-Completeness Proofs of All or Nothing, Water Walk, and Remembered Length Using the T-Metacell Framework 692
Pakapim Eua-Anant, Papangkorn Apinyanon, Thunyatorn Jirachaisri, Nantapong Ruangsuksriwong and Suthee Ruangwises

Incremental k-lowest planes and planar k-nearest neighbor with optimal query time ... 706
John Iacono, Yakov Nekrich and Martin P. Seybold

On rectilinear drawings of the hypercube 713
Todor Antić, Niloufar Fuladi, Anna Margarethe Limbach and Pavel Valtr

Calculating with Pennies and Marbles

Marcus Schaefer[1]

1 **DePaul University Chicago**
mschaefer@cdm.depaul.edu

Abstract

The recognition problem for penny graphs, contact graphs of unit disks in the plane, has been known to be NP-hard since the work by Breu and Kirkpatrick in the 90s. In this talk we settle the complexity of the problem by showing it is exactly as hard as deciding truth in the existential theory of the reals (ER). We will also give an introduction and overview of the recent complexity class ER.

Joint work with Anna Lubiw.

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

A natural metric for curves – 35 years of Fréchet distance computation

Maike Buchin[1]

1 Ruhr University Bochum
maike.buchin@rub.de

Abstract

The Fréchet distance is arguably the most popular similarity measure for curves in computational geometry. It was first studied from an algorithmic perspective in 1991 by Helmut Alt and Michael Godau and has received considerable attention since then. In this talk I will outline some notable results in this line of research and end with some open problems.

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

Compact Representations

Jean Cardinal[1]

1 **Université libre de Bruxelles**
jean.cardinal@ulb.be

Abstract

In computational geometry, one often meets graphs representing various kinds of geometric relations: intersection, containment, incidence, visibility, etc. While these graphs have natural geometric representations, these representations might sometimes be costly. We will consider biclique decompositions of such graphs, that allow compact representations using o(n) bits per vertex. In particular, we will describe compact representations of semialgebraic graphs - including segment and disk intersection graphs, semilinear graphs - including intersection graphs of rectilinear geometric objects, and various flavors of visibility graphs. We will argue that such representations are practical alternatives to the usual adjacency list representations. The presentation will involve classical and more recent results, including a joint work with Yelena Yuditsky (ESA 2025).

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

Tilt Automata: Gathering Particles With Uniform External Control

Sándor P. Fekete[*1], Jonas Friemel[*2], Peter Kramer[*3], Jan-Marc Reinhardt[2], Christian Rieck[†4], and Christian Scheffer[*2]

1 **Department of Computer Science, TU Braunschweig and L3S, Germany**
s.fekete@tu-bs.de
2 **Department of Electrical Engineering and Computer Science, Bochum University of Applied Sciences, Germany**
{jonas.friemel,jan-marc.reinhardt,christian.scheffer}@hs-bochum.de
3 **Department of Computer Science, TU Braunschweig, Germany**
kramer@ibr.cs.tu-bs.de
4 **Institute of Mathematics, University of Kassel, Germany**
christian.rieck@mathematik.uni-kassel.de

Abstract

Motivated by targeted drug delivery, we investigate the gathering of particles in the full tilt model of externally controlled motion planning: A set of particles is located at the tiles of a polyomino. They react uniformly to an external force by moving maximally in one of the four axis-parallel directions until they hit the boundary. The goal is to choose a sequence of directions that moves all particles to a common position. We show how to compute gathering sequences for filled polyominoes and prove it PSPACE-complete to determine if the particles in a partially filled polyomino can be gathered. The results build on a connection we establish between tilt models and synchronizing automata.

Related Version *Full Version*: arXiv:2603.02796 [13]

1 Introduction

We investigate a problem motivated by targeted drug delivery [15]: How can initially dispersed drug-carrying particles be gathered at a common position? In micro-scale robotics [16], environmental constraints and the severely limited capabilities of the robotic agents rule out autonomous movement. Instead, a global external force, e.g., an electromagnetic field, can be used to move and steer simple robotic particles. The challenge is to use geometry to control a large swarm of particles through uniform signals within a non-uniform environment [3, 7].

Abstract models [2, 8], commonly called *tilt models*, have been developed to tackle this problem algorithmically. They model the environment as a *polyomino*, i.e., a finite, connected region of the square tiling, with particles positioned at tiles and moving on axis-parallel paths. Two main variants exist: In the *single step model* (S1), particles move via unit steps to adjacent positions, whereas in the *full tilt model* (FT), they move maximally until blocked.

Algorithms and complexity results for a variety of applications [6, 9, 14, 17] have been discovered using tilt models. We consider particles located at the tiles of a polyomino and look for a *gathering sequence*, i.e., a sequence of moves that relocates all particles to the same tile. In S1, gathering sequences of length $\mathcal{O}(n_c D^2)$ always exist, where n_c is the number of convex corners of the polyomino and D its diameter, but finding sequences of minimum length is NP-hard [4]. On the other hand, very little is known about gathering in FT.

[*] Supported by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – 530918134.
[†] Supported by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – 522790373.

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

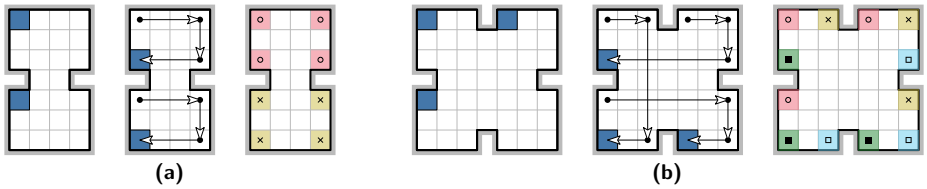


Figure 1 Two examples where particles cannot be gathered. Both depict, from left to right, a set of particles in a polyomino, moves leading to all possible configurations, and congruent positions.

Observe that particles cannot always gather in FT. The authors of [18] note that particles may be trapped in separate regions, see Figure 1(a). But gathering can even be impossible if all tiles reachable by the initial particles are mutually reachable, see Figure 1(b).

Our contributions. Let a polyomino have n corners; n_c are convex. We show the following.

- If particles are initially positioned at every tile of a polyomino, a gathering sequence of length $\mathcal{O}(n_c n^2)$ can be computed in $\mathcal{O}(n_c n^2)$ time, if one exists at all (Theorem 3.2).
- There are polyominoes with shortest gathering sequences of length $\Omega(n^2)$ (Theorem 3.3).
- It is PSPACE-complete to decide if a subset of particles can be gathered (Theorem 4.3).

We utilize a previously overlooked connection between tilt models and synchronizing automata. We expect that more insights for tilt models can be gained by exploring this approach further. Conversely, the geometry of tilt models may lead to new ways to tackle open questions in automata theory. The discrepancy between the bounds on the worst-case length of gathering sequences is related to a longstanding open problem known as Černý's conjecture, see Theorem 4.2.

Due to limited space, proofs of statements marked by ($\star$) can be found in the full version [13]. Additionally, the full version contains an extensive overview of related literature and several additional results on approximating shortest gathering sequences, gathering subsets in hole-free polyominoes, and the parameterized complexity of tilt problems.

2 Preliminaries

In this section, we give the rigorous definitions omitted from Section 1 in favor of intuition. A polyomino P has an associated *dual graph* $G_P = (V, E)$, see Figure 2(b), and a *boundary* ∂P, an axis-aligned polygon separating V from $\mathbb{Z}^2 \setminus V$. We identify the vertices of V with the tiles of P and call them *pixels*. As a convention, n denotes the number of corners of ∂P, $n_c \leq n$ the number of convex corners, and $N = |V|$ the number of pixels of a given polyomino. Pixels are uniquely determined as the intersection of two *segments*, a *row segment* and a *column segment*, which are maximally contiguous parts of rows and columns; a *corner pixel* is adjacent to two perpendicular sides of ∂P, i.e., next to a convex corner, see Figure 2(a).

A polyomino is *simple* if it contains no holes. We also consider a supplementary class of polyominoes we call *mazes*: collections of horizontal and vertical segments that only intersect in T-shaped and +-shaped junctions. Formally, a maze is a *thin* polyomino (it contains no 2×2 squares) in which all corner pixels have degree 1, see Figure 2(b).

A set $C \subseteq V$ of all pixels containing particles is called a *configuration*. Pixels $p \in C$ are *occupied*. We use the set $\mathbb{D} = \{\text{U}, \text{D}, \text{L}, \text{R}\}$ as shorthand for the directions up, down, left, and right. Formally, a tilt model defines a function $\delta : \mathbb{D} \to (2^V \to 2^V)$, i.e., it maps directions to transformations of configurations. A transformation $\delta(v)$, $v \in \mathbb{D}$, is called a *move*. In

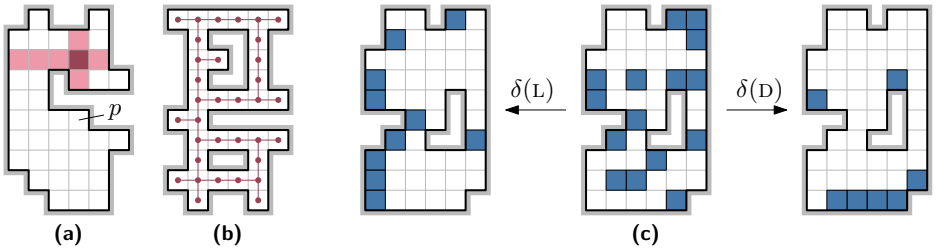


Figure 2 (a) A simple polyomino with a darkly shaded pixel at the intersection of two segments and a corner pixel p. **(b)** A non-simple maze and its dual graph. **(c)** Leftward and downward moves.

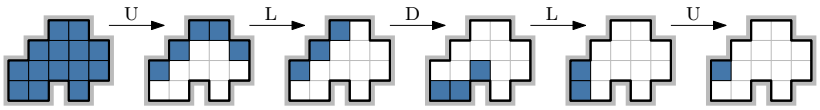


Figure 3 The sequence ULDLU is a gathering sequence for this polyomino.

this work, we consider the merging variant of the full tilt model, where particles merge upon colliding. For the direction L, $\delta(\text{L})$ is defined as follows: $p \in \delta(\text{L})(C)$ if and only if p is the leftmost pixel of the row segment R containing p and $R \cap C \neq \varnothing$. Definitions for the other directions are analogous. We extend δ to a function on $\mathbb{D}^*$, the set of sequences of directions, by setting $\delta(wv) = \delta(v) \circ \delta(w)$, for $v \in \mathbb{D}$ and $w \in \mathbb{D}^*$, and $\delta(\varepsilon)(C) = C$, where ε is the empty sequence. For simplicity, we may write $C \cdot w$ instead of $\delta(w)(C)$ and $C \xrightarrow{w} C'$ instead of $C' = \delta(w)(C)$, and use moves $\delta(v)$ and directions $v \in \mathbb{D}$ interchangeably. For singleton configurations, we define $\delta^1 : \mathbb{D}^* \to (V \to V)$ as $\delta^1(w)(p) = q$ whenever $\delta(w)(\{p\}) = \{q\}$. The uniform movement of particles allows us to reduce the behavior of arbitrary configurations to the motion of single particles.

▶ **Observation 2.1.** *For $w \in \mathbb{D}^*$ and $C \subseteq V$, $\delta(w)(C) = \bigcup_{p \in C} \{\delta^1(w)(p)\}$.*

A *gathering sequence for a configuration* C is a sequence of directions $w \in \mathbb{D}^*$ such that $|C \cdot w| = 1$, in which case C is a *gatherable configuration*. Moreover, a *gathering sequence for a polyomino* P is a gathering sequence for the configuration V, see Figure 3, making P a *gatherable polyomino*. We define $\text{sgs}(C)$ to be the length of a *shortest gathering sequence* for a gatherable configuration C, and assign $\text{sgs}(P) = \text{sgs}(V)$ for a gatherable polyomino P.

We investigate two problems in the full tilt model for a given polyomino P.

- FullGathering: Find a gathering sequence for P or decide that P is not gatherable.
- SubsetGathering: Decide if a given configuration $C \subseteq V$ is gatherable.

3 Gathering particles by synchronizing automata

The notation introduced in Section 2 may be considered unnecessarily heavy, but it highlights the connection between tilt models and automata. An *automaton* is a triple (Q, Σ, δ), where Q is a finite, non-empty set of *states*, Σ is a finite, non-empty *alphabet*, and $\delta : \Sigma \to (Q \to Q)$ is a *transition function* mapping letters of Σ to transformations of Q. Again, δ gets extended to a function over *words* $w \in \Sigma^*$. Now, a tilt model defines an automaton $(2^V, \mathbb{D}, \delta)$ for a polyomino P with dual graph $G_P = (V, E)$. This approach shows its merit when we consider the analogue to gathering sequences, synchronizing words.

A word $w \in \Sigma^*$ is a *synchronizing word* for a set $Q' \subseteq Q$ if there is a state $q \in Q$ such that $p \xrightarrow{w} q$ for all $p \in Q'$, in which case Q' is a *synchronizing set*. The minimum length

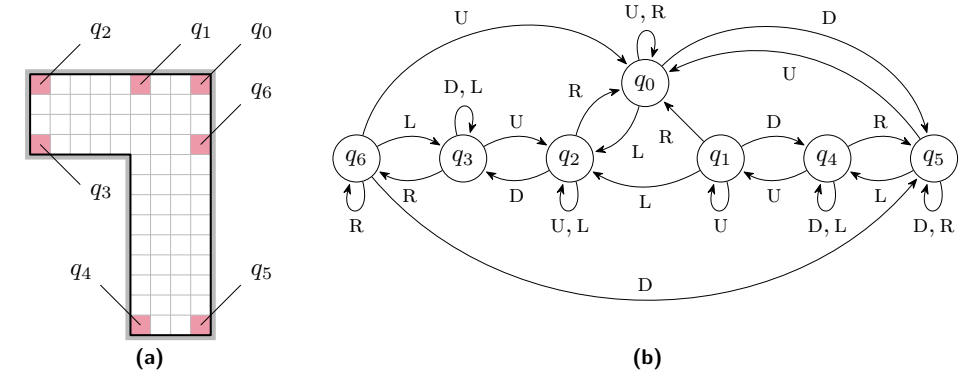


Figure 4 (a) A polyomino P and its significant pixels. **(b)** The full tilt automaton $A(P)$.

of a synchronizing word for a synchronizing set Q' is called the *reset threshold* $\mathrm{rt}(Q')$. An automaton $A = (Q, \Sigma, \delta)$ is synchronizing if Q is, in which case we define $\mathrm{rt}(A) = \mathrm{rt}(Q)$.

Finding a synchronizing word of size $\mathcal{O}(|Q|^3)$, if one exists, can be done in polynomial time, e.g., with Eppstein's algorithm [12]. Černý's conjecture, named in honor of foundational work by Černý [11], states that $\mathrm{rt}(A) \leq (|Q|-1)^2$ holds for every synchronizing automaton A. More details on synchronizing automata can be found in several excellent surveys [21, 24, 25].

Observation 2.1 tells us that a gathering sequence for a polyomino P corresponds precisely to a synchronizing word for the automaton $(V, \mathbb{D}, \delta^1)$. Thus, we can solve FullGathering by applying Eppstein's algorithm [12] to $(V, \mathbb{D}, \delta^1)$ to decide if it is synchronizing in $\mathcal{O}(N^2)$ time and, if so, find a gathering sequence for P in $\mathcal{O}(N^3)$ time. We now sketch algorithms with improved running times of $\mathcal{O}(n^2)$ and $\mathcal{O}(n_c n^2)$, respectively, that use only ∂P as input.

The key observation is that δ very quickly reduces the set of occupied pixels to one of size $\mathcal{O}(n)$. Consider the set $S \subseteq V$ of *significant pixels* that contains all corner pixels and possibly *helper pixels*, which are found by extending sides of ∂P that meet at reflex corners until they hit another side of ∂P. Figure 4(a) illustrates the significant pixels of a polyomino P with helper pixels q_1 and q_6. Observe that $|S| \leq 2n - n_c$ because at most two helper pixels are included per reflex corner. Any sequence of two moves in perpendicular directions changes the position of a particle from any pixel $p \in V$ to one in S, e.g., $\delta^1(\mathrm{DL})(p) \in S$. S is closed under δ^1, i.e., after the move sequence DL we can focus on δ^1 restricted to S. We call the resulting automaton $A(P) = (S, \mathbb{D}, \delta^1)$ the *full tilt automaton* of P, see Figure 4(b).

▶ **Lemma 3.1.** *A polyomino P is gatherable if and only if the full tilt automaton $A(P)$ is synchronizing, in which case* $\mathrm{rt}(A(P)) \leq \mathrm{sgs}(P) \leq \mathrm{rt}(A(P)) + 1$ *holds.*

Proof. A gathering sequence for P is a synchronizing word for $A(P)$ because $S \subseteq V$. Conversely, assume w is a synchronizing word for S. If $w = \varepsilon$, then P is a single pixel and ε a gathering sequence. Otherwise, assume without loss of generality that $w = \mathrm{L}w'$ for some $w' \in \mathbb{D}^*$ (rotate P if necessary). Then $\mathrm{DL}w \in S$ and $\mathrm{DL}w = \mathrm{DLL}w'$ is a gathering sequence for P, which can be shortened to $\mathrm{DL}w' = \mathrm{D}w$ because moves in the full tilt model are idempotent, i.e., $\delta(v) \circ \delta(v) = \delta(v)$ for all $v \in \mathbb{D}$. ◀

It is a well-known fact [19, 25] that an automaton (Q, Σ, δ), $|Q| > 1$, is synchronizing if and only if every unordered pair of states $\{s, t\} \subseteq Q$ has a synchronizing word, suggesting a strategy of iteratively merging pairs. To efficiently determine synchronizing words for pairs in $A(P)$, we construct the *pair automaton* $A(P)^{\leq 2} = (S^{\leq 2}, \mathbb{D}, \delta^{\leq 2})$, where $S^{\leq 2} = \{\{p, q\} :$

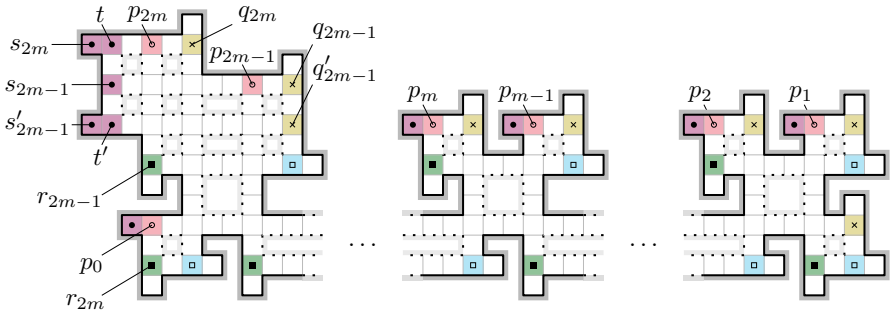


Figure 5 A family of polyominoes P_m with $\mathrm{sgs}(P_m) \in \Omega(n^2)$. Important classes of pixels are marked with distinct colors and symbols. Without the dotted areas, the polyominoes become mazes.

$p, q \in S\}$ and $\delta^{\leq 2}(v)(\{p,q\}) = \{\delta^1(v)(p)\} \cup \{\delta^1(v)(q)\}$. $S^{\leq 2}$ also contains all singletons $\{p\} \subseteq S$. A word w with $p \cdot w = q \cdot w$ in $A(P)$ corresponds precisely to a path labeled w from $\{p,q\}$ to $\{p \cdot w\}$ in the digraph underlying $A(P)^{\leq 2}$. Using BFS on this digraph allows us to compute the next step on a shortest path to a singleton for every pair of states, if one exists.

To cut down the number of pairs that need to be considered, we initially reduce S to convex corner pixels by repeatedly applying two perpendicular moves, e.g., DL. Observe that if $p \in S$ is a helper pixel, then $p \cdot \mathrm{DL} < p$ in terms of their lexicographic order, as at least one coordinate decreases. Thus, $\mathcal{O}(n)$ applications of DL suffice to eliminate all helper pixels (and all corner pixels not at lower-left corners) from S and $\mathcal{O}(n_c)$ pairs remain.

▶ **Theorem 3.2** (⋆). *Given the boundary ∂P of a polyomino P, one can decide in $\mathcal{O}(n^2)$ time if P is gatherable, and, if so, compute a gathering sequence of length $\mathcal{O}(n_c n^2)$ in $\mathcal{O}(n_c n^2)$ time.*

▶ **Theorem 3.3** (⋆). *There is an infinite family P_m, $m \in \mathbb{N}$, of gatherable simple polyominoes such that $\mathrm{sgs}(P_m) \in \Omega(n^2)$, where the number of corners n grows with m.*

Proof sketch. The basic idea of the construction, depicted in Figure 5, is to place two particles on a directed cycle of length $2m+1$, spaced apart by m positions. Moving against the direction of the cycle leads to dead ends, except at p_{2m-1}. There, LU allows to skip one step on the cycle, which can be used to reduce the distance between the particles by 1. To reduce the distance again, the whole cycle has to be traversed, resulting in $\Omega(m)$ iterations of $\Omega(m)$ moves to gather the particles, for a total of $\Omega(m^2) = \Omega(n^2)$ moves. ◀

4 Simulating binary automata with uniformly controlled particles

If Černý's conjecture is true, then the $\Omega(n^2)$ lower bound on $\mathrm{sgs}(P)$ is actually tight.

▶ **Conjecture 4.1.** *For every gatherable polyomino P with n corners,* $\mathrm{sgs}(P) \in \mathcal{O}(n^2)$.

The geometric structure of polyominoes should make Conjecture 4.1 easier to prove than Černý's conjecture. Alas, we must dampen this hope, because particles in polyominoes can be used to simulate binary automata, i.e., automata with alphabet $\Sigma = \{0,1\}$.

To simulate an automaton $A = (Q, \{0,1\}, \delta)$, the main idea is to build a polyomino $P(A)$ that has every state $q \in Q$ represented by a pixel $\overline{q}$, and place a single particle whose presence at $\overline{q}$ signifies that A is in state q. We use the clockwise cycle URDL and the counter-clockwise cycle DRUL to express transitions for 0 and 1, respectively. By carefully constructing the boundary of $P(A)$, we guarantee that all minimal sequences of moves between two pixels $\overline{q_i}$

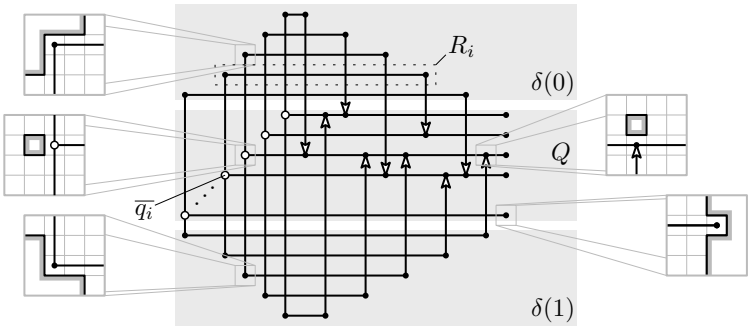


Figure 6 Schematic overview of the construction of $P(A)$ from a binary automaton A. Three blocks represent, from top to bottom, transitions for 0, states $q_i \in Q$, and transitions for 1.

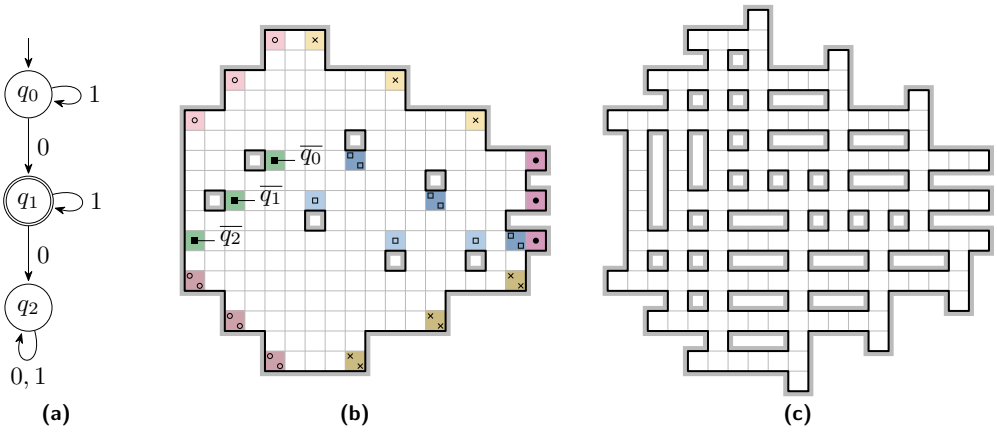


Figure 7 (a) The binary automaton A_0. **(b)** The polyomino $P(A_0)$. Colors and symbols indicate congruence classes of pixels reachable from representatives $\overline{q_i}$. **(c)** The maze $M(A_0)$.

and $\overline{q_j}$ can be decomposed into combinations of URDL and DRUL, thus ensuring that minimal sequences of moves between pixels representing states correspond to transitions in A.

Without loss of generality, let $Q = \{q_0, q_1, \ldots, q_{|Q|-1}\}$, and set $\overline{Q'} = \{\overline{q} : q \in Q'\}$ for sets $Q' \subseteq Q$. Our construction of $P(A)$ is illustrated in Figure 6. Every state $q_i \in Q$ gets assigned a row of width $6|Q|$ including the representative pixel $\overline{q_i}$. These rows are stacked, with left-aligned rows of width $6|Q| - 1$ in between, and the $\overline{q_i}$ are aligned on an upward diagonal, from $\overline{q_{|Q|-1}}$ in the bottom left up to $\overline{q_0}$. A 1×1 hole is placed to the left of every $\overline{q_i}$.

To implement the transitions $\delta(0)(q_i)$ in the top block, rows R_i of width $6i+3$ are stacked from top to bottom and aligned on the left with $\overline{q_i}$. A 1×1 hole is placed below the pixel at the intersection of the rightmost column of R_i and the row of $\overline{q_j}$ when $q_j = \delta(0)(q_i)$. This allows us to simulate a transition for 0 by the sequence of moves URDL. Similar rows and holes are constructed for $\delta(1)(q_i)$ in the bottom block from bottom to top, allowing us to simulate 1 by DRUL. $P(A)$ has $n = 12(2|Q| - 1)$ corners. An example is shown in Figure 7.

▶ **Theorem 4.2** (⋆). *A set $S \subseteq Q$ of states of an automaton $A = (Q, \{0, 1\}, \delta)$ is synchronizing if and only if its set of representatives $\overline{S}$ is gatherable in $P(A)$, in which case* $\mathrm{sgs}(\overline{S}) = 4\,\mathrm{rt}(S)$.

Thus, a quadratic upper bound for $\mathrm{sgs}(\overline{S})$ yields a quadratic upper bound for $\mathrm{rt}(S)$. While this would not settle Černý's conjecture, it would be remarkable progress. The existence of such an upper bound is still open, suggesting that a proof for Conjecture 4.1 may be elusive.

We can also adapt the construction of $P(A)$ to get a maze $M(A)$, possibly with $\Omega(|Q|^2)$ corners, by adding more holes as well as pixels of degree 1 next to corner pixels, see Figure 7(c).

▶ **Theorem 4.3.** SubsetGathering *is* PSPACE*-complete, even in mazes.*

Proof. It is easy to see that a non-deterministic algorithm for SubsetGathering could run in polynomial space, which implies containment in NPSPACE = PSPACE [23].

We reduce from SubsetSynchronizability, which is PSPACE-hard [20] even for binary automata [27]: Given a binary automaton $A = (Q, \{0, 1\}, \delta)$ and a set $S \subseteq Q$, determine if S is synchronizing. We construct $P(A)$ for a given binary automaton $A = (Q, \{0, 1\}, \delta)$ in polynomial time. By Theorem 4.2, $S \subseteq Q$ is synchronizing if and only if $\overline{S}$ is gatherable. This property remains true when $P(A)$ is turned into a maze $M(A)$. ◀

Theorem 4.3 also follows from the PSPACE-hardness proof for the so-called Occupancy problem from [1], but we welcome an alternative proof in line with our automata theme.

References

1 Jose Balanza-Martinez, Timothy Gomez, David Caballero, Austin Luchsinger, Angel A. Cantu, Rene Reyes, Mauricio Flores, Robert T. Schweller, and Tim Wylie. Hierarchical shape construction and complexity for slidable polyominoes under uniform external forces. In *Symposium on Discrete Algorithms (SODA)*, pages 2625–2641, 2020. doi:10.1137/1.9781611975994.160.

2 Aaron T. Becker, Erik D. Demaine, Sándor P. Fekete, Golnaz Habibi, and James McLurkin. Reconfiguring massive particle swarms with limited, global control. In *Symposium on Algorithms and Experiments for Sensor Systems, Wireless Networks and Distributed Robotics (ALGOSENSORS)*, pages 51–66, 2013. doi:10.1007/978-3-642-45346-5_5.

3 Aaron T. Becker, Erik D. Demaine, Sándor P. Fekete, Hamed Mohtasham Shad, and Rose Morris-Wright. Tilt: The video – designing worlds to control robot swarms with only global signals. In *International Symposium on Computational Geometry (SoCG)*, pages 16–18, 2015. doi:10.4230/LIPICS.SOCG.2015.16.

4 Aaron T. Becker, Sándor P. Fekete, Li Huang, Phillip Keldenich, Linda Kleist, Dominik Krupke, Christian Rieck, and Arne Schmidt. Targeted drug delivery: Algorithmic methods for collecting a swarm of particles with uniform, external forces. In *International Conference on Robotics and Automation (ICRA)*, pages 2508–2514, 2020. doi:10.1109/ICRA40945.2020.9196551.

5 Aaron T. Becker, Sándor P. Fekete, Phillip Keldenich, Dominik Krupke, Christian Rieck, Christian Scheffer, and Arne Schmidt. Tilt assembly: Algorithms for micro-factories that build objects with uniform external forces. In *International Symposium on Algorithms and Computation (ISAAC)*, pages 11:1–11:13, 2017. doi:10.4230/LIPIcs.ISAAC.2017.11.

6 Aaron T. Becker, Sándor P. Fekete, Phillip Keldenich, Dominik Krupke, Christian Rieck, Christian Scheffer, and Arne Schmidt. Tilt assembly: Algorithms for micro-factories that build objects with uniform external forces. *Algorithmica*, 82(2):165–187, 2020. A conference version related to this paper appeared at ISAAC 2017 [5]. doi:10.1007/S00453-018-0483-9.

7 Aaron T. Becker, Golnaz Habibi, Justin Werfel, Michael Rubenstein, and James McLurkin. Massive uniform manipulation: Controlling large populations of simple robots with a common input signal. In *International Conference on Intelligent Robots and Systems (IROS)*, pages 520–527, 2013. doi:10.1109/IROS.2013.6696401.

8 David Caballero, Angel A. Cantu, Timothy Gomez, Austin Luchsinger, Robert T. Schweller, and Tim Wylie. Hardness of reconfiguring robot swarms with uniform external control in limited directions. *Journal of Information Processing*, 28:782–790, 2020. doi:10.2197/IPSJJIP.28.782.

9 David Caballero, Angel A. Cantu, Timothy Gomez, Austin Luchsinger, Robert T. Schweller, and Tim Wylie. Fast reconfiguration of robot swarms with uniform control signals. *Natural Computing*, 20(4):659–669, 2021. doi:10.1007/S11047-021-09864-0.

10 Ján Černý. Poznámka k homogénnym experimentom s konečnými automatmi. *Matematicko-fyzikálny časopis*, 14(3):208–216, 1964. URL: https://dml.cz/dmlcz/126647.

11 Ján Černý. A note on homogeneous experiments with finite automata. *Journal of Automata, Languages and Combinatorics*, 24(2–4):123–132, 2019. The original article was published in 1964 in Slovak [10]. English translation by Markus Holzer and Bianca Truthe. doi:10.25596/jalc-2019-123.

12 David Eppstein. Reset sequences for monotonic automata. *SIAM Journal on Computing*, 19(3):500–510, 1990. doi:10.1137/0219033.

13 Sándor P. Fekete, Jonas Friemel, Peter Kramer, Jan-Marc Reinhardt, Christian Rieck, and Christian Scheffer. Tilt automata: Gathering particles with uniform external control. *CoRR*, 2026. arXiv:2603.02796.

14 Phillip Keldenich, Sheryl Manzoor, Li Huang, Dominik Krupke, Arne Schmidt, Sándor P. Fekete, and Aaron T. Becker. On designing 2D discrete workspaces to sort or classify polyominoes. In *International Conference on Intelligent Robots and Systems (IROS)*, pages 1–9, 2018. doi:10.1109/IROS.2018.8594150.

15 Matthias Konitzny, Yitong Lu, Julien Leclerc, Sándor P. Fekete, and Aaron T. Becker. Gathering physical particles with a global magnetic field using reinforcement learning. In *International Conference on Intelligent Robots and Systems (IROS)*, pages 10126–10132, 2022. doi:10.1109/IROS47612.2022.9982256.

16 Fabian C. Landers, Lukas Hertle, Vitaly Pustovalov, Derick Sivakumaran, Cagatay M. Oral, Oliver Brinkmann, Kirstin Meiners, Pascal Theiler, Valentin Gantenbein, Andrea Veciana, Michael Mattmann, Silas Riss, Simone Gervasoni, Christophe Chautems, Hao Ye, Semih Sevim, Andreas D. Flouris, Josep Puigmartí-Luis, Tiago Sotto Mayor, Pedro Alves, Tessa Lühmann, Xiangzhong Chen, Nicole Ochsenbein, Ueli Moehrlen, Tilman Schubert, Zsolt Kulcsar, Philipp Gruber, Miriam Weisskopf, Quentin Boehler, Salvador Pané, and Bradley J. Nelson. Clinically ready magnetic microrobots for targeted therapies. *Science*, 390(6774):710–715, 2025. doi:10.1126/science.adx1708.

17 Arun Mahadev, Dominik Krupke, Sándor P. Fekete, and Aaron T. Becker. Mapping and coverage with a particle swarm controlled by uniform inputs. In *International Conference on Intelligent Robots and Systems (IROS)*, pages 1097–1104, 2017. doi:10.1109/IROS.2017.8202280.

18 Arun V. Mahadev, Dominik Krupke, Jan-Marc Reinhardt, Sándor P. Fekete, and Aaron T. Becker. Collecting a swarm in a grid environment using shared, global inputs. In *International Conference on Automation Science and Engineering (CASE)*, pages 1231–1236, 2016. doi:10.1109/COASE.2016.7743547.

19 Balas K. Natarajan. An algorithmic approach to the automated design of parts orienters. In *Symposium on Foundations of Computer Science*, pages 132–142, 1986. doi:10.1109/SFCS.1986.5.

20 Igor K. Rystsov. Polynomial complete problems in automata theory. *Information Processing Letters*, 16(3):147–151, 1983. doi:10.1016/0020-0190(83)90067-4.

21 Sven Sandberg. Homing and synchronizing sequences. In *Model-Based Testing of Reactive Systems*, pages 5–33, 2004. doi:10.1007/11498490_2.

22 Walter J. Savitch. Deterministic simulation of non-deterministic turing machines (detailed abstract). In *Symposium on Theory of Computing (STOC)*, pages 247–248, 1969. doi:10.1145/800169.805439.

23 Walter J. Savitch. Relationships between nondeterministic and deterministic tape complexities. *Journal of Computer and System Sciences*, 4(2):177–192, 1970. A conference version of this paper appeared at STOC 1969 [22]. doi:10.1016/S0022-0000(70)80006-X.

24 Mikhail V. Volkov. Synchronizing automata and the Černý conjecture. In *Language and Automata Theory and Applications (LATA)*, pages 11–27, 2008. doi:10.1007/978-3-540-88282-4_4.

25 Mikhail V. Volkov. Synchronization of finite automata. *Russian Mathematical Surveys*, 77(5):819–891, 2022. doi:10.4213/rm10005e.

26 Vojtěch Vorel. Subset synchronization of transitive automata. In *Automata and Formal Languages (AFL)*, pages 370–381, 2014. doi:10.4204/EPTCS.151.26.

27 Vojtěch Vorel. Subset synchronization and careful synchronization of binary finite automata. *International Journal of Foundations of Computer Science*, 27(5):557–578, 2016. A conference version of this paper appeared at AFL 2014 [26]. doi:10.1142/S0129054116500167.

Locally Correct Interleavings between Merge Trees

Thijs Beurskens, Tim Ophelders, Bettina Speckmann, and Kevin Verbeek

Dept. Mathematics and Computer Science, TU Eindhoven, The Netherlands
[t.p.j.beurskens|t.a.e.ophelders|b.speckmann|k.a.b.verbeek]@tue.nl

Abstract

To analyze time-varying terrains or scalar fields efficiently, one generally needs an abstraction as well as a method to compare and match them over time. In this paper we consider merge trees as the method of abstraction, and the interleaving distance as a method to match and compare them. The interleaving distance is defined in terms of interleavings, which is a pair of maps between two merge trees. Due to the bottleneck nature of the interleaving distance, these maps fail to capture local similarities between the trees. We hence propose a notion of local optimality for interleavings; we call the resulting interleavings *locally correct interleavings*. We give a constructive proof that a locally correct interleaving always exists.

Related Version arXiv:2512.16474

1 Introduction

Terrains, or more generally scalar fields, that vary over time are a common data type in various application areas, such as scientific computing or geographic information science. Topological data analysis (TDA) provides so-called topological descriptors to analyze such data sets efficiently [13]. In this paper we focus in particular on *merge trees*: graph-based descriptors of the evolution of connected components of sublevel sets of a scalar field (see Figure 1). To analyze temporal sequences of merge trees one additionally needs methods to compare and match them over time. For merge trees, several distance measures have been proposed, including the interleaving distance [6], the Wasserstein distance [9], and the merge tree edit distance [10]. However, a distance measure returns only a numeric value that quantifies the similarity of two merge trees T_1 and T_2, but in many application scenarios, we are interested in a matching between T_1 and T_2.

The interleaving distance relies on a type of matching between the two trees, the so-called *interleaving*. Two merge trees typically admit many optimal interleavings. The interleaving distance is a bottleneck measure and hence a given optimal interleaving might be locally quite "loose", and thereby fail to capture local similarities between the trees; see Figure 2.

Contributions. Inspired by the concept of locally correct Fréchet matchings for curves [3], we introduce a notion of locally optimal interleavings which allows us to distinguish between

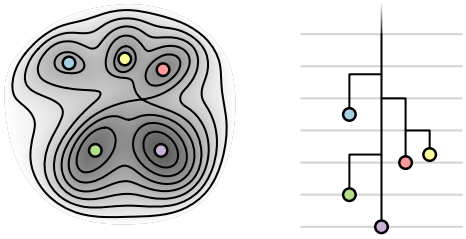


Figure 1 A terrain and its merge tree; leaves and internal vertices represent minima and saddles.

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

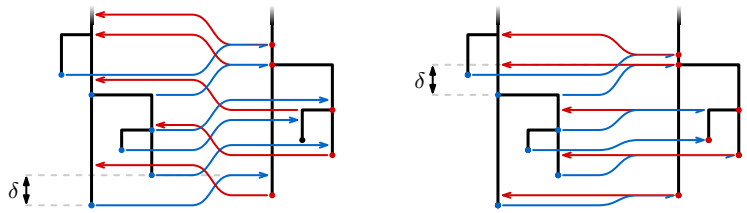


Figure 2 Two optimal interleavings between two merge trees. Intuitively, the right interleaving induces a tighter and more meaningful matching than the left interleaving.

"tight" and "loose" optimal interleavings. In particular, we generalize the interleaving distance to accommodate for constraints on the allowed maps: we use constraints that force a point x to be mapped to an ancestor of a particular point y within the other tree. We refer to the resulting distance as the *residual interleaving distance*. We build upon the residual interleaving distance to define *locally correct interleavings*, which we show always exist.

We review the interleaving distance in Section 2. In Section 3, we introduce a novel generalization of the interleaving distance that accommodates constraints on the maps. In Section 4 we define locally correct interleavings and constructively prove that a locally correct interleaving always exists. All omitted proofs can be found in the full version on ArXiv.

Related work. Morozov *et al.* [6] define the interleaving distance between merge trees as a means to study the stability of merge trees. Agarwal *et al.* [1] give the first (approximation) algorithm to compute the distance, and they argue that approximating the interleaving distance within a factor of 3 is NP-hard. The first—and to our knowledge only—exact algorithm to compute the distance is given by Touli and Wang [11], who leverage an equivalent definition in terms of a single map. Gasparovic *et al.* [5] show that the interleaving distance can be formulated in terms of a related metric [7] for labeled merge trees, and their results have inspired heuristics for computing the interleaving distance [4] and for comparing terrains [12, 14]. Pegoraro [8] propose another heuristic, based on a novel formulation in terms of couplings. Lastly, Beurskens *et al.* [2] establishe a relation to the Fréchet distance between one-dimensional curves, via so-called *ordered merge trees*.

2 Interleaving Distance between Merge Trees

Let T be a rooted tree, that is, a tree with one vertex identified as the *root*. We identify T with a *topological realization*: we represent each edge of T by a unit segment $[0, 1]$ and connect these segments according to the adjacencies of their corresponding edges in T. We distinguish between elements of the combinatorial tree, called *vertices*, and elements of the topological realization, called *points*. A *merge tree* is a pair (T, f), where T is a finite rooted tree, and f is a continuous *height function* on (the topological realization of) T that is strictly increasing towards the root and such that the root has height ∞ (see Figure 1). The function f naturally induces an ancestor-relation on the points of T.

Interleavings. The *interleaving distance* between two merge trees (T_1, f_1) and (T_2, f_2) is defined in terms of δ-compatible maps [6]: a continuous map from T_1 to T_2 that takes each point x of T_1 to a point of T_2 exactly δ higher, and a similar map from T_2 to T_1. We refer to this height difference δ as a *shift*. We use a non-standard definition where we allow maps that satisfy a weaker notion of continuity and use varying shifts; a similar definition is used in [1].

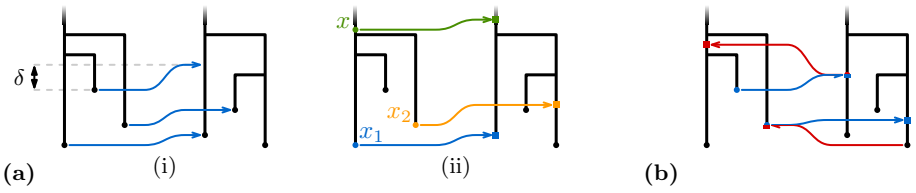


Figure 3 (a) An up-map (i) maps points to a higher point in the other tree, and (ii) preserves ancestors: the image of x is an ancestor of the images of both x_1 and x_2. **(b)** Composing both maps of an interleaving takes each point to an ancestor.

First, we refer to a pair of points (x, y), with x a point of T_1 and y a point of T_2, as an *arrow* if the height of x is less than the height of y. We define the *shift* of (x, y) as the height difference between x and y, denoted $\sigma(x, y) \coloneqq f_2(y) - f_1(x)$. We call a (not necessarily continuous) map α from T_1 to T_2 an *up-map* if (i) for each point x of T_1, the height of $\alpha(x)$ is greater than the height of x, and (ii) for all points x_1 and x_2 of T_1, if x_1 is an ancestor of x_2 then $\alpha(x_1)$ is also an ancestor of $\alpha(x_2)$ (see Figure 3a). By definition, each pair $(x, \alpha(x))$ is an arrow; we define the *shift* of α as the supremum shift over all arrows of α:

$$\sigma(\alpha) \coloneqq \sup\{\sigma(x, \alpha(x)) \mid x \in T_1\}. \tag{1}$$

We define arrows and up-maps from T_2 to T_1 symmetrically. An *interleaving* between T_1 and T_2 is then an up-map α from T_1 to T_2 and an up-map β from T_2 to T_1 such that for all points x of T_1, the point $\beta(\alpha(x))$ is an ancestor of x, and symmetrically for all points y of T_2, the point $\alpha(\beta(y))$ is an ancestor of y (see Figure 3b). We define the *shift* of the interleaving (α, β) as the maximum of the shift of α and the shift of β.

▶ **Definition 2.1.** For two merge trees (T_1, f_1) and (T_2, f_2), we define the distance $d(T_1, T_2)$ as the smallest δ for which an interleaving between T_1 and T_2 with shift δ exists.

We can show[1] that Definition 2.1 is well-defined, i.e. there always exists an *optimal interleaving*. Moreover, we can show[1] that $d(T_1, T_2)$ is equal to the original interleaving distance as defined in [6]. In the remainder of the paper, we hence refer to $d(T_1, T_2)$ as the *interleaving distance*.

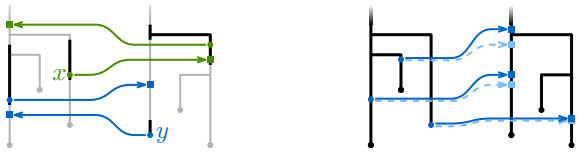
Partial interleavings. Let S_1 and S_2 be subsets of points of T_1 and T_2, respectively. A map φ from S_1 to T_2 is a *partial up-map* if properties (i) and (ii) hold for all points in S_1, and symmetrically for a map ψ from S_2 to T_1. A pair of partial up-maps (φ, ψ) is a *partial interleaving* if for all points x of S_1 and y of S_2, if y is an ancestor of $\varphi(x)$ then $\psi(y)$ is an ancestor of x, and if x is an ancestor of $\psi(y)$ then $\varphi(x)$ is an ancestor of y (see Figure 4a). We call S_1 and S_2 the *domains* of φ and ψ, denoted $\mathrm{dom}(\varphi)$ and $\mathrm{dom}(\psi)$, respectively. The shift of a partial up-map or partial interleaving is defined analogously to Eq. (1).

Lastly, we say a φ *extends* an arrow (x, y) if $\varphi(x)$ is an ancestor of y. Similarly, φ *extends* another partial up-map φ' if φ extends all arrows of φ', and (φ, ψ) *extends* another partial interleaving (φ', ψ') if φ extends φ' and ψ extends ψ' (see Figure 4b).

3 Residual Interleaving Distance

Fix a partial interleaving $\mathcal{P} = (\varphi, \psi)$, and assume it is *finite*: both $\mathrm{dom}(\varphi)$ and $\mathrm{dom}(\psi)$ contain only finitely many points. We refer to a (partial) interleaving that extends $\mathcal{P}$ as a

[1] Theorem 9 and Lemma 7 in the full version, respectively.



(a) (b)

Figure 4 (a) A partial interleaving: any ancestor of $\varphi(x)$ in the domain of ψ must map to an ancestor of x (and vice versa ancestors of $\psi(y)$.) **(b)** An extension (solid arrows) of a partial up-map (dashed arrows) takes each point to an ancestor of the original image.

(*partial*) $\mathcal{P}$*-extension*. There always exists a trivial $\mathcal{P}$-extension by mapping each point to the root at ∞ of the other tree. Our goal is to find a "tightest possible" $\mathcal{P}$-extension, that is, an interleaving that extends $\mathcal{P}$ and minimizes the shift of the remaining arrows. Consider the example in Figure 5. Intuitively, the map α_2 describes a "tighter" extension of φ than the map α_1. However, if we only disregard the shift of the arrows of φ, the shift of the remaining arrows of α_1 is equal to the (supremum) shift of the remaining arrows of α_2. The arrow (x, y) induces a "fan" of arrows that all valid up-maps need to extend. Specifically, any ancestor of x needs to be mapped to an ancestor of y. To distinguish between α_1 and α_2, we therefore not only disregard the shift of (x, y), but of any arrow that is "contained" within such a fan.

Formally, we define the *fan* of an arrow (x, y) as the set of arrows (x', y) for all ancestors x' of x with height at most that of y, written,

$$F[(x, y)] \coloneqq \{(x', y) \mid x' \in T_1 \text{ such that } x' \text{ is an ancestor of } x \text{ with } f_1(x') \leq f_2(y)\}.$$

The *fan* of a partial up-map φ, denoted $F[\varphi]$, is the union of all fans $F[(x, \varphi(x))]$.

For an arrow a from T_1 to T_2, the φ-residual shift of a, denoted $\sigma_\varphi(a)$, is 0 if a lies within $F[\varphi]$, and $\sigma(a)$ otherwise. Similar to Eq. 1, the φ*-(residual) shift* of an up-map φ', denoted $\sigma_\varphi(\varphi')$, is the supremum φ-shift over all arrows of φ':

$$\sigma_\varphi(x, y) \coloneqq \begin{cases} 0 & \text{if } (x, y) \in F[\varphi], \\ \sigma(x, y) & \text{else,} \end{cases} \qquad \sigma_\varphi(\varphi') \coloneqq \sup\{\sigma_\varphi(x, \varphi'(x)) \mid x \in \mathrm{dom}(\varphi')\}.$$

In Figure 5, the φ-shift of α_2 is δ', whereas the φ-shift of α_1 is δ.

The $\mathcal{P}$*-(residual) shift* of a partial interleaving (φ', ψ') is defined as the maximum of the φ-residual shift of φ' and the ψ-residual shift of ψ'.

▶ **Definition 3.1.** The $\mathcal{P}$-residual interleaving distance $d_{\mathcal{P}}(T_1, T_2)$ between T_1 and T_2 is the infimum δ for which there exists a $\mathcal{P}$-extension whose $\mathcal{P}$-residual shift is δ.

A $\mathcal{P}$-extension is *optimal* if it has shift equal to $d_{\mathcal{P}}$. An optimal extension always exists.

▶ **Theorem 3.2** (Theorem 15 in the full version)**.** *Any finite partial interleaving $\mathcal{P}$ between any two merge trees T_1 and T_2 admits an optimal $\mathcal{P}$-extension.*

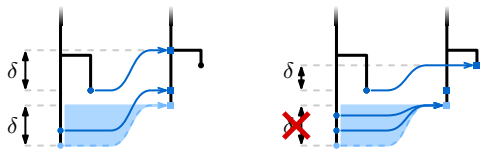


Figure 5 For the residual shift of a partial up-map, the shift of arrows within fans (the filled area) is disregarded, so the residual shift of the right map is strictly less the residual shift of the left map.

4 Locally Correct Interleavings

Building upon the residual interleaving distance, we define a notion of local optimality for interleavings. Let $\mathcal{P} = (\varphi, \psi)$ be a partial interleaving. We define the *restriction* of φ to a set of points S_1 in the domain of φ as the map $\varphi|_{S_1}$ given by $\varphi|_{S_1}(x) \coloneqq \varphi(x)$ for all x in S_1. This restriction is again a partial up-map, and φ trivially extends $\varphi|_{S_1}$. Moreover, for any sets of points S_1 and S_2 in the domains of respectively φ and ψ, the pair $\mathcal{R} = (\varphi|_{S_1}, \psi|_{S_2})$ is a partial interleaving such that $\mathcal{P}$ extends $\mathcal{R}$. We call $\mathcal{R}$ a *restriction* of $\mathcal{P}$. An interleaving $\mathcal{I}$ is *locally correct* if for all restrictions $\mathcal{R}$ of $\mathcal{I}$, the $\mathcal{R}$-residual shift of $\mathcal{I}$ is equal to the $\mathcal{R}$-residual interleaving distance. Intuitively, if we fix any number of arrows of I, the remaining arrows are as good as they can be given the fixed part.

▶ **Definition 4.1.** An interleaving $\mathcal{I} = (\alpha, \beta)$ is *locally correct* (see Figure 6) if and only if all restrictions $\mathcal{R} = (\alpha|_{S_1}, \beta|_{S_2})$ satisfy:

$$d_{\mathcal{R}}(T_1, T_2) = \sigma_{\mathcal{R}}(\mathcal{I}). \tag{2}$$

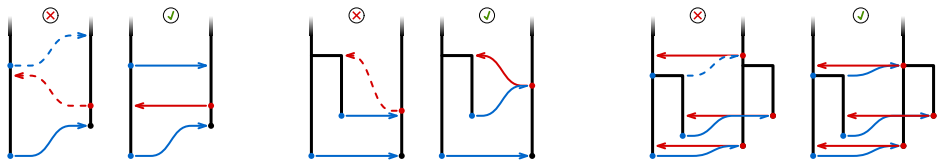


Figure 6 Three pairs of merge trees. We show locally-correct (✓) and non-locally correct interleavings (⊗). The dashed arrows in the ⊗ cases form restrictions that violate Eq. (2).

Taking $S_1 = \emptyset = S_2$, we see that a locally correct interleaving $\mathcal{I}$ satisfies $d(T_1, T_2) = \sigma(\mathcal{I})$. In other words, $\mathcal{I}$ is an optimal interleaving. The reverse is not necessarily true; not every optimal interleaving is locally correct (recall Figure 2). That raises the question: does a locally correct interleaving always exist? In Theorem 4.3, we prove that it does. To do so, we incrementally build a partial interleaving $\mathcal{P}$ by augmenting it with "bottleneck" arrows. We show that with every augmentation, the $\mathcal{P}$-residual interleaving distance strictly decreases, until, after a finite number of iterations, it becomes zero. For the resulting $\mathcal{P}$, we then show that any optimal $\mathcal{P}$-extension is locally correct.

Bottlenecks. Locally, we might improve an (optimal) interleaving by "pushing down" some of its arrows. However, we cannot do this for all arrows; otherwise, we would obtain an interleaving whose shift is strictly smaller than the interleaving distance. We hence want to identify a minimal set of arrows that we cannot "push down". We say a $\mathcal{P}$-extension $\mathcal{Q}$ is a *$\mathcal{P}$-augmentation* if (i) the $\mathcal{P}$-residual shift of $\mathcal{Q}$ is at most $d_{\mathcal{P}}$, that is, $\sigma_{\mathcal{P}}(\mathcal{Q}) \leq d_{\mathcal{P}}$, and (ii) the $\mathcal{Q}$-residual interleaving distance is strictly less than $d_{\mathcal{P}}$, that is, $d_{\mathcal{Q}} < d_{\mathcal{P}}$. We further say that $\mathcal{Q}$ is *minimal* if it does not extend any other $\mathcal{P}$-augmentation (see Figure 7).

▶ **Lemma 4.2** (Lemma 20 in the full version). *Any finite interleaving $\mathcal{P}$ with $d_{\mathcal{P}} > 0$ admits a finite minimal $\mathcal{P}$-augmentation.*

▶ **Theorem 4.3** (Theorem 19 in the full version). *Any two merge trees T_1 and T_2 admit a locally correct interleaving.*

Proof sketch. Let $\mathcal{P}$ be the empty interleaving. By Lemma 4.2, we can iteratively replace $\mathcal{P}$ with a finite minimal $\mathcal{P}$-augmentation. After each iteration, we can show that $\mathcal{P}$ satisfies

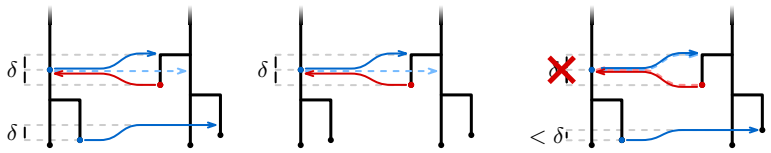


Figure 7 A partial interleaving (dashed arrow) and two augmentations (left and middle). The left augmentation is non-minimal, as the middle augmentation is strictly smaller. The right figure shows that if we take the middle augmentation, the resulting residual shift becomes strictly smaller than δ.

two properties. First, we can show the domains of φ and ψ have grown and now include an additional vertex of T_1 or T_2. Specifically, this means that after at most n iterations, where n is the total number of vertices of T_1 and T_2, there are no more vertices left "unspecified". We can use this to show that for the corresponding partial interleaving $\mathcal{P}$, the $\mathcal{P}$-residual interleaving distance must be zero. Secondly, we can show that $\mathcal{P}$ is *partial locally correct*, that is, for all restriction of $\mathcal{P}$, it holds that $d_{\mathcal{R}}(T_1, T_2) \geq \sigma_{\mathcal{R}}(\mathcal{P})$. To do so, we prove that the $\mathcal{P}$-residual interleaving distance attains a value in a (finite) set of $\mathcal{P}$-*critical values*. Each critical value corresponds to a *critical pair* of points of T_1 and T_2. Each arrow of a minimal augmentation then corresponds to such a critical pair, which gives us more information about the resulting partial interleaving. In particular, we use this to argue that the $\mathcal{R}$-residual interleaving distance is at least the shift of any arrow of $\mathcal{P}$ that is not used by $\mathcal{R}$. This, in turn, is exactly the $\mathcal{R}$-shift of $\mathcal{P}$; it follows that $\mathcal{P}$ is partial locally correct. Once the $\mathcal{P}$-residual interleaving distance becomes zero, we stop iterating. We then conclude the proof by arguing that any $\mathcal{P}$-extension is locally correct. ◀

Computing a locally correct interleaving. We sketch how the above construction can be translated into an incremental algorithm. To compute the residual interleaving distance, we modify the dynamic program by Touli and Wang [11] to also incorporate constraints induced by partial interleavings. We use the output to extract some augmentation. We then iteratively reduce the set of critical pairs by repeatedly calling the modified dynamic program until we obtain a (finite) minimal augmentation. This minimal augmentation, in turn, reduces the residual interleaving distance; we iterate until it has dropped to zero. To obtain a locally correct interleaving, we select suitable entries from the table filled by the modified dynamic program. A naive implementation of this algorithm takes n iterations, each of which needs to compute the residual interleaving distance $O(n^2)$ times to obtain a minimal augmentation (since an augmentation contains $O(n^2)$ critical pairs). So, the total running time of our algorithm is $O(n^3 T(n))$, where $T(n)$ is the running time of computing the residual interleaving distance. The algorithm by Touli and Wang [11] takes $O(n^4 2^\tau \tau^{\tau+2} \log n)$ time, where τ is a parameter related to the maximum degrees in the input trees.

Acknowledgments. Thijs Beurskens and Tim Ophelders are supported by the Dutch Research Council (NWO) under project numbers OCENW.M20.089 and VI.Veni.212.260, respectively.

References

1 P.K. Agarwal, K. Fox, A. Nath, A. Sidiropoulos, and Y. Wang. Computing the Gromov-Hausdorff distance for metric trees. *ACM Transactions on Algorithms*, 14(2):1–20, 2018. doi:10.1145/3185466.

2 T. Beurskens, T. Ophelders, B. Speckmann, and K. Verbeek. Relating interleaving and Fréchet distances via ordered merge trees. In *Proc. ACM-SIAM Symposium on Discrete Algorithms (SODA25)*, pages 5027–5050. Society for Industrial and Applied Mathematics, 2025. doi:10.1137/1.9781611978322.170.

3 K. Buchin, M. Buchin, W. Meulemans, and B. Speckmann. Locally correct Fréchet-matchings. *Computational Geometry*, 76(1):1–18, 2019. doi:10.1016/j.comgeo.2018.09.002.

4 J. Curry, H. Hang, W. Mio, T. Needham, and O. B. Okutan. Decorated merge trees for persistent topology. *Journal of Applied and Computational Topology*, 6(3):371–428, February 2022. doi:10.1007/s41468-022-00089-3.

5 E. Gasparovic, E. Munch, S. Oudot, K. Turner, B. Wang, and Y. Wang. Intrinsic interleaving distance for merge trees. *La Matematica*, 4(1):40–65, 2025. doi:10.1007/s44007-024-00143-9.

6 D. Morozov, K. Beketayev, and G. Weber. Interleaving distance between merge trees. Manuscript (accessed on 06-03-2025), 2013. URL: https://mrzv.org/publications/interleaving-distance-merge-trees/manuscript/.

7 E. Munch and A. Stefanou. The ℓ^∞-cophenetic metric for phylogenetic trees as an interleaving distance. In *Research in Data Science*, volume 17 of *Association for Women in Mathematics Series*, pages 109–127. Springer International Publishing, 2019. doi:10.1007/978-3-030-11566-1_5.

8 M. Pegoraro. A graph-matching formulation of the interleaving distance between merge trees. *AIMS Mathematics*, 10(6):13025–13081, 2025. doi:10.3934/math.2025586.

9 M. Pont, J. Vidal, J. Delon, and J. Tierny. Wasserstein distances, geodesics and barycenters of merge trees. *IEEE Transactions on Visualization and Computer Graphics*, 28(1):291–301, January 2022. doi:10.1109/TVCG.2021.3114839.

10 R. Sridharamurthy, T. B. Masood, A. Kamakshidasan, and V. Natarajan. Edit distance between merge trees. *IEEE Transactions on Visualization and Computer Graphics*, 26(3):1518–1531, 2020. doi:10.1109/TVCG.2018.2873612.

11 E.F. Touli and Y. Wang. FPT-algorithms for computing the Gromov-Hausdorff and interleaving distances between trees. *Journal of Computational Geometry*, 13:89–124, 2022. doi:10.20382/jocg.v13i1a4.

12 L. Yan, T. B. Masood, F. Rasheed, I. Hotz, and B. Wang. Geometry aware merge tree comparisons for time-varying data with interleaving distances. *IEEE Transactions on Visualization and Computer Graphics*, 29(8):3489–3506, 2022. doi:10.1109/TVCG.2022.3163349.

13 L. Yan, T. B. Masood, R. Sridharamurthy, F. Rasheed, V. Natarajan, I. Hotz, and B. Wang. Scalar field comparison with topological descriptors: Properties and applications for scientific visualization. *Computer Graphics Forum*, 40(3):599–633, 2021. doi:10.1111/cgf.14331.

14 L. Yan, Y. Wang, E. Munch, E. Gasparovich, and B. Wang. A structural average of labeled merge trees for uncertainty visualisation. *IEEE Transactions on Visualization and Computer Graphics*, 26(1):832–842, 2020. doi:10.1109/TVCG.2019.2934242.

Multi-Block Grids via Polycubes

Maxim Snoep[1], Stevie-Ray Janssen[2], Bettina Speckmann[1], and Kevin Verbeek[1]

1 **TU Eindhoven, The Netherlands**
[m.snoep | b.speckmann | k.a.b.verbeek]@.tue.nl
2 **Royal Netherlands Aerospace Centre, The Netherlands**
stevie.ray.janssen@nlr.nl

Abstract

Multi-block grids are widely used for high-fidelity computational fluid dynamics (CFD) simulations in aerospace. However, constructing high-quality multi-block grids for complex geometries is still a largely manual, time-intensive process which relies on human expertise. In this abstract we report on our efforts to automate a crucial step in the multi-block grid pipeline, namely the construction of a so-called polycube segmentation of the input geometry.

1 Introduction

Multi-block grids remain central in computational fluid dynamics (CFD) for applications that require high near-wall resolution and low numerical dissipation, such as external aerodynamics, turbomachinery, and aeroacoustics. The regular connectivity of multi-block grids allows for strongly anisotropic and direction-aligned spacing. It supports high-performance implementations due to stencil locality and direct addressing of data.

Multi-block grids have clear structural advantages over unstructured grids, however, these advantages usually come at high labor costs: up to now, there are no automatic pipelines to create multi-block grids from CAD models and maintain them through a processing pipeline. Specific challenges include valid non-intersecting block extrusion and acceptable cell quality near junctions, concave regions, and block interfaces, all of which tend to require manual intervention. Each manual step in a processing pipeline not only slows down turnaround time and but also introduces significant user-to-user variability.

In this abstract, we describe a method that fully automates a significant portion of the multi-block grid generation pipeline (see Figure 1). In particular, we show how DualCube [4], a recent automatic polycube segmentation method, can replace the manual process of constructing polycube segmentations in NLR's multi-block grid generation pipeline. Our proof-of-concept implementation creates multi-block grids that are well-suited for CFD applications:

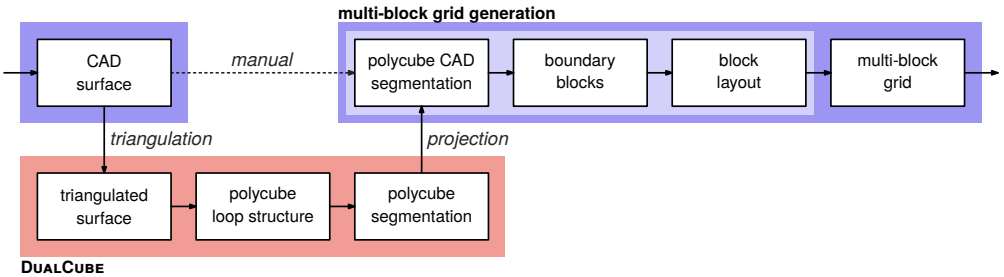


Figure 1 The integration of polycube segmentation via DualCube [4] into the generation of multi-block grids by the NLR.

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

they are boundary-conforming, have positive Jacobians, and controlled wall-normal growth. The automated construction of (guaranteed to be) valid polycube segmentations with DualCube has the potential to reduce the mesh-to-first-solution turnaround to approximately one day for aerospace configurations.

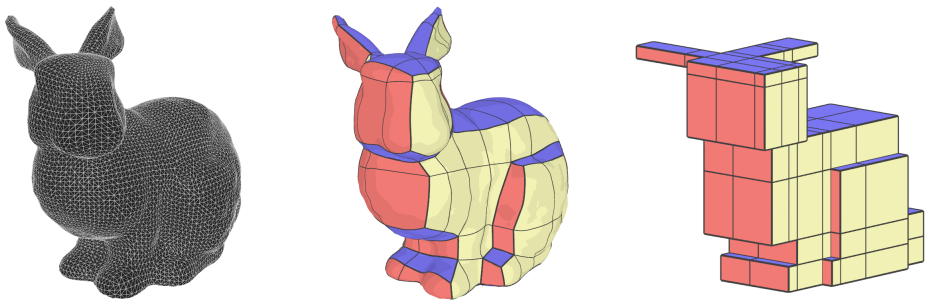
The improved pipeline is the result of a collaboration between the Royal Netherlands Aerospace Centre (NLR) and researchers at TU Eindhoven, who developed DualCube. Section 2 introduces polycube segmentations and describes how DualCube constructs them. Section 3 outlines the NLR multi-block grid generation pipeline and explains how DualCube is integrated into it. Finally, Section 4 showcases results from our proof-of-concept integration.

2 Polycubes

A polycube is an orthogonal polyhedron with axis-aligned (quadrilateral) faces (see Figure 2c). To obtain a polycube segmentation of a given closed 2-dimensional input surface $\mathcal{M}$, we must compute a continuous map f from a polycube Q to the given surface $\mathcal{M}$, where Q and $\mathcal{M}$ must necessarily have the same genus. We refer to such a map f as a *polycube map*. The edges of Q mapped onto $\mathcal{M}$ define a segmentation of $\mathcal{M}$ into *patches*, where each patch corresponds to a (quadrilateral) face of Q. This segmentation is called a *polycube segmentation* (see Figure 2b). Each patch has a corresponding *label* $\{+X, -X, +Y, -Y, +Z, -Z\}$, indicating the direction of the normal vector of the corresponding face in Q.

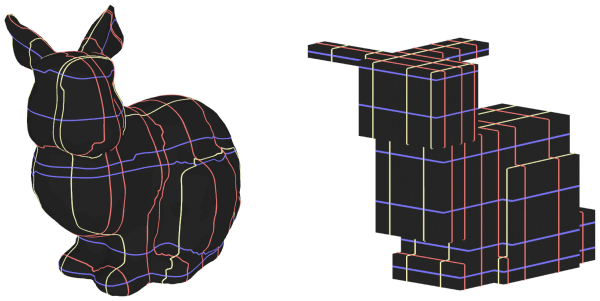
Polycube maps have been studied extensively as a tool for solving various problems, such as texture mapping [7] and hexahedral meshing [1]. In recent years, several different polycube segmentation algorithms have been proposed. These methods are based on deformation [1], voxelization [9], mixed-integer optimization [2], or deep learning [10]. However, these methods do not always yield valid polycube segmentations and do not easily support human interaction [4].

The polycube segmentation of a given input surface is not unique. The choice of segmentation should be informed by the downstream task. Generally, the quality of a polycube segmentation is a tradeoff between *topological simplicity* and *low parametric distortion*. In practice, these characteristics correspond to a polycube segmentation with a small number of patches with geodesic boundaries, orthogonal corners, and locally homogeneous surface normals.



(a) **(b)** **(c)**

Figure 2 The Stanford bunny (a), a polycube segmentation (b) and corresponding polycube (c).



(a) (b)

Figure 3 A polycube loop structure on the Stanford bunny (a) and corresponding polycube (b).

2.1 Polycubes via Dual Loops

Our method DualCube [4] computes guaranteed valid polycube segmentations. In our method, a polycube segmentation is defined implicitly by constructing a *loop structure* (see Figure 3a): a set of labeled and oriented loops on the input surface. These loops form the dual representation of the polycube segmentation. A corresponding polycube segmentation can then be constructed from this loop structure via a *primalization* step in which segmentation corners are placed in loop regions and adjacent corners connected via non-intersecting paths. For more details see [4]. The validity of the segmentation follows from the validity of the loop structure, which is straightforward to check and maintain [3]. Compared to other existing polycube segmentation algorithms, DualCube offers several advantages:

Guaranteed validity. DualCube is based on a complete characterization of polycubes and, as such, guarantees validity.

Incremental refinement. The loop structure supports coarse-to-fine design; loops can be both added and deleted to introduce or remove detail. This incremental approach directly supports the general quality tradeoff between simplicity and distortion.

Human-in-the-loop. DualCube supports interactive addition and deletion of loops and repositioning of polycube segmentation corners via an intuitive user interface.

3 Multi-Block Grid Generation

A multi-block grid for CFD decomposes the flow domain into hexahedral blocks with node-matching interfaces, each containing a logically structured hexahedral grid. The high-level pipeline to generate such a grid follows three steps: (1) preparing the CAD input surface, (2) generating the block layout, and (3) per-block grid generation (edge/face constraints and elliptic interior smoothing). We follow Spekreijse and Boerstoel [5] for the second and third steps, but in this abstract we focus only on the second.

The second step, creating a viable block layout for complex input surfaces, is a challenging and time-intensive step. Spekreijse and Boerstoel [6] defined the structure and requirements of a block layout. To construct a block layout according to their specifications, we follow three steps: (2.a) create a polycube segmentation of the input surface, (2.b) extrude each surface patch of the segmentation into a boundary block, and (2.c) complete the block layout by subdividing the surrounding space according to the polycube segmentation.

As mentioned before, a polycube segmentation is a set of quadrilateral surface patches whose patch adjacency and edge pairing admit a one-to-one correspondence with the faces of

a polycube. This structure allows the extrusion into boundary blocks and the final completion of a valid block layout. The polycube segmentation hence carries the full geometric and topological information needed to create the multi-block grid.

When used in the context of multi-block grid generation for CFD with aerospace models, additional requirements arise beyond the general trade-off between topological simplicity and low parametric distortion. First of all, engineered models usually contain many sharp edges; ideally, the boundaries of the polycube segmentation align with these edges. Furthermore, aerospace geometries may have features that are extremely thin in one of the dimensions, such as the trailing edge of a wing or tail. Another challenge are ambiguously oriented features that do not align naturally with the global Cartesian axes, such as swept wings or v-tails. Capturing such diagonal features using staircases of cubes can lead to degenerate or even self-intersecting blocks in the multi-block grids. Hence one needs a representation with few axis-aligned patches and suitable transformations. Finally, aerospace models frequently exhibit features on multiple geometric scales. Large-scale components such as the fuselage and wings should be captured just as well as smaller details, for example struts or vanes. Furthermore, when the input geometry exhibits symmetry, it is generally desirable for the polycube segmentation to preserve this symmetry, as to avoid introducing artificial asymmetry in the multi-block grid and, consequently, in the CFD.

Previously, constructing such polycube segmentations required substantial manual effort. The DualCube algorithm automates this step and thereby significantly accelerates the pipeline. During the automatic loop placement and the subsequent primalization step, DualCube attempts to satisfy the requirements outlined above. When the automatically constructed polyucube segmentation does not yield a desirable result, the user can refine it interactively. In particular, loops can be added or removed to increase or decrease the complexity of the segmentation, and polycube corners can be repositioned to better align the segmentation with geometric features.

For computational efficiency reasons, DualCube operates on triangulated surfaces. Hence we add two steps to the pipeline that first convert the input CAD surface into a triangulated surface and then project the results of DualCube back onto the original CAD geometry.

4 Results

We tested our proof-of-concept pipeline on the NASA Common Research Model (CRM) [8] in the WBNPH configuration (wing–body with nacelle/pylon installation and horizontal tailplane), which is a representative transport-aircraft geometry used widely in CFD studies. The WBNPH configuration is challenging for structured meshing because the nacelle–pylon–wing and tail–body integrations feature multiple junctions, tight concave intersections, and slender passages that strongly constrain how blocks can be arranged and connected.

We showcase our results in Figure 4. The polycube has one-to-one correspondence with the polycube segmentation. This in turn is used to construct the block layout, which enables per-block prescription of spacing and stretching, resulting in a high-quality multi-block grid.

The resulting mesh of about 22M cells follows the geometry cleanly, tracking the sharp edges, fillets, and junctions. Cell spacing transitions smoothly from the near-wall region to the farfield, with controlled growth and no abrupt changes in size. Near-wall grid lines remain largely orthogonal to the surface, supporting accurate resolution of flow gradients. Overall, the mesh meets the established quality criteria for high-fidelity CFD simulations and was accepted by NLR's in-house CFD solver suite.

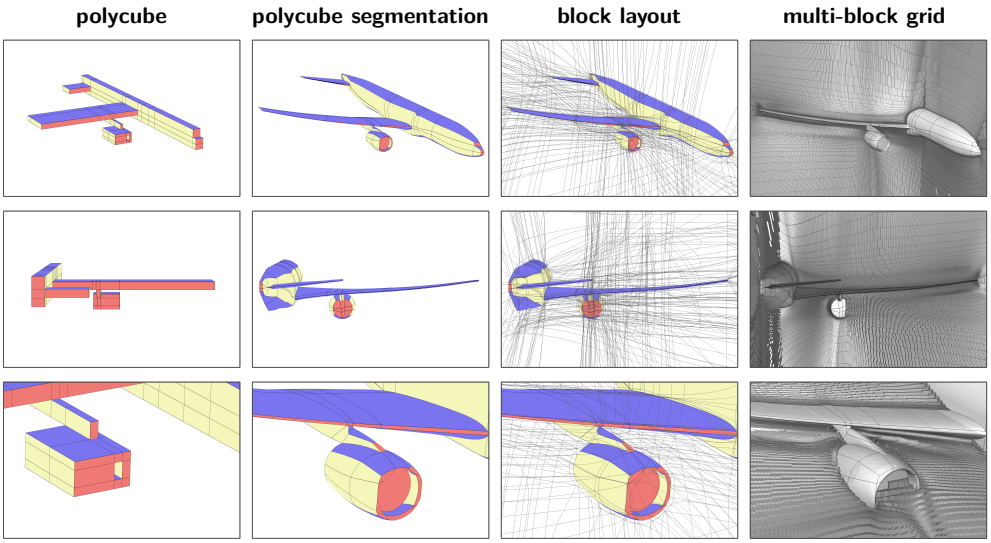


Figure 4 Our pipeline applied to the NASA CRM configuration [8].

References

1 James Gregson, Alla Sheffer, and Eugene Zhang. All-hex mesh generation via volumetric polycube deformation. *Computer Graphics Forum*, 30(5), 2011. doi:10.1111/j.1467-8659.2011.02015.x.

2 Manish Mandad, Ruizhi Chen, David Bommes, and Marcel Campen. Intrinsic mixed-integer polycubes for hexahedral meshing. *Computer Aided Geometric Design*, 94, 2022. doi:10.1016/j.cagd.2022.102078.

3 Maxim Snoep, Bettina Speckmann, and Kevin Verbeek. Polycubes via dual loops. In *Proceedings of the 2025 SIAM International Meshing Roundtable (IMR)*, 2025. doi:10.1137/1.9781611978575.7.

4 Maxim Snoep, Bettina Speckmann, and Kevin Verbeek. Robust construction for polycube segmentations via dual loops. *Computer Graphics Forum*, 44(5), 2025. doi:10.1111/cgf.70195.

5 Stephanus P. Spekreijse and Jan W. Boerstoel. Multiblock grid generation. Part I: Elliptic grid generation methods for structured grids. Technical report, National Aerospace Laboratory (NLR), 1996. URL: http://hdl.handle.net/10921/1355.

6 Stephanus P. Spekreijse and Jan W. Boerstoel. Multiblock grid generation. Part II: Multiblock aspects. Technical report, National Aerospace Laboratory (NLR), 1996. URL: http://hdl.handle.net/10921/1356.

7 Marco Tarini, Kai Hormann, Paolo Cignoni, and Claudio Montani. PolyCube-Maps. *ACM Transactions on Graphics*, 23(3), 2004. doi:10.1145/1015706.1015810.

8 John C. Vassberg, Anthony J. Sclafani, and Mark A. DeHaan. Development of a common research model for applied CFD validation studies. In *Proceedings of the 26th AIAA Applied Aerodynamics Conference*, 2008. doi:10.2514/6.2008-6919.

9 Yang Yang, Xiaoming Fu, and Ligang Liu. Computing surface polycube-maps by constrained voxelization. *Computer Graphics Forum*, 38(7), 2019. doi:10.1111/cgf.13838.

10 Yuxuan Yu, Yuzhuo Fang, Hua Tong, and Yongjie J. Zhang. DL-Polycube: Deep learning enhanced generalized polycube method for high-quality hexahedral mesh generation and volumetric spline construction. *Engineering with Computers*, 41(3), 2025. doi:10.1007/s00366-025-02161-9.

Counting d-Dimensional Polycubes, Revisited

Gill Barequet[1] **and Tom Feldman**[1]

1 **Dept. of Computer Science,**
Technion—Israel Inst. of Technology,
Haifa 3200003, Israel
barequet@cs.technion.ac.il
tom.p@campus.technion.ac.il

Abstract

Polycubes are connected structures formed by unit cubes in three or more dimensions. No closed-form formula for the number of polycubes of size n is currently known, and counting them is a fundamental computational challenge in combinatorial geometry. We explain the combinatorial principles underlying the correctness of the Redelmeier-Dodds's method for counting 3-dimensional polycubes, generalize it to higher dimensions, and extend it further by optimizing its recursive process. We report our new counts of polycubes in 3 to 5 dimensions.

1 Introduction

A *polyomino* is a connected set of cells on the two-dimensional square lattice, where connectivity is through edges only. The symbol $A(n)$ denotes the number of "fixed" polyominoes[1] of size n (or up to size n), where size is simply the number of squares a polyomino contains. Similarly, in a fixed dimension d, the symbol $A_d(n)$ denotes the number of fixed polycubes of volume n, that is, fixed polycubes consisting of exactly n cubes. The primary objective of this paper is to compute $A_d(n)$ in some dimensions d for the largest possible values of n.

2 Redelmeier's Algorithm

Redelmeier's algorithm [9] is the earliest and most well-known nontrivial algorithm for counting polyominoes.[2] Aleksandrowicz and Barequet [1] later showed that Redelmeier's algorithm is not limited to two dimensions but can also be generalized to higher dimensions. The algorithm is shown in Figure 1.

There are a few important issues to note:

1. A total lexicographic order is defined on the cells of the lattice. One cell is marked as the "origin," and only cells that are lexicographically greater than the origin are considered.
2. The "new neighbors" mentioned in Step 4a are neighbors of u that are **not** neighbors of any cell already contained by the polyomino.
3. The order in which cells in the untried set U are processed is not significant.
4. Once a cell u is removed from U, it cannot re-enter U *in the same recursion branch*, since it cannot become a new neighbor again. After U is fully exhausted and the recursion unfolds to the previous level, cell u may become untried again if it becomes a new neighbor of some other expanded cell.

[1] Fixed polyominoes are distinct up to translation only, while "free polyominoes" are distinct up to translation, rotation, and mirroring.

[2] Currently, the most-efficient known algorithm for counting polyominoes is the transfer-matrix method of Jensen [5], however, generalizing it to $d \geq 3$ dimensions is considered impractical.

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

Initialize the parent to be the empty polyomino, and the untried set to contain only the origin. The following steps are repeated until the untried set is exhausted.

1. Remove an arbitrary element from the untried set.
2. Place a cell at this point [element].
3. Count this new polyomino.
4. If the size is less than n:
   a. Add new neighbors to the untried set.
   b. Call this algorithm recursively with the new parent being the current polyomino, and the new untried set being a copy of the current one.
   c. Remove the new neighbors from the untried set.
5. Remove newest cell.

Figure 1 Redelmeier's algorithm [9, p. 196].

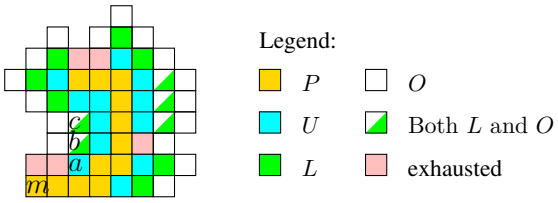


Figure 2 A sample snapshot of the algorithm (m is the origin).

3 Dodds's Version of the Algorithm

Dodds's counts of 3D polycubes up to size 20 were published in The On-Line Encyclopedia of Integer Sequences [10, Seqs. A000162 and A001931]. His software, which contained many programming tricks, was never accompanied by a paper publication or a technical description. The same approach was subsequently employed by Thompson for computing the values of $A_3(21)$ [7] and $A_3(22)$ [8], using more powerful computational hardware.

Dodds optimized Redelmeier's algorithm by replacing the overhead of the calls in the last levels of the recursion by local combinatorial calculations. Let us first examine this idea at the last level of the recursion, where it reaches depth $n-1$, *i.e.*, when only one cube remains to be added to complete a polycube of size n. Since the final cube must be taken from the current version of U, further recursive calls can be avoided by simply returning $|U|$. Dodds's version of the algorithm builds on this idea. The recursion is halted at depth $n-3$, and then all the valid extensions are counted by using combinatorial arguments.

Let P denote the current polycube. For every cube $u \in U$, let $L(u)$ denote the set of neighbors of u that would become *new neighbors* if u were added to the polycube. Define $\hat{\ell}_1(u) = |L(u)|$ and let $L = \bigcup_{u \in U} L(u)$. For a cube $\ell \in L$, let $O(\ell)$ be the set of neighbors of ℓ that are free and not in U, and let $\hat{o}(\ell) = |O(\ell)|$. Define $O = \bigcup_{\ell \in L} O(\ell)$. Finally, for every $\ell \in L$, let $\hat{u}(\ell)$ denote the number of its neighbors that belong to the untried set U. Figure 2 illustrates these notations. In this example, we have that $\hat{u}(b) = 2$, $\hat{o}(b) = 2$, and $\hat{\ell}_1(a) = 1$. Note that a cube may simultaneously belong to both L and O if it is adjacent to cubes in both U and L. For instance, in this figure, the cube b belongs to L as it is a new neighbor of a, and also to O since it is a free neighbor of $c \in L$ that is not in U.

At depth $n-3$ of the recursion, we have to determine how many valid configurations of three additional cubes can be formed, given the current sets U and L. There are three principal cases to consider, which we review in order of complexity.

Case 1: All three cubes are from U

This is the simplest case. Since all three cubes are selected from U, the number of configurations is $\binom{|U|}{3}$, the number of 3-element subsets of U.

Case 2: Exactly one cube from U

In this case, the sought number of configurations is computed in a brute-force manner: For each cube $u \in U$, assume that u is added to the current polycube, and enumerate all possible two-cube extensions of u using cubes that are not in U. That is, for each cube $\ell \in L(u)$, consider every cube $v \in O(\ell)$, and count all distinct pairs (ℓ, v) that can extend u by two cubes. This computation can be performed by inspecting only the "local" environment of u, observing its immediate neighbors and their immediate neighbors. Obviously, only cubes that are free, that is, not already part of the current polycube, are considered.

Case 3: Exactly two cubes from U

In this case, two cubes come from U and the third one from L. We analyze this case by iterating over all cubes $\ell \in L$ and counting the combinations of two cubes in U that can form (together with ℓ) a connected extension of the current polycube.

There are two kinds of valid extensions that contain ℓ:

- The two other cubes are neighbors of ℓ in U. The number of these extensions is $\binom{\hat{u}(\ell)}{2}$.
- One cube is a neighbor of ℓ in U, and the other one is a cube in U that is not adjacent to ℓ. The number of these extensions is $\hat{u}(\ell) \cdot (|U| - \hat{u}(\ell))$.

Summing over all $\ell \in L$, we see that the total number of valid extensions in this case is

$$\sum_{\ell \in L} \left(\binom{\hat{u}(\ell)}{2} + \hat{u}(\ell)(|U| - \hat{u}(\ell)) \right).$$

4 Optimizing the Algorithm Further

In Dodds's version of the algorithm, each time the recursion reaches depth $n-3$, a full traversal of the sets U, L, and O is performed in order to compute the number of possible 3-cube extensions of the current polycube. This computation is repeated every time the fourth-to-last cube is added. We can exploit the fact that two polycubes in the same computation branch and at the same level differ by one cube (the cube that is replaced by another one) and save a significant portion of the computational effort.

We first describe the combinatorial counting mechanism employed by the algorithm when the recursion is halted at depth $n-3$. The core of Dodds's optimization lies in counting the ways to add the last three cubes (when the recursion reaches depth $n-3$) using the sets U, L, and O. We analyze the second case, in which exactly one cube is taken from U. Given a cube $u \in U$, there are two possible ways to extend the polycube by adding u and two other cubes:

1. Both remaining cubes are neighbors of u that belong to L. There are $\binom{\hat{\ell}(u)}{2}$ such options.
2. Exactly one of the two remaining cubes, say, ℓ, is a neighbor of u that belongs to L. The other cube must then be a neighbor of ℓ that belongs to O. (There are $\hat{o}(\ell)$ such neighbors.) The number of extensions in this subcase is, thus, $\sum_{\ell \in L(u)} \hat{o}(\ell)$.

Summing over all $u \in U$, we see that the total number of extensions from Case 2 is

$$\sum_{u\in U}\left(\binom{\hat{\ell}(u)}{2}+\sum_{\ell\in L(u)}\hat{o}(\ell)\right)=\sum_{u\in U}\binom{\hat{\ell}(u)}{2}+\sum_{u\in U}\sum_{\ell\in L(u)}\hat{o}(\ell)\underset{(*)}{=}\sum_{u\in U}\binom{\hat{\ell}(u)}{2}+\sum_{\ell\in L}\hat{u}(\ell)\hat{o}(\ell).$$

The equality marked with $(*)$ follows from a rearrangement of the terms in the summation. Indeed, in the term $\sum_{u\in U}\sum_{\ell\in L(u)}\hat{o}(\ell)$, each cube $\ell \in L$ contributes $\hat{o}(\ell)$ to the sum for each $u \in U$ that is a neighbor of ℓ. The number of such cubes u, for a given cube ℓ, is precisely $\hat{u}(\ell)$. Thus, the inner summation over $L(u)$ can be transformed into a sum over L, where each ℓ is weighted by its number of neighbors in U, *i.e.*, $\hat{u}(\ell)\hat{o}(\ell)$.

Now, we add the contribution from Case 3 to the contribution from Case 2, and obtain

$$\sum_{u\in U}\underbrace{\binom{\hat{\ell}(u)}{2}}_{\text{case 2.1}}+\sum_{\ell\in L}\left(\underbrace{\binom{\hat{u}(\ell)}{2}+\hat{u}(\ell)(|U|-\hat{u}(\ell))}_{\text{case 3}}+\underbrace{\hat{u}(\ell)\hat{o}(\ell)}_{\text{case 2.2}}\right),$$

which can be simplified to

$$=\sum_{u\in U}\binom{\hat{\ell}(u)}{2}+|U|\sum_{\ell\in L}\hat{u}(\ell)+\frac{1}{2}\sum_{\ell\in L}\hat{u}(\ell)\left(2\hat{o}(\ell)-\hat{u}(\ell)-1\right).$$

For convenience, let us denote the components of this sum by

$$\mathscr{S}_{\mathrm{U}}=\sum_{u\in U}\binom{\hat{\ell}(u)}{2};\quad \mathscr{S}_{\mathrm{L}}=\sum_{\ell\in L}\hat{u}(\ell);\quad \text{and}\quad \mathscr{S}_{\mathrm{UL}}=\sum_{\ell\in L}\hat{u}(\ell)\left(2\hat{o}(\ell)-\hat{u}(\ell)-1\right).$$

In summary, we have the following:

▶ **Theorem 4.1.** *The total number of extensions of the last three cubes is*

$$\binom{|U|}{3}+\mathscr{S}_U+|U|\mathscr{S}_L+\mathscr{S}_{UL}/2.$$

We present an efficient local-update mechanism for maintaining $\mathscr{S}_{\mathrm{U}}$, $\mathscr{S}_{\mathrm{L}}$, and $\mathscr{S}_{\mathrm{UL}}$. The central observation is that when a cube is moved from U to the polycube, the resulting changes to these quantities depend only on the local configuration of its immediate neighbors and their current states. Exploiting this locality allows us to avoid repeated global recomputation of these quantities. Our approach distinguishes between two phases of the recursion. The values of $\mathscr{S}_{\mathrm{U}}$, $\mathscr{S}_{\mathrm{L}}$, and $\mathscr{S}_{\mathrm{UL}}$ are computed, using a brute-force procedure, only when the recursion reaches depth $n-4$. When the fourth-to-last cube is added, the values of $\mathscr{S}_{\mathrm{U}}$, $\mathscr{S}_{\mathrm{L}}$, and $\mathscr{S}_{\mathrm{UL}}$ are updated incrementally, replacing the last three levels of the recursion by applying local correction rules derived directly from the expressions that define $\mathscr{S}_{\mathrm{U}}$, $\mathscr{S}_{\mathrm{L}}$, and $\mathscr{S}_{\mathrm{UL}}$. Each update affects only cubes whose neighborhood relationships change as a result of local state transitions between the sets U, L, and O. To illustrate the local nature of our update rules, consider a cube $u \in U$. Its contribution appears only in $\mathscr{S}_{\mathrm{U}}$ and depends solely on the number $\hat{\ell}(u)$ of its neighbors currently in L. Consequently, the contribution of u changes only when the L-neighborhood of u changes. If a neighboring cube $v \in L$ is moved into U, then $\hat{\ell}(u)$ decreases by one, and $\mathscr{S}_{\mathrm{U}}$ is updated by subtracting the corresponding change. Conversely, if a new neighbor of u is added to L, then $\hat{\ell}(u)$ increases by one, and $\mathscr{S}_{\mathrm{U}}$ is updated accordingly. This local-update framework eliminates the need to iterate over the entire untried set after the fourth-to-last cube is added, as the final count is obtained directly via Theorem 4.1. In the full version of the paper, we present the complete set of local update rules and provide a detailed complexity analysis that accounts for the resulting performance improvement over Dodds's original implementation.

Table 1 A comparison between our and Dodds's versions (3D polycubes); all times are in seconds.

n	Original	Dodds's	Ours	Speedup
13	42	0.9995	0.823	1.21 ± 0.05
14	322	7.095	5.890	1.20 ± 0.02
15	2,315	51.005	41.871	1.22 ± 0.01
16	17,825	374.269	304.717	1.23 ± 0.00
17		2,789.502	2,273.903	1.23 ± 0.01
18		21,528.939	17,039.300	1.26 ± 0.01

5 Implementation Details

Redelmeier's algorithm lends itself naturally to parallelization because each subtree of the computation corresponds to an independent extension of an intermediate polycube. Since the execution of every subtree of the computation returns the algorithm to the same state (aside from counters that aggregate the number of polycubes of each size and the untried set from which the expanded cell is removed), it can be executed separately and independently either on a different thread or on a remote machine. Thus, we designed a parallel and distributed architecture based on a task-level decomposition. This approach mirrors the strategy outlined by Aleksandrowicz and Barequet [2] and enhances it with respect to using modern super computers, each having a large number of cores. Specifically, a task-generation process executes the recursion, using heuristics at each branch to decide whether to continue or to stop and create a task that will process the corresponding computation subtree. The task is then placed in a central queue, and client processes repeatedly request tasks from a server that assigns them from the queue. The clients process tasks in parallel and report their partial results back to the server, which aggregates them to produce the final count.

6 Results

We applied our improved version of Redelmeier's algorithm to counting polycubes in higher dimensions and obtained new counts in dimensions 3 through 5.

It has been proven by Barequet *et al.* [4] that when we fix the size n and refer to the dimension d as a variable, the quantity $A_d(n)$ is a polynomial in d of degree $n-1$. These polynomials are currently known for $1 \le n \le 14$ [6], that is, values of $A_d(n)$ are currently known in *any* dimension up to size 14. Hence, at this time, in order to improve on the state-of-the-art, we had to compute values of $A_d(n)$ for n at least 15.

Values of $A_3(n)$ were previously known up to $n = 22$ [8], values of $A_4(n)$ up to $n = 16$ [6], and values of $A_5(n)$ up to $n = 16$ [6]. Results obtained by our software are consistent with all previously-reported values. We found the new values $A_3(23) = 2,397,764,499,745,276,015$, $A_3(24) = 18,813,985,802,083,290,357$, $A_4(17) = 8,282,281,164,122,166$, $A_4(18) = 99,695,859,234,798,748$, $A_4(19) = 1,206,367,179,263,956,112$, $A_4(20) = 14,666,598,062,721,100,290$, $A_5(16) = 60,453,385,633,783,062$, and $A_5(17) = 997,838,348,899,663,080$. All computations were performed on the Technion's Newton HPC cluster, which consists of multiple nodes equipped with Intel Xeon and AMD EPYC CPUs, ranging from 64 to 256 cores per node, and 187 GB to 2 TB of RAM per node. Computing $A_3(23)$ and $A_3(24)$ required 19 and 160.8 CPU years, resp.; computing $A_4(17)$, $A_4(18)$, $A_4(19)$, and $A_4(20)$ required 19 days, 212 days, 6.5 years, and 70.9 CPU years, resp.; and computing $A_5(16)$ and $A_5(17)$ required 2 hours and 3 CPU days, resp.

Table 2 Comparison of lower bounds on the growth constants λ_d.

d	Previous	Improved
3	6.6621	6.8942
4	9.7714	10.2143
5	12.9569	13.3481

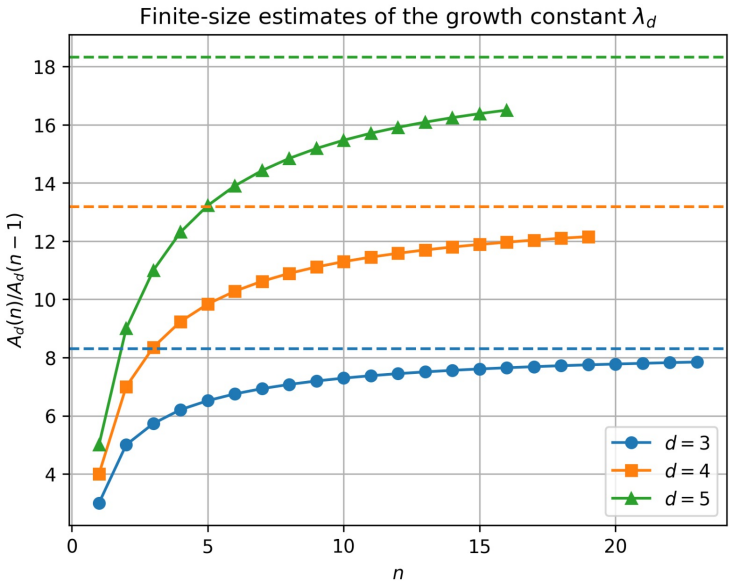


Figure 3 Numerical estimates of the growth constants λ_d derived from computed values of $A_d(n)$.

We compared our version with Dodds's version of the algorithm using the `hyperfine` benchmarking tool. For each 3D polycube of size between 13 and 18, we executed 10 to 50 runs. Our implementation consistently outperformed Dodds's version (while both running on the same hardware), achieving speedups between 1.20× and 1.26×. The relative speedup showed a slight upward trend with increasing polycube size. Table 1 shows the mean running times in these experiments.

Using the newly-computed values of $A_3(n)$, $A_4(n)$, and $A_5(n)$, we applied the recursive-bounding technique of Ref. [3] to improve the best-known lower bounds on the growth constants λ_3, λ_4, and λ_5. Table 2 compares the previous bounds reported there with the improved bounds obtained in this work. Figure 3 shows the numerical approximations $\lambda_3 \approx 8.31$, $\lambda_4 \approx 13.19$, and $\lambda_5 \approx 18.33$, obtained by using Richardson extrapolation through ChatGPT for estimating the limits of the sequences of the ratios $A_d(n)/A_d(n-1)$ (based on the known values).

References

1 G. Aleksandrowicz and G. Barequet. Counting d-dimensional polycubes and nonrectangular planar polyominoes. *Int. J. of Computational Geometry and Applications*, 19(3):215–229, 2009.

2 G. Aleksandrowicz and G. Barequet. Counting polycubes without the dimensionality curse. *Discrete Mathematics*, 309(16):4576–4583, 2009.

3 G. Barequet, G. Ben-Shachar, and M.C. Osegueda. Concatenation arguments and their applications to polyominoes and polycubes. *Computational Geometry*, 98:101790, 2021.

4 R. Barequet, G. Barequet, and G. Rote. Formulae and growth rates of high-dimensional polycubes. *Combinatorica*, 30(3):257–275, 2010.

5 I. Jensen. Enumerations of lattice animals and polygons. *J. of Statistical Physics*, 109(3-4):577–602, 2003.

6 S. Luther and S. Mertens. Counting lattice animals in high dimensions. *Institut für Theoretische Physik, Otto-von-Guericke Universität*, 2011.

7 P. Thompson. Counting polycubes of size 21, 2024. URL: `https://philthompson.me/2024/Counting-Polycubes-of-Size-21.html`.

8 P. Thompson. Counting polycubes of size 22, 2024. URL: `https://philthompson.me/2024/Counting-Polycubes-of-Size-22.html`.

9 D.H. Redelmeier. Counting polyominoes: Yet another attack. *Discrete Mathematics*, 36(2):191–203, 1981.

10 The OEIS Foundation, Inc. The On-Line Encyclopedia of Integer Sequences. available at `https://oeis.org`.

A 44-Point Configuration Not Coverable by Disjoint Unit Disks

Takuto Nakai[1] **and Shuya Bundo**[2]

1 **Independent Researcher**
`takutoyakke@gmail.com`
2 **E9Technologies Inc.**
`s.bundo@e9tech.com`

Abstract

We study the problem of covering a set of points in the plane with pairwise disjoint open unit disks. Let k denote the smallest number of points for which there exists a configuration that cannot be covered. It is known that $k \geq 13$. A floating-point-based computer search by Aloupis et al. reported a 45-point configuration that appears not to be coverable, suggesting $k \leq 45$, but leaving a fully rigorous proof open.

In this paper, we prove $k \leq 44$ by presenting a 44-point configuration that cannot be covered by pairwise disjoint open unit disks. Our proof is computer-assisted but exact: we scale the input coordinates to integers and verify non-coverability using exact arithmetic (without floating-point computations). The verification proceeds in two stages: (i) enumerate candidate point-to-disk assignments via exact-cover enumeration, and (ii) for each assignment, determine feasibility by maintaining axis-aligned rectangles for disk centers and applying recursive subdivision with conservative geometric pruning. This yields a rigorous computer-assisted proof independent of floating-point tolerances.

1 Introduction

In 2008, Naoki Inaba proved by the probabilistic method that every configuration of ten points in the plane can be covered by pairwise disjoint unit disks, and posed the question: what is the maximum number of points for which a covering is always guaranteed to exist? [4, 5]

In this paper, we study whether a finite point set in the plane can be covered by a family of pairwise disjoint unit disks. Throughout the paper, all disks are open. Let k denote the smallest number of points for which there exists a configuration that cannot be covered. Inaba's result implies $k \geq 11$, and Aloupis et al. [2] later improved the lower bound to $k \geq 13$. On the other hand, the same paper proved $k \leq 50$ via a 50-point lattice configuration, and reported a 45-point configuration that appears not to be coverable. However, since the 45-point search relied on floating-point arithmetic, a rigorous proof of $k \leq 45$ remained open.

1.1 Results

▶ **Theorem 1.1** (Main result). *There exists a configuration of* 44 *points in the plane that cannot be covered by any family of pairwise disjoint unit disks. Consequently,* $k \leq 44$.

The proof of Theorem 1.1 is computer-assisted but exact: we scale the input coordinates so that they are integers and perform all subsequent decisions using exact arithmetic. This yields a verification that does not depend on numerical errors arising from floating-point arithmetic.

Our contributions are as follows:

- We present a 44-point configuration that cannot be covered (a rigorous proof that $k \leq 44$).

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

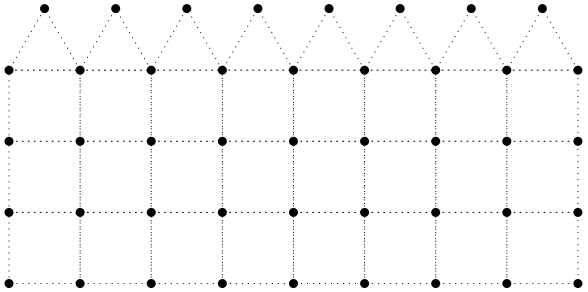


Figure 1 A 44-point configuration not coverable by pairwise disjoint unit disks. The points consist of a 9×4 grid with spacing $a = 0.2531$ and 8 additional points placed above the top row.

- We propose a verification algorithm based on exact arithmetic.

In the next section, we define the 44-point configuration, then describe the verification algorithm, and finally conclude with remarks and future work.

2 A 44-Point Configuration

Figure 1 shows our 44-point configuration. We first place 36 points on a 9×4 grid with spacing $a = 0.2531$. Next, for each pair of adjacent points on the top row, we add one point above them so that the three points form (approximately) an equilateral triangle; this adds 8 points. This gives 44 points in all. This configuration was initially found by one of the authors (Nakai).

The spacing is motivated by the following observation: when three mutually tangent unit disks are arranged, the side length s of the largest square whose four vertices lie on the three boundary circles satisfies $\sqrt{s - s^2/4} + \sqrt{1 - s^2/4} = \sqrt{3} - s$, giving $s \approx 0.25314$. With grid spacing at this scale, any unit disk centered in such a gap must cover at least one nearby grid point. A similar analysis for the equilateral triangle gives a threshold of $2 - \sqrt{3} \approx 0.26795$, which is larger than 0.2531, so replacing some squares with triangles preserves the non-coverability. A grid with only four rows can be covered by placing unit disks along each row, and a grid with only eight columns can also be covered, so the 9×4 grid with additional triangular points appears to be close to the minimum size needed.

For the input to our verification algorithm, however, we do not use exact equilateral triangles. Instead, we use triangles with base length 0.2531 and height $0.21919103 \simeq (\sqrt{3}/2) \times 0.2531$. The reason is that, after scaling by a factor of 10^8, all vertex coordinates become integers, allowing the instance to be handled by exact arithmetic on integers. We believe that the configuration with exact equilateral triangles is also not coverable, but we do not address that case in this paper.

Our implementation is available at [1]. For $a = 0.2531$, the verification terminates with all candidate partitions ruled out and no undetermined cases. The same holds for $a = 0.25$, with the triangle height rounded similarly; the running time is approximately 11 seconds on a MacBook Air with an Apple M1 processor and 16 GB of RAM, compared with approximately 4 hours for $a = 0.2531$. For $a = 0.2493$ and $a = 0.2532$, the configuration is coverable; see Figure 2. Note that $a = 0.2532$ exceeds the threshold $s \approx 0.25314$, consistent with the geometric observation above. We have also verified that removing any single point from our 44-point configuration yields a coverable 43-point set.

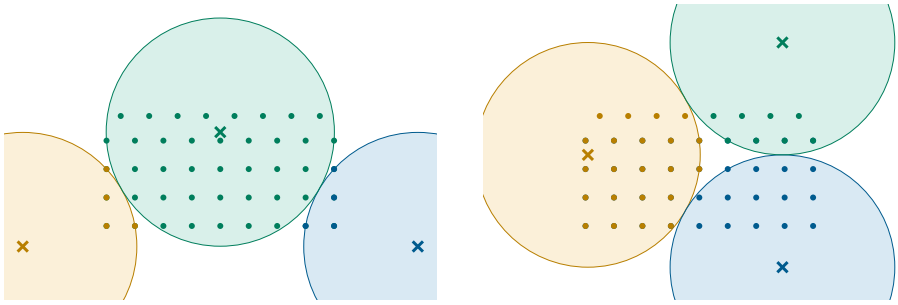


Figure 2 Coverable 44-point configurations ($a = 0.2493$ left, $a = 0.2532$ right).

3 Verification Algorithm

3.1 Overview

In this section we describe the algorithm used for the verification.

To avoid errors from floating-point arithmetic, our algorithm performs all computations using exact arithmetic. Specifically, we fix a large positive integer scale and restrict ourselves to point sets whose coordinates are integer multiples of 1/scale. By multiplying every coordinate by scale, we obtain an equivalent instance with integer coordinates. If we scale the unit-disk radius by the same factor, the problem is transformed as follows: given n points $P = \{\boldsymbol{p}_1, \boldsymbol{p}_2, \ldots, \boldsymbol{p}_n\} \subseteq \mathbb{Z}^2$ and radius $R =$ scale, find a placement of pairwise disjoint disks of radius R that covers P, prove that no such placement exists, or return undetermined.

Since all computations use exact arithmetic, there is no rounding error. When the algorithm reports that a covering exists or does not exist, this conclusion is mathematically rigorous. The algorithm may return undetermined if recursive subdivision reaches the granularity limit without deciding feasibility.

Our algorithm consists of two steps: (1) **Partitioning**: enumerate candidate assignments of points to disks, pruning clearly infeasible ones; and (2) **Placement Search**: for each partition, search for a disk placement satisfying all constraints. If every candidate partition is ruled out in the placement search, the algorithm concludes that no covering exists.

Aloupis et al. [2] also adopt a two-stage approach, exploring assignments dynamically during search. We instead enumerate partitions upfront via the exact-cover method of Chun et al. [3], and adapt Aloupis et al.'s rectangular subdivision to exact integer arithmetic. The exact-cover pruning substantially reduces the number of candidate partitions in our setting, making upfront enumeration practical and the separation of the two stages natural.

More specifically, we maintain a rectangular region for each disk center and narrow it down by recursive subdivision. In the original method of Aloupis et al., floating-point arithmetic was used, and they noted that the subdivision might not converge. In our algorithm, rectangle endpoints are stored as integers and all distance comparisons use squared values, avoiding irrational numbers; subdivision stops when all side lengths reach 1. If the existence of a valid placement is not determined at that point, the algorithm returns undetermined.

3.2 Partitioning

In the Partitioning step, we enumerate candidate partitions that could be realized by a disk placement covering the given point set.

As a pruning technique, we use the exact-cover approach of Chun et al. [3]. Here, *exact*

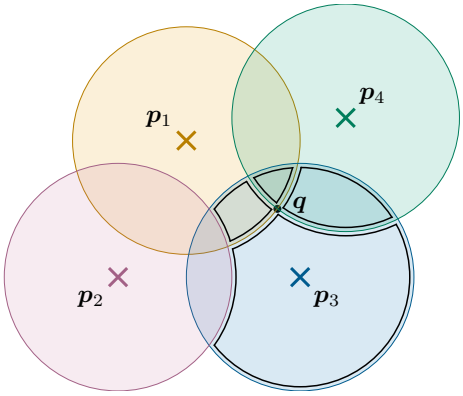


Figure 3 An arrangement of four circles centered at p_1, p_2, p_3, p_4. The point q is an intersection of the circles centered at p_1 and p_4. The four neighboring cells of q are distinguished by whether they lie inside or outside each of these two circles, yielding four combinations. Since q lies outside the circle of p_2 and inside the circle of p_3, all four cells lie outside of p_2 and inside of p_3. Thus, a disk centered in each cell covers: $\{p_3\}$, $\{p_1, p_3\}$, $\{p_3, p_4\}$, or $\{p_1, p_3, p_4\}$, respectively.

cover refers to the problem of finding a set of disk centers that cover a given point set; unlike the disjoint covering considered in this paper, overlaps between disks are allowed, and the only requirement is that each point is covered by exactly one disk.

We now describe Chun et al.'s approach in our setting. First, a point p is covered by a disk of radius R centered at c if and only if c lies in the disk of radius R centered at p. Therefore, finding an exact cover of $P = \{p_1, \ldots, p_n\}$ by disks of radius R reduces to the following: given the family of disks $\{D_{p_1}, \ldots, D_{p_n}\}$ of radius R centered at the points, find a set of points (disk centers) such that each disk contains exactly one selected point.

The boundary circles of these disks subdivide the plane into cells. To enumerate the cells, we consider all intersection points of pairs of boundary circles. For each intersection point, we enumerate the neighboring cells by considering, for each circle passing through the point, whether the cell lies inside or outside that circle (see Figure 3).

Since the input points are lattice points and the radius R is an integer, each intersection point has coordinates of the form $a + b\sqrt{c}$ for some rationals a, b and non-negative integer c. This allows exact computation of whether a point lies inside a given disk.

For any two points in the same cell, the set of disks that contain them is identical, so only the choice of cell matters, not the exact position of each disk center. This reduces the problem to a discrete exact-cover problem, and all solutions can be enumerated. Since any disjoint covering is in particular an exact cover, this enumeration does not miss any valid disjoint covering.

3.3 Placement Search

For each partition obtained in the Partitioning step, the Placement Search step searches for a disk placement satisfying all constraints. This step is based on the method of Aloupis et al. [2].

We represent the feasible region for each disk center as an axis-aligned rectangle. The center of a disk covering a point p_i must lie inside the disk of radius R centered at p_i. In particular, it must lie inside the open axis-aligned square of side length $2R$ centered at p_i (a conservative estimate). Similarly, the center of a disk covering points $p_{i_1}, p_{i_2}, \ldots, p_{i_k}$

must lie in the intersection of the open axis-aligned squares of side length $2R$ centered at each of these points. Since the intersection of rectangles is a rectangle (or empty), we can conservatively compute an initial feasible region for each disk center. (If this region is empty, the partition is ruled out.)

We then refine the feasible regions by recursively subdividing them. Given axis-aligned rectangles as feasible regions, we perform the following test-and-split procedure:

1. For each rectangle, select a representative point at the nearest integer coordinates to its center as a candidate disk center. If the resulting centers satisfy the conditions of a disjoint covering, a solution is found and we terminate.
2. Otherwise, check whether the current rectangles can be ruled out using the following sufficient conditions:
   - For a rectangle B_k and a point $\boldsymbol{p}_i$ that the corresponding disk must cover, if $\min_{x \in B_k} \|x - \boldsymbol{p}_i\| \geq R$, then no disk centered in B_k can cover $\boldsymbol{p}_i$, so this branch is infeasible.
   - For two distinct rectangles B_k and B_l, if $\max_{x \in B_k, y \in B_l} \|x - y\| < 2R$, then any disks centered in B_k and B_l must overlap, so this branch is infeasible.

   If either condition holds, the current branch is ruled out.
3. If neither condition holds, subdivide further: select a rectangle with the longest side, bisect it at an integer midpoint, and apply the same procedure recursively. If all rectangles have side length 1, return undetermined.

We repeat this procedure for each partition. If any partition yields a solution, we terminate. If every partition is ruled out, we conclude that no disjoint covering exists. If any partition returns undetermined and no solution has been found, the overall result is undetermined.

4 Conclusion

We have shown that $k \leq 44$ by presenting a 44-point configuration that cannot be covered by pairwise disjoint unit disks. The configuration consists of a 9×4 grid with spacing $a = 0.2531$ (36 points) plus 8 additional points forming approximate equilateral triangles above the top row, with triangle heights given by a rational approximation to ensure integer coordinates after scaling.

We also proposed a verification algorithm using exact arithmetic, addressing the lack of rigor in previous floating-point-based searches. The algorithm proceeds in two stages: (i) partition enumeration with pruning based on exact cover, and (ii) recursive subdivision of rectangular feasible regions for disk centers. This yields a rigorous computer-assisted proof of non-coverability (with no undetermined cases for our instance).

Combined with the lower bound $k \geq 13$ by Aloupis et al. [2], the current best bounds are $13 \leq k \leq 44$. Future work includes finding an analytical proof of non-coverability and further tightening these bounds.

Acknowledgements

We thank evima, who runs the Discord server "evima Lab Secret Base" (えびまラボ秘密基地), for helpful discussions and comments related to this problem.

References

1 Supplemental code. https://github.com/sammiya/disjoint-unit-disk-cover.

2 Greg Aloupis, Robert A. Hearn, Hirokazu Iwasawa, and Ryuhei Uehara. Covering points with disjoint unit disks. In *Canadian Conference on Computational Geometry*, 2012. URL: https://2012.cccg.ca/papers/paper13.pdf.

3 Ji Hoon Chun, Christian Kipp, and Sandro Roch. On exact covering with unit disks. *Computational Geometry*, 129:102193, 2025. doi:10.1016/j.comgeo.2025.102193.

4 Naoki Inaba. Ten dots, 2008. Original title: 「１０個の点」(in Japanese). URL: http://inabapuzzle.com/hirameki/suuri_4.html.

5 Naoki Inaba. Ten dots - answer, 2008. Original title: 「１０個の点解答」(in Japanese). URL: http://inabapuzzle.com/hirameki/suuri_ans4.html.

Grounded String Representations of Series-Parallel Graphs without Transitive Edges*

Sabine Cornelsen[1], Jan Kratochvíl[2], Miriam Münch[3], Giacomo Ortali[4], Alexandra Weinberger[5], and Alexander Wolff[6]

1 **University of Konstanz**
sabine.cornelsen@uni-konstanz.de
2 **Faculty of Mathematics and Physics, Charles University**
honza@kam.mff.cuni.cz
3 **University of Passau**
muenchm@fim.uni-passau.de
4 **Università Telematica Internazionale Uninettuno**
giacomo.ortali@uninettunouniversity.net
5 **FernUniversität in Hagen**
alexandra.weinberger@fernuni-hagen.de
6 **Universität Würzburg**

Abstract

In a *grounded string representation* of a graph there is a horizontal line ℓ and each vertex is represented as a simple curve below ℓ with one end point on ℓ such that two curves intersect if and only if the respective vertices are adjacent. A grounded string representation is a *grounded* L-⅃-*representation* if each vertex is represented by a 1-bend orthogonal polyline. It is a *grounded* L-*representation* if in addition all curves are L-shaped. We show that every biconnected series-parallel graph without edges between the two vertices of a separation pair (i.e., *transitive edges*) admits a grounded L-⅃-representation if and only if it admits a grounded string representation. Moreover, we can test in linear time whether such a representation exists. We also construct a biconnected series-parallel graph without transitive edges that admits a grounded L-⅃-representation, but no grounded L-representation.

Related Version http://arxiv.org/abs/2603.02827

1 Introduction

In this paper, we are interested in intersection representations of graphs where each vertex is represented by an orthogonal polyline with one bend. In particular, we consider the grounded case where all polylines start at a horizontal line and are drawn in the half-plane below that line. More precisely, in a *string representation* [7, 13] of a graph G, each vertex v of G is represented by a simple curve $\mathsf{L}(v)$ and the curves of two vertices v and w intersect if and only if v and w are adjacent. A string representation is a *1-string representation* if any two curves intersect at most once. In a *grounded string representation* there is a horizontal line ℓ such that the curve of any vertex v is below ℓ and has one endpoint on ℓ. This point is called the *anchor* $\ell(v)$ of v. A graph admits a grounded string representation if and only it admits an *outerstring representation* [10] where each curve is drawn within a disk and the anchors are drawn on the boundary of this disk. A graph is *outerstring* if it has a grounded string representation and *outer-1-string* if it has a grounded string representation that is

* This research was initiated at the Bertinoro Workshop on Graph Drawing 2025. The second author acknowledges support of Czech Science Foundation by Research grant No. GAČR 23-04949X.

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

1-string. While the computational complexity of recognizing outer-1-string graphs is open, recognizing outerstring graphs is NP-hard [12].

Curves with a low complexity are of particular interest. E.g., in [2], monotone curves with few bends and slopes ± 1 were studied. These are special rotated orthogonal curves. A string representation is *orthogonal* if each curve is an orthogonal polyline, i.e., a sequence of vertical and horizontal segments. An orthogonal string representation is an *L-⊥-representation* if each curve has at most one bend and this point is a bottommost point of the curve. If a graph has a (grounded) L-⊥-representation, then it has a (grounded) string representation in which every curve is a straight-line segment [11]. An orthogonal representation is an *L-representation* if every curve has exactly one bend and this bend is the leftmost and bottommost point of the curve. Every planar graph admits an L-representation [8].

Jelínek and Töpfer [9] discussed a hierarchy of grounded string representations. In particular, they showed that not every graph that admits a grounded L-⊥-representation also admits a grounded L-representation. Outerplanar graphs, interval graphs, and complete graphs always admit a grounded L-representation [9]. Monotone L-representations, where each curve is anchored at its bend on a diagonal line, can always be found for outerplanar graphs, interval graphs, 2D-ray graphs, convex bipartite graphs, and complete graphs [1, 4].

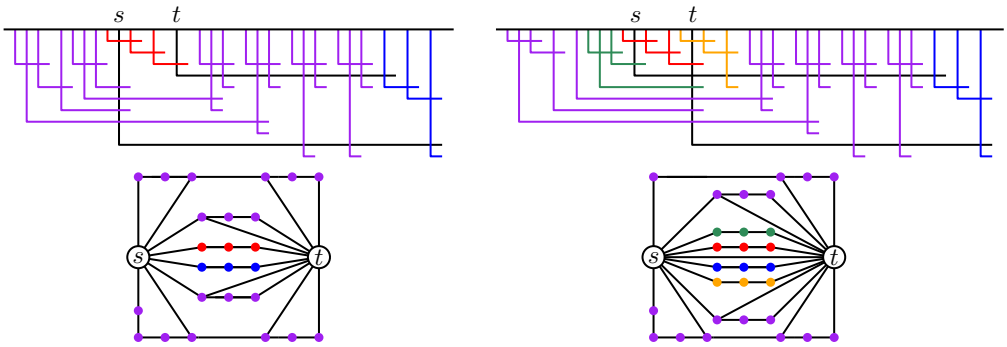
A graph together with an ordering of the vertices is an *ordered graph*. In a *constrained grounded string representation* [13] of an ordered graph, the anchors are in the order prescribed by the input order. Given an ordered graph with n vertices and m edges, it can be decided in $\mathcal{O}(n^4)$ time whether it admits a constrained grounded L-representation [9], in $\mathcal{O}(m + n \log n)$ time whether it has a constrained monotone L-representation [1], and in $\mathcal{O}(n^2)$ time whether it admits a constrained grounded L-⊥-representation [3]. In general, one bend per curve (assuming orthogonal curves) does not suffice for ordered trees or cycles with a constrained grounded 1-string representation, but two bends per curve are always sufficient and such a representation can be computed in linear time [3].

Our Results. We first give a necessary condition for biconnected outerstring graphs in terms of components of a separation pair; see Section 2. In Theorem 3.2, we then demonstrate that these conditions even guarantee grounded L-⊥-representations for biconnected series-parallel graphs without transitive edges; where a biconnected graph is *series-parallel* if and only if it does not contain a subdivision of the complete graph K_4 with four vertices [6], and a *transitive edge* is an edge between the two vertices of a separation pair. In this case, the necessary and sufficient condition can be tested in linear time; see Corollary 3.3. Finally, we give an example of a biconnected series-parallel graph without transitive edges that admits a grounded L-⊥-representation, but no grounded L-representation; see Theorem 3.4.

For full proofs, we refer to the long version of the paper [5].

2 A Necessary Condition for Biconnected Outerstring Graphs

We give a necessary condition for biconnected outerstring graphs in terms of components of a separation pair. For a graph G and a vertex s of G, we let $G - s$ denote the graph obtained from G by removing s and its incident edges. Given a biconnected graph G, a *separation pair* is a pair (s, t) of vertices of G such that $G - s - t$ is disconnected. A *component* G' of G w.r.t. (s, t) is the subgraph induced by s, t, and a connected component of $G - s - t$. The vertices s and t are the *poles* of G'; the other vertices of G' are *internal vertices*. A component G' w.r.t. a separation pair (s, t) is *heavy* if it contains an internal vertex that is neither adjacent to s nor to t; otherwise G' is *light*. A component G' is *critical* if



(a) poles not adjacent **(b)** poles adjacent

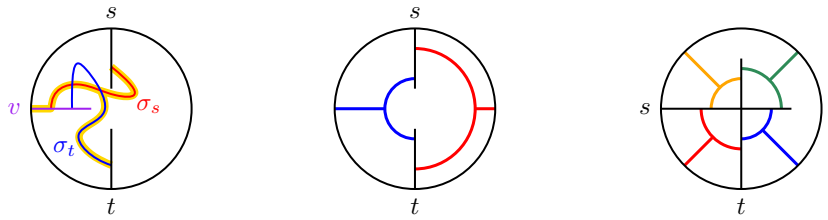
Figure 1 There can be many heavy components w.r.t. a separation pair (s,t) in a grounded L-representation; the purple components are not critical.

1. it contains an internal vertex v that is neither adjacent to s nor to t (in particular G' is heavy), and
2. there is a path $s - p_s - v - p_t - t$ in G' such that no internal vertex in the subpath p_s is adjacent to t and no internal vertex in the subpath p_t is adjacent to s.

A biconnected outer-1-string graph can have many heavy components w.r.t. some separation pair; see Fig. 1. However, only few of them can be critical.

▶ **Lemma 2.1.** *Let (s,t) be a separation pair of a biconnected outerstring graph G.*

1. *If s and t are not adjacent, then at most two components w.r.t. (s,t) are critical.*
2. *If s and t are adjacent and the curves of s and t cross once, then at most four components w.r.t. (s,t) are critical.*



(a) curves **(b)** poles not adjacent **(c)** poles adjacent

Figure 2 Illustration of Lemma 2.1.

Proof. Consider an outerstring representation of G. Let G' be a critical component w.r.t. (s,t), and let v be a vertex in G' that is neither adjacent to t nor to s, together with an s-t-path $s - p_s - v - p_t - t$ such that no vertex in p_s is adjacent to t and no vertex in p_t is adjacent to s. In the representation of the vertices of p_s, consider the parts between the intersection points between the curves of any two consecutive vertices. This yields a simple curve σ_s. Define σ_t analogously for p_t. By the choice of p_s and p_t, σ_t must not intersect the curve of s and σ_s must not intersect the curve of t; see Fig. 2a. The combination of σ_s, σ_t, and the representation of v contains a 3-star connecting the curve of t, the curve of s, and the boundary at the anchor of v; see the thick yellow curve in Fig. 2a. If s and t are not

adjacent, there can be at most two such 3-stars with the property that they are disjoint; see Fig. 2b. If the curves of s and t cross exactly once, then there can be at most four such 3-stars with the property that they are disjoint; see Fig. 2c. ◀

3 Series-Parallel Graphs without Transitive Edges

In this section, we show that the necessary condition in Lemma 2.1 is also sufficient for biconnected series-parallel graphs without transitive edges and that in this case the condition can be tested in linear time. If there are no transitive edges, then the second condition in the definition of critical components is obsolete, and we obtain the following lemma.

▶ **Lemma 3.1.** *A component of a biconnected series-parallel graph without transitive edges is critical if and only if it is heavy.*

The separation pair (s, t) is *k-heavy* if it has k heavy components. A graph G is *k-heavy* if k is the smallest integer such that every separation pair of G is at most k-heavy.

▶ **Theorem 3.2.** *If G is a biconnected series-parallel graph without transitive edges, then the following statements are equivalent.*

1. *G is at most 2-heavy.*
2. *G has an outerstring representation.*
3. *G has a grounded ⌞-⌟-representation.*

Proof Sketch. (3) implies (2). If there are no transitive edges, then there is no edge between any two poles. Thus, by Lemmas 2.1 and 3.1, if G is outerstring, then G is at most 2-heavy. It remains to show that (1) implies (3).

Let (s, t) be a k-heavy separation pair with maximum k, and let G_1 and G_2 be two components of (s, t) such that G_1 is heavy if $k \geq 1$, and G_2 is heavy if $k = 2$. For $i \in \{1, 2\}$, let p_i be a longest s-t-path in G_i. The paths p_1 and p_2 are induced paths.The concatenation of p_1 and the reverse of p_2 is the *heavy cycle* c. We draw c as shown in Fig. 3a.

In order to also insert the remainder of the graph, consider a connected component G' of $G - p_1 - p_2$. Then G' is connected to exactly two vertices u and v on p_1 or on p_2 and G' together with u and v is light w.r.t. (u, v). Moreover, in a series-parallel graph without transitive edges, a light component G' w.r.t. a separation pair (u, v) looks as follows: G' either contains a single vertex adjacent to both u and v, or G' contains an independent set V_u of vertices adjacent to u and an independent set V_v of vertices adjacent to v such that V_v and V_u are disjoint and V_v or V_u contains exactly one vertex. Moreover, the separation pairs corresponding to these light components are nested along p_1 and p_2.

Now we draw the components of $G - p_1 - p_2$ inside out as shown in Figs. 3b and 3c. We start with the components that are not adjacent to s. Let $p = c - s$. For two vertices u and v on p, we write $u < v$ if the vertical part of u is to the left of the vertical part of v. Let G' and G'' be two components of $G - p_1 - p_2$, and let $u < v$ and $u'' < v''$ be the pairs of vertices on p that are adjacent to G' and G'', respectively. We process G' before G'' if

1. G' is nested inside G'', i.e., if $u'' \leq u$ and $v < v''$, or
2. G' is to the left of G'', i.e., $v \leq u''$.

When drawing a component G' with poles $u < v$ on p, we use the horizontal part of u and the vertical part of v to connect the curves of G'. See Fig. 3b.

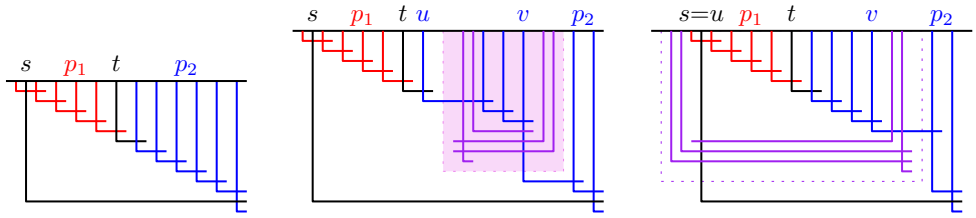
If G' contains a single vertex w adjacent to u and v, we anchor w immediately to the right of the anchor of u and draw the horizontal part of $\mathsf{L}(w)$ immediately above the horizontal

part of $\mathsf{L}(v)$ until it intersects the vertical part of v. Otherwise, we anchor the vertices adjacent to v immediately to the left of the anchor of the successor of v on p and draw them as J. Then we anchor the vertices adjacent to u immediately to the right of the anchor of u and draw them as L. In any case, we draw the horizontal parts of the curves immediately above the horizontal part of v. The horizontal parts of the vertices adjacent to v must be long enough such that they intersect the vertical parts of the vertices adjacent to u.

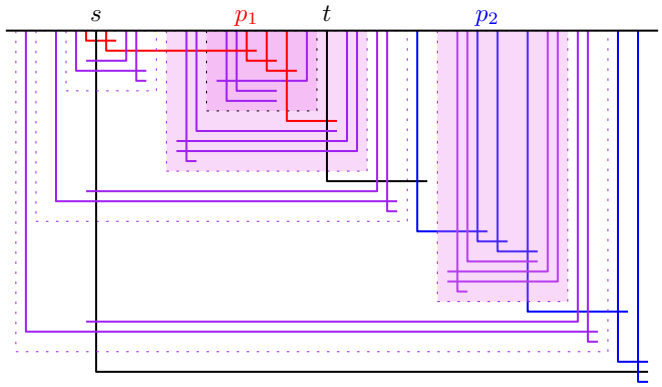
In the end, we draw the components that are adjacent to s. See Fig. 3c. We draw a component G' before a component G'' if $v < v''$ where v and v'' are the vertices on p other than s incident to G' and G'', respectively. Observe that v and v'' cannot be incident to s since otherwise p_1 or p_2 would not have been longest paths.

When processing a component G' with poles s and v on p, we use the vertical part of s and the horizontal part of v. Let v' be the successor of v on p. If G' contains a vertex w adjacent to s and v, then we anchor w immediately to the left of the anchor of v' and draw the horizontal part of the J representing w immediately above the horizontal part of $\mathsf{L}(v)$ until it reaches the vertical part of $\mathsf{L}(s)$. All other vertices are drawn as Ls. We first anchor the vertices adjacent to s to the left of any anchor and then the vertices adjacent to v immediately to the left of v'. We draw the horizontal parts of the curves immediately above the horizontal part of v'. The horizontal parts of the vertices adjacent to s must be long enough such that they intersect the vertical parts of the vertices adjacent to v. ◀

▶ **Corollary 3.3.** *It can be tested in linear time whether a biconnected series-parallel graph without transitive edges is an outerstring graph, an outer-1-string graph, admits a grounded*



(a) heavy cycle $c = p_1p_2$ **(b)** light component without s **(c)** light component with s



(d) light components are disjoint or nested

Figure 3 How to construct a grounded L-J-representation for a series-parallel graph without transitive edges if there is one. The black Ls are the poles, the red and the blue Ls represent a longest cycle through the poles, and the purple curves represent light components.

string representation with segments, or admits a grounded L-J*-representation.*

Proof Sketch. Since a grounded L-J-representation implies a grounded representation with segments, which in turn implies that the graph is outerstring, the four properties are fulfilled if and only if the graph is at most 2-heavy. By the special structure of light components (see page 4), a component is heavy if and only if a longest path between the poles has length at least four. Proceeding first bottom-up and then top-down in the decomposition tree, we can compute the length of a longest path. ◀

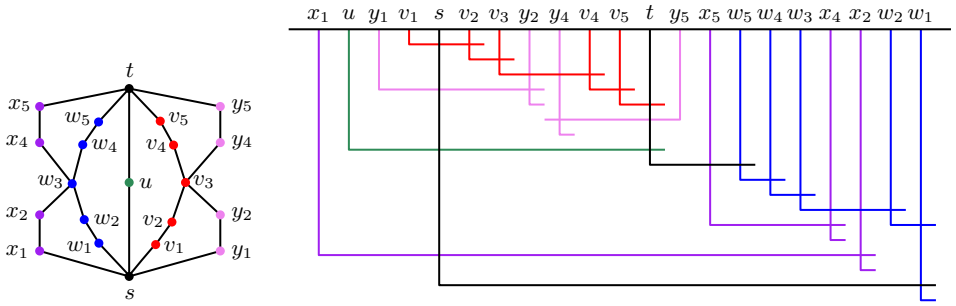


Figure 4 A series-parallel graph that admits a grounded L-J- but no grounded L-representation.

▶ **Theorem 3.4.** *Graph G in Fig. 4 is biconnected, series-parallel, and has no transitive edges. It admits a grounded* L-J*-representation but no grounded* L*-representation.*

Proof Sketch. Fig. 4 depicts the L-J-representation. In order to show that G does not admit a grounded L-representation, we analyze three different cases, depending on the relative positions of $\mathsf{L}(u)$, $\mathsf{L}(s)$, and $\mathsf{L}(t)$; see Fig. 5. The curves $\mathsf{L}(s)$, $\mathsf{L}(t)$, and a part of $\mathsf{L}(u)$ divide the area of the representation into a bounded (shaded) and an unbounded region. We prove that $\{v_2, v_3, v_4\}$ has to be in one of them and $\{w_2, w_3, w_4\}$ in the other. For each case, we show that there are few ways to obtain a grounded L-representation of G without the x_i and y_i and that for each of them, it is impossible to extend it to a representation of G. See Fig. 6 for examples where $\mathsf{L}(y_4)$ or $\mathsf{L}(y_5)$ (left) and $\mathsf{L}(x_1)$ or $\mathsf{L}(x_2)$ (right) cannot be added. ◀

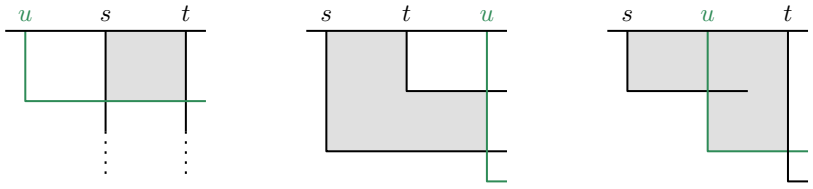


Figure 5 The three cases for the proof of Theorem 3.4.

References

1 Abu Reyan Ahmed, Felice De Luca, Sabin Devkota, Alon Efrat, Md. Iqbal Hossain, Stephen G. Kobourov, Jixian Li, Sammi Abida Salma, and Eric Welch. L-graphs and monotone L-graphs. arXiv report, 2017. URL: `http://arxiv.org/abs/1703.01544`.

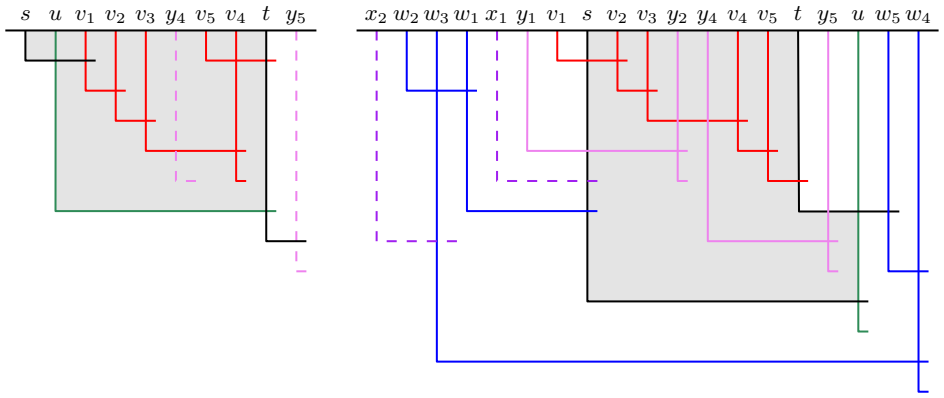


Figure 6 Representations that cannot be extended to a grounded L-representation of G.

2 Todor Antić, Vít Jelínek, Martin Pergel, Felix Schröder, Peter Stumpf, and Pavel Valtr. The bend number of cocomparability graphs. In Vida Dujmović and Fabrizio Montecchiani, editors, *33rd International Symposium on Graph Drawing and Network Visualization (GD)*, volume 357 of *LIPIcs*, pages 10:1–10:23. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2025. doi:10.4230/LIPICS.GD.2025.10.

3 Therese Biedl, Sabine Cornelsen, Jan Kratochvíl, and Ignaz Rutter. Constrained outer-string representations. In Stefan Felsner and Karsten Klein, editors, *32nd International Symposium on Graph Drawing and Network Visualization (GD)*, volume 320 of *LIPIcs*, pages 10:1–10:18. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024. doi:10.4230/LIPIcs.GD.2024.10.

4 Daniele Catanzaro, Steven Chaplick, Stefan Felsner, Bjarni V. Halldórsson, Magnús M. Halldórsson, Thomas Hixon, and Juraj Stacho. Max point-tolerance graphs. *Discrete Appl. Math.*, 216:84–97, 2017. Special Graph Classes and Algorithms – in Honor of Professor Andreas Brandstädt on the Occasion of His 65th Birthday. doi:https://doi.org/10.1016/j.dam.2015.08.019.

5 Sabine Cornelsen, Jan Kratochvíl, Miriam Münch, Giacomo Ortali, Alexandra Weinberger, and Alexander Wolff. Grounded string representations of series-parallel graphs without transitive edges. arXiv report, 2026. URL: https://arxiv.org/abs/2603.02827.

6 Richard James Duffin. Topology of series-parallel networks. *J. Math. Anal. Appl.*, 10:303–318, 1965.

7 Gideon Ehrlich, Shimon Even, and Robert Endre Tarjan. Intersection graphs of curves in the plane. *J. Combin. Theory Ser. B*, 21(1):8–20, 1976. doi:10.1016/0095-8956(76)90022-8.

8 Daniel Gonçalves, Lucas Isenmann, and Claire Pennarun. Planar graphs as L-intersection or L-contact graphs. In Artur Czumaj, editor, *29th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 172–184, 2018. doi:10.1137/1.9781611975031.12.

9 Vít Jelínek and Martin Töpfer. On grounded L-graphs and their relatives. *Electron. J. Comb.*, 26(3):P3.17, 2019. doi:10.37236/8096.

10 Jan Kratochvíl. String graphs. In *Graphs and Other Combinatorial Topics, Proceedings Third Czechoslovak Symposium on Graph Theory, Prague*, volume 59 of *Teubner Texte zur Mathematik*, pages 168–172. Teubner, Berlin, 1982.

11 Matthias Middendorf and Frank Pfeiffer. The max clique problem in classes of string graphs. *Discrete Math.*, 108(1–3):365–372, 1992. doi:10.1016/0012-365X(92)90688-C.

12 Alexandre Rok and Bartosz Walczak. Outerstring graphs are χ-bounded. *SIAM J. Discrete Math.*, 33(4):2181–2199, 2019. doi:10.1137/17M1157374.

13 Frank W. Sinden. Topology of thin film RC circuits. *Bell Syst. Tech. J.*, 45(9):1639–1662, 1966. doi:10.1002/j.1538-7305.1966.tb01713.x.

Planar Convex Obstacle Number of Trees

Emilio Di Giacomo[1], Carolina Haase[2], Philipp Kindermann[2], and Giuseppe Liotta[1]

1 **University of Perugia, Italy**
{emilio.digiacomo, giuseppe.liotta}@unipg.it
2 **Trier University, Germany**
{haasec, kindermann}@uni-trier.de

Abstract

An obstacle representation of a graph G is constructed by drawing vertices at distinct points in the plane and by choosing a set of connected regions (the obstacles) such that G is the visibility graph of those points with respect to the obstacles. The *convex obstacle number* of G is the smallest number of convex obstacles that make it possible to construct an obstacle representation of G. It is known that every outerplanar graph has convex obstacle number five and that the convex obstacle number is at least four for some trees. In this paper we study the *planar convex obstacle number* of trees, that is the smallest number of convex obstacles that are needed to construct a *crossing-free* obstacle representation of a tree. We show that this number is $\Theta(n)$ for n-vertex trees by proving that $\frac{n-1}{4}$ convex obstacles are necessary even for 1-subdivisions of stars, while $n-1$ unit disks are sufficient to construct a crossing-free obstacle representation of any tree.

1 Introduction

An *obstacle representation* of a graph G consists of a set of pairwise disjoint simply-connected closed regions in the plane, called *obstacles*, and a one-to-one mapping of the vertices of G to points in the plane such that two vertices are adjacent in G if and only if the line segment connecting the two corresponding points does not intersect any obstacle. A *convex obstacle representation* of a graph G is an obstacle representation where all obstacles are convex regions. The *obstacle number* $obs(G)$ of a graph G is the smallest number of obstacles in any obstacle representation of G. The *convex obstacle number* $obs_c(G)$ is defined analogously as the smallest number of convex obstacles in any convex obstacle representation of G.

Alpert, Koch, and Laison [1] introduced the notion of the obstacle number of a graph. They showed that $obs(K^*_{a,b}) \leq 2$ for any $a \leq b$, where $K^*_{a,b}$ is the complete bipartite graph $K_{a,b}$ minus a matching of size a. Firman, Kindermann, Klawitter, Klemz, Klesen, and Wolff [5, 6] studied the graphs that can be drawn with a single obstacle in the outer face of the drawing, while Fulek, Saeedi, and Sarioz [7] gave bounds on the convex obstacle number of outerplanar and bipartite permutation graphs. Obviously, any n-vertex graph has obstacle number $O(n^2)$. Balko, Cibulka, and Valtr [3] improved this to $O(n \log n)$. On the other hand, Balko, Chaplick, Gupta, Hoffmann, Valtr, and Wolff [2] showed that there are n-vertex graphs whose obstacle number is $\Omega(n/\log\log n)$, improving previous lower bounds, e.g., [1, 4, 8, 9]. They also gave a purely existential proof that, when restricting obstacles to *convex* polygons, for some n-vertex graphs $\Omega(n)$ obstacles are needed.

In this paper, we study *planar convex obstacle representations*, which are convex obstacle representations where it is required that the line segments connecting corresponding points of adjacent vertices do not cross each other. The *planar convex obstacle number* $obs_{pc}(G)$ of a graph G is the smallest number of obstacles in any planar convex obstacle representation. A planar convex obstacle representations whose obstacles are all disks of radius one is a

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

planar unit-disk obstacle representation; the *planar unit-disk obstacle number* $obs_{pu}(G)$ is the minimum number of obstacle in any such representation. Note that $obs_{pc}(G) \leq obs_{pu}(G)$. We prove the following.

▶ **Theorem 1.1.** *For every $n \geq 7$, there exists an n-vertex tree T such that $obs_{pc}(T) \in \Omega(n)$. Also, for every n-vertex tree T, we have $obs_{pu}(T) \in O(n)$.*

We remark that planar obstacle representations trivially require one obstacle for each non-triangular face and that Fulek, Saeedi and Sariöz [7] exploit Ramsey's theorem to prove that for any $k > 0$ there exists a large enough tree requiring more than k convex obstacles. Proofs marked by a '⋆' are deferred to an upcoming full version.

2 Lower Bound on the Planar Convex Obstacle Number

In this section, we prove the first part of Theorem 1.1 by describing an infinite family of trees whose planar convex obstacle number has a linear lower bound. Let T_k be a *subdivided star* with k leaves, i.e., the tree obtained by subdividing exactly once every edge of a star with k leaves. Let r be the root of T_k. We refer to the children of r as the *parent vertices*. Consider any placement of r and any set of non-overlapping convex obstacles in the plane. We split the plane into multiple regions, defined by r and the tangents from r to the obstacles[1]. We can also assume, without loss of generality, that obstacles have non-zero area. Otherwise, if an obstacle is a line segment, then there is some small empty area around it, so we can slightly increase its width without changing the number and the type of the resulting regions.

Consider any obstacle O and one of its tangents t. If t does not intersect any other obstacle before O, then we draw it and extend it until it touches any other obstacle; we call those tangents *finite tangents*. If we can extend it to infinity without touching any other obstacle, then we call it an *infinite tangent*. Tangents that cross another obstacle before touching O can be disregarded. Note that this implies that obstacles that are completely hidden from the root by one or more other obstacles do not contribute any tangents.

Using the obstacles and finite and infinite tangents, we divide the plane into regions. Unbounded regions that contain r on their boundary are called *parent regions*. Note that they are bounded from both sides by infinite tangents. Each finite tangent touches exactly two obstacles: the obstacle that defines the tangent in a point p, and the obstacle that we extended the tangent to in a point q. If a tangent is defined by the side of a polygon, we use as point p the point on that side that is furthest away from r. We call the line segment $\overline{pq}$ a *degenerate parent region*. Lastly, we call the bounded regions that are fully visible to r *cone regions*; see Fig. 1.

▶ **Lemma 2.1.** *Let T_k be a subdivided star such that $k \geq 3$. Then $obs_{pc}(T_k) \geq \frac{n-1}{4}$.*

Proof. Consider any planar convex obstacle representation of T_k. We will show that this representation has at least $\frac{k}{2} = \frac{n-1}{4}$ convex obstacles. We say that a parent vertex u *uses* a (degenerate) parent region P if it is either placed inside P or P is the first region crossed by the edge uv, where v is the child of u. For now, we only consider those parent regions that have an angle $\leq 180°$ at r.

First, we show that every parent vertex uses some (degenerate) parent region. Since a parent vertex has to be visible to the root, it can either be placed inside a cone region or

[1] A tangent of a polygon is a line touching the polygon such that the polygon lies on one side of the line.

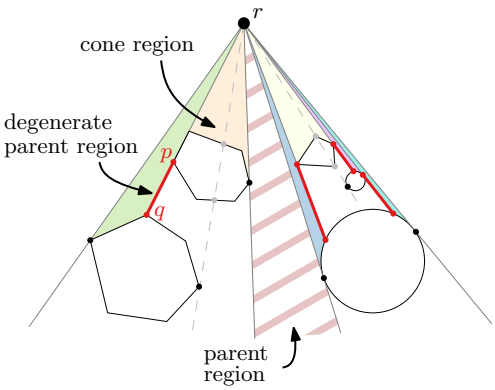


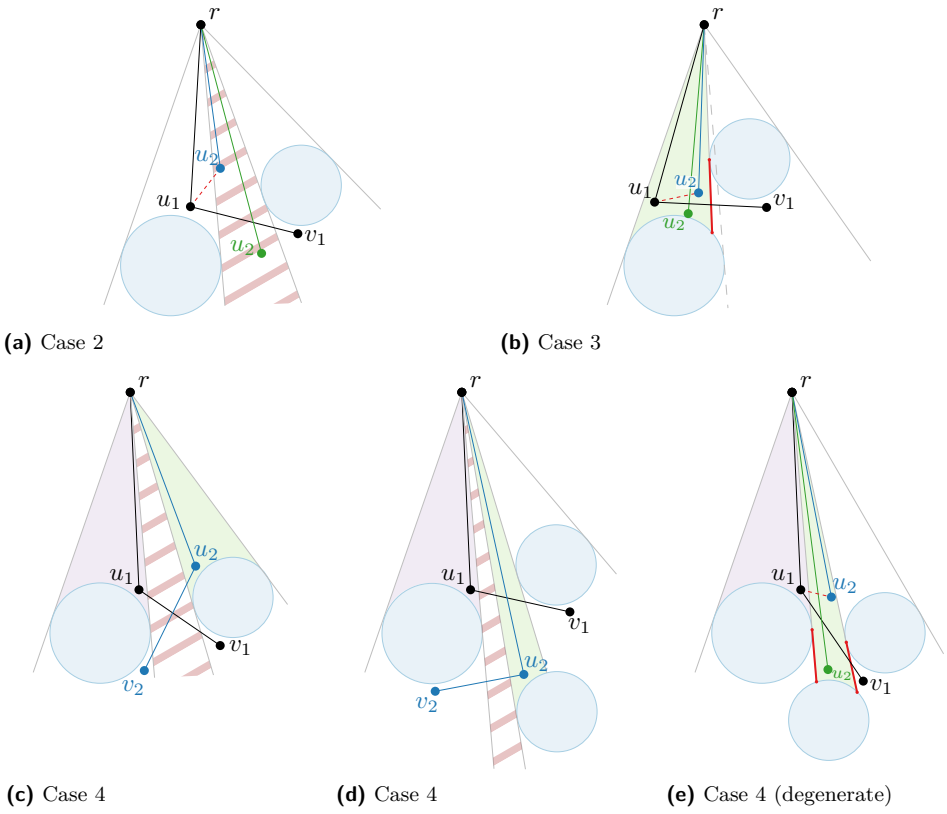
Figure 1 Illustration of the construction of the (degenerate) parent and cone regions.

inside a parent region. If a vertex is placed inside a parent region, by definition, it uses this region. If a parent vertex u is placed inside a cone region, then its child v has to be hidden from the root by some obstacle D and hence the edge uv crosses at least one (degenerate) parent region. The vertex u uses the first of those regions.

Second, we show that there is no pair u_1, u_2 of parent vertices that uses the same (degenerate) parent region. There are four different cases.
Case 1: Both u_1 and u_2 are inside the same parent region. This cannot happen, as then they would be visible to each other. Hence, they have to use different parent regions. **Case 2:** The vertex u_1 is inside a cone region, and u_2 is inside the parent region that u_1 uses. If u_2 lies above the edge u_1v_1, then u_2 is visible from u_1. If it lies below it, ru_2 crosses u_1v_1; see Fig. 2a. **Case 3:** Both u_1 and u_2 are in the same cone region and u_1v_1 and u_2v_2 cross the same (degenerate) parent region. Assume without loss of generality, that $x(u_1) < x(u_2) < x(v_1)$. Similar to the previous case, u_2 can either be placed above the edge u_1v_1, in which case u_2 is visible from u_1; or it can be placed below u_1v_1, in which case ru_2 crosses u_1v_1; see Fig. 2b. **Case 4:** The vertices u_1 and u_2 are in different cone regions and u_1v_1 and u_2v_2 cross the same parent region. Thus, either u_1v_1 and u_2v_2 cross (Fig. 2c), or ru_1 and u_2v_2 cross, or ru_2 and u_1v_1 cross (Fig. 2d). If they cross the same degenerate parent region, the argument is the same as in Case 2; see Fig. 2e.

There can be at most one parent region with an angle $> 180°$ at r. If we split such a region into two regions P_0 and P_1 with an angle $\leq 180°$ each, we can put at most one parent vertex into each of P_0 and P_1, otherwise they see each other. Further, if an edge u_1v_1 connecting a parent vertex u_1 to its child v_1 crosses such a region P_i with $i \in \{0, 1\}$, then either u_1 is inside P_i and any vertex placed inside $P_{|i-1|}$ is visible to u_1 or u_1 is placed in neither P_0 nor P_1 and the leaf v_1 is visible to r. Hence, a parent region with an angle $> 180°$ at r can be used by at most two parent vertices. Each obstacle creates at most two (degenerate) parent regions or one parent region with an angle $> 180°$. Thus, we can place two parent vertices per obstacle, which gives us the lower bound. ◀

It is easy to observe that a linear number of convex obstacles suffices for every planar graph G: Consider any planar straight-line drawing of G and place a convex obstacle between any pair of consecutive edges forming a convex angle inside a face. For those pairs of edges forming obtuse angles inside a face, three convex obstacles are always sufficient; see Fig. 3. For every vertex v, there is at most one obtuse angle inside a face, and there are $\deg(v)$ angles in total, so we need at most $\deg(v) + 2$ obstacles at v. Thus, in total, we use at most



(a) Case 2 **(b)** Case 3

(c) Case 4 **(d)** Case 4 **(e)** Case 4 (degenerate)

Figure 2 Illustration of the cases in the proof of Lemma 2.1.

$\sum_{v\in V}(\deg(v)+2) = \sum_{v\in V}\deg(v) + 2n \leq 2\cdot(3n-6)+2n = 8n-12$ obstacles.

▶ **Observation 2.2.** *Let G be a planar graph with n vertices. Then $obs_{pc}(G) \leq 8n-12$.*

In the next section, we reduce the gap between the upper bound of Observation 2.2 and the lower bound of Lemma 2.1 for the special case of trees. In our construction, we shall use unit disks as obstacles which, in some sense, strengthens our result.

3 Planar Unit-disk Obstacle Representations of Trees

In this section we prove the upper bound stated in Theorem 1.1. We start by describing an algorithm that computes a unit-disk obstacle representation of a rooted tree T whose internal vertices have at least two children. We denote by $\deg(v)$ the number of children of a vertex v of T. Let T be a tree; we denote by $int(T)$ the set of internal vertices of T, by Δ_T the *maximum degree* of a vertex of T, i.e., the maximum number of children of a vertex of T, and by h_T the *height* of T, i.e., the number of edges in the longest path from a leaf to the root. Let v be a vertex of T; we denote by $d(v)$ the *depth* of v, i.e., the number of edges in the path from v to the root, and by $sd(v)$ the quantity $h_T - d(v)$, which we call *slack depth* of v. The algorithm is recursive, and in order to state its invariants, we define a sequence of values δ_j recursively as: $\delta_0 = 1$ and $\delta_j = \frac{\delta_{j-1}}{2\Delta_T^j - 2}$ for $j > 0$. It is easy to see that $\delta_j < \frac{\delta_0}{2^j}$ for every $j > 0$ and hence $\sum_{j=0}^{i}\delta_j < 2$, for any $i \geq 0$. For each subtree T_v rooted

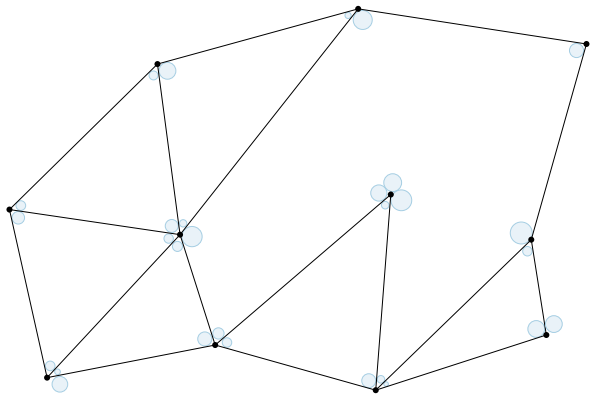


Figure 3 A planar convex obstacle representation of a planar graph.

at a vertex v with $sd(v) = i$ (for $i = 0, 1, \ldots, h_T$) the algorithm maintains these invariants.

- **I1.** The drawing Γ_v of T_v, excluding the obstacles, is completely contained in a rectangle R_v whose width is $W_i = 2(\Delta^{i+1} - 1)$ and whose height is $H_i = \sum_{j=0}^{i} \delta_j$; no obstacle intersects the top, left, or right side of R_v;
- **I2.** The root v of T_v is drawn as a point of the top side of R_v that is not a corner of R_v;
- **I3.** All non-root vertices of T_v have vertical distance at least δ_i from the top side of R_v;
- **I4.** The number of obstacles used in Γ_v is equal to $\sum_{u \in int(T_v)} (\deg(u) - 1)$.

In the base case $i = sd(v) = 0$, T_v consists only of the vertex v and Γ_v consists of a single point, representing v. Assume now that $i = sd(v) > 0$. Let $k = \deg(v)$ be the number of children of v; notice that $0 \leq k \leq \Delta$, but $k \neq 1$. If $k = 0$ then T_v consists only of vertex v, and again Γ_v consists of a single point. Otherwise, the children $u_1, u_2, \ldots, u_k$ of v have slack depth $i - 1$ and thus the subtrees T_{u_1}, T_{u_2}, ..., T_{u_k} recursively admit drawings that satisfy Invariant **I1**–**I4**. Let Γ_{u_1}, Γ_{u_2}, ...,Γ_{u_k} be such drawings. We construct the drawing of T_v by placing the drawings Γ_{u_1}, Γ_{u_2}, ..., Γ_{u_k} one after the other from left to right in such a way that the rectangles R_{u_1}, R_{u_2}, ...,R_{u_k} enclosing them have the top side at the same y-coordinate and the horizontal distance between R_{u_j} and $R_{u_{j+1}}$ ($j = 1, 2, \ldots, k-1$) is exactly two (so that a disk of radius one can be placed in the strip between two consecutive rectangles). See Fig. 4a for an illustration.

The root v of T_v is placed above the top sides of the rectangles R_{u_1}, R_{u_2}, ...,R_{u_k} at a vertical distance δ_i with a x-coordinate that is halfway between R_{u_1} and R_{u_2} (i.e., at a distance $w_{i-1} + 1$ from the left side of R_{u_1}). Denote by p_j the point on the left side of R_{u_j} at distance δ_{i-1} from the top side; further denote by q_j the point on the right side of R_{u_j} at distance δ_{i-1} from the top side. We place an obstacle O_j in the space between every pair of consecutive rectangles R_{u_j} and $R_{u_{j+1}}$ ($j = 1, 2, \ldots, k-1$). The vertical position of obstacle O_1 is chosen in such a way that O_1 intersects the segments $\overline{vp_1}$ and $\overline{vq_2}$ and it does not intersect the segments $\overline{vu_1}$ and $\overline{vu_2}$ (see Fig. 4b). Notice that such a placement is possible because the point u_1 (resp. u_2) is above the segments $\overline{vp_1}$ (resp. $\overline{vq_2}$) and therefore the segment $\overline{vu_1}$ (resp. $\overline{vu_2}$) is above the segment $\overline{vp_1}$ (resp. $\overline{vq_2}$), except for the common point v. The fact that u_1 (resp. u_2) is above $\overline{vp_1}$ (resp. $\overline{vq_2}$) is true by construction. Namely, we claim that $\overline{vp_1}$ (resp. $\overline{vq_2}$) passes through the top-right corner of R_{u_1} (resp. the top-left corner of R_{u_2}). We prove this claim for segment $\overline{vp_1}$, the proof for $\overline{vq_2}$ is symmetric. Denote by p_0 the top-right corner of R_1; segment $\overline{p_0p_1}$ has slope $\frac{\delta_{i-1}}{w_{i-1}}$ (see Fig. 4c); on the other

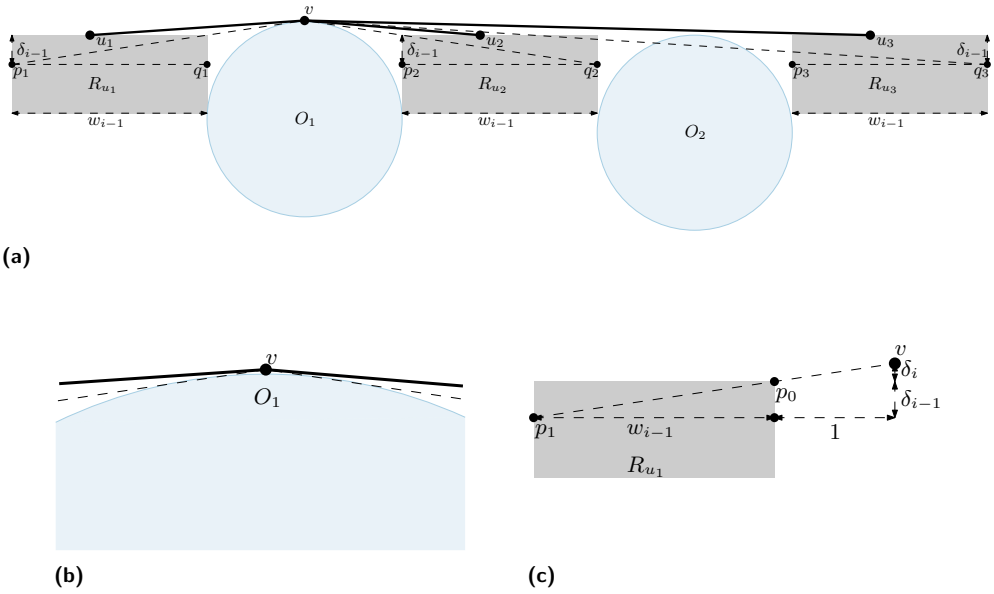


Figure 4 (a) Schematic construction of a unit-disk obstacle representation of a tree T_v rooted at a vertex v with $sd(v) = i$; (b) Detail of the placement of O_1; (c) The segment $\overline{vp_1}$ passes through the top-right corner p_0 of R_{u_1}.

hand, segment $\overline{vp_1}$ has slope $\frac{\delta_i+\delta_{i-1}}{w_{i-1}+1}$ which is equal to $\frac{\delta_{i-1}}{w_{i-1}}$ because $\delta_i = \frac{\delta_{i-1}}{2\Delta_T^i-2} = \frac{\delta_{i-1}}{w_{i-1}}$. The vertical position of O_j, for $j = 2, 3, \ldots, k-1$ is chosen so that O_j intersects the segment $\overline{vq_{j+1}}$ and it does not intersect the segment $\overline{vu_{j+1}}$. Also in this case, the placement is possible because, by construction and by Invariant **I2**, u_{j+1} is above the segment $\overline{vq_{j+1}}$.

▶ **Lemma 3.1** (⋆). *Let T be a rooted tree with n vertices such that every non-leaf vertex has at least two children. Then $obs_{pu}(T) \leq n-1-n'$, where n' is the number of non-leaf vertices.*

By adding a dummy child to every vertex with a single child, we can prove the following.

▶ **Lemma 3.2** (⋆). *Let T be a rooted tree with n vertices. Then $obs_{pu}(T) \leq n-1-n''$, where n'' is the number of vertices with at least two children.*

We observe that the number of obstacles stated in Lemma 3.2 is at most $n-1$, which gives the upper bound of Theorem 1.1. The value of $n-1$ is obtained for a tree whose internal vertices all have exactly one child, i.e., for a path. In fact, the technique described in the proof of Lemma 3.2 uses $n-1$ obstacles for paths. Notice, however, that for paths – and more in general for caterpillars – it is possible to construct a planar unit-disk obstacle representation with just one obstacle (see Fig. 5 for examples).

▶ **Lemma 3.3** (⋆). *Let T be a caterpillar. Then $obs_{pu}(T) = 1$.*

4 Open Problems

The lower bound of Lemma 2.1 only holds for crossing-free convex obstacle representation. It is not hard to construct a unit-disk obstacle representation with edge crossings of the tree used in the proof of Lemma 2.1 using only 4 unit disks (see Fig. 6). We conjecture however

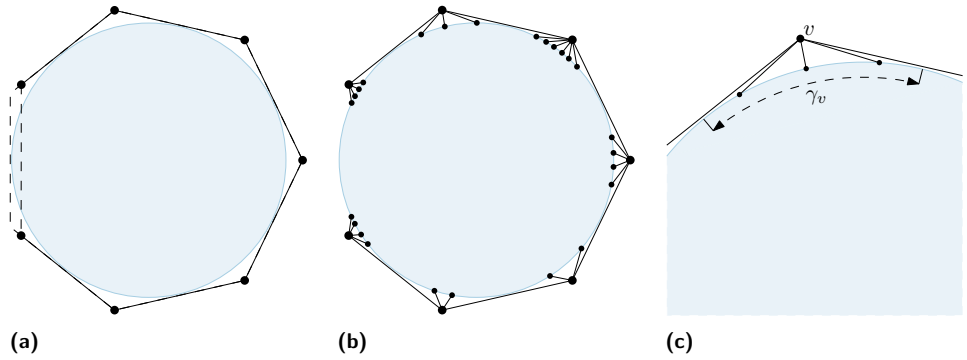


Figure 5 Unit-disk obstacle representation of: (a) a path; (b) a caterpillar. (c) Leaves addition.

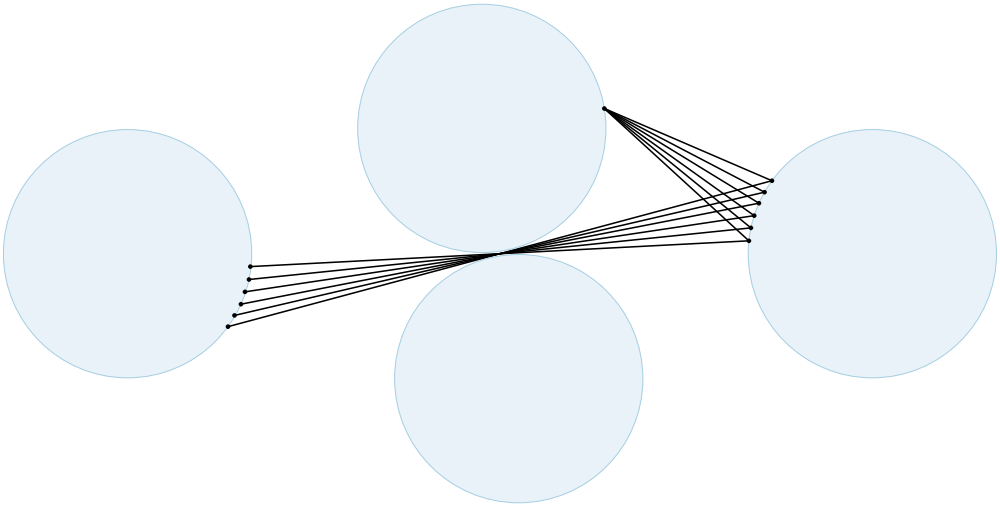


Figure 6 A non-planar unit-disk crossing representation with four obstacles of a subdivided star.

that a linear lower bound similar to that of Lemma 2.1 holds for trees even when we allow edge crossings and use convex regions of the same shape and size as obstacles.

References

1 Hannah Alpert, Christina Koch, and Joshua D. Laison. Obstacle numbers of graphs. *Discrete Comput. Geom.*, 44(1):223–244, 2010. doi:10.1007/s00454-009-9233-8.

2 Martin Balko, Steven Chaplick, Siddharth Gupta, Michael Hoffmann, Pavel Valtr, and Alexander Wolff. Bounding and computing obstacle numbers of graphs. In Shiri Chechik, Gonzalo Navarro, Eva Rotenberg, and Grzegorz Herman, editors, *Proc. 30th Europ. Symp. Algorithms (ESA'22)*, volume 244 of *LIPIcs*, pages 11:1–11:13. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPIcs.ESA.2022.11.

3 Martin Balko, Josef Cibulka, and Pavel Valtr. Drawing graphs using a small number of obstacles. *Discrete Comput. Geom.*, 59(1):143–164, 2018. doi:10.1007/s00454-017-9919-2.

4 Vida Dujmović and Pat Morin. On obstacle numbers. *Electr. J. Combin.*, 22(3):paper #P3.1, 7 pages, 2015. doi:10.37236/4373.

5 Oksana Firman, Philipp Kindermann, Jonathan Klawitter, Boris Klemz, Felix Klesen, and Alexander Wolff. Outside-obstacle representations with all vertices on the outer face. In Patrizio Angelini and Reinhard von Hanxleden, editors, *Proc. 30th Int. Symp. Graph Drawing and Network Vis. (GD'22)*, volume 13764 of *LNCS*, pages 432–440. Springer, 2022. doi:10.1007/978-3-031-22203-0_31.

6 Oksana Firman, Philipp Kindermann, Jonathan Klawitter, Boris Klemz, Felix Klesen, and Alexander Wolff. Outside-obstacle representations with all vertices on the outer face. *Comput. Geom. Topol.*, 4(1):2:1–2:21, 2025. doi:10.57717/CGT.V4I1.64.

7 Radoslav Fulek, Noushin Saeedi, and Deniz Sarioz. Convex obstacle numbers of outerplanar graphs and bipartite permutation graphs, 2011. arXiv:1104.4656.

8 Padmini Mukkamala, János Pach, and Dömötör Pálvölgyi. Lower bounds on the obstacle number of graphs. *Electr. J. Combin.*, 19(2):paper #P32, 8 pages, 2012. doi:10.37236/2380.

9 Padmini Mukkamala, János Pach, and Deniz Sarıöz. Graphs with large obstacle numbers. In Dimitrios M. Thilikos, editor, *Proc. Conf. Graph-Theoretic Concepts Comput. Sci. (WG'10)*, volume 6410 of *LNCS*, pages 292–303. Springer, 2010. doi:10.1007/978-3-642-16926-7_27.

General Visibility Graphs*

Franz-J. Brandenburg[1]

1 **University of Passau, 94032 Passau, Germany**
brandenb@informatik.uni-passau.de

Abstract

A *region* is a closed disc-homeomorph set of points in the plane or on the sphere. A *general visibility graph* consists of a finite set of pairwise disjoint regions as its vertex set, where two vertices are adjacent if and only if the corresponding regions can see one another. That is, there exist points on the boundaries of the two regions that can be connected by an unobstructed line of sight. String graphs and map graphs have been defined similarly, where edges are defined by intersection and touching, respectively. Due to compactness, every generalized visibility graph admits a representation as the visibility graph of polygons or polylines. Every triangulated planar graph and every hole-free map graph, defined by the touching of regions, is a generalized visibility graph, whereas n-cycle graphs for $n \geq 4$ and graphs that are obtained from biconnected graphs by subdividing each edge at least once are not general visibility graphs. Finally, we show that the recognition problem for generalized visibility graphs is decidable, where it is decidable in polynomial space if the size of each polygon (polyline) corresponding to a vertex is polynomally bounded in the size of the vertex set.

1 Introduction

Graphs are often defined by the following principle: There are one-to-one correspondences between the set of vertices V of a graph and a finite set of objects O and between the set of edges E and a binary relation ρ on the objects. The mapping $(V, E) \to (O, \rho)$ is called a *representation* of G. A common example is a drawing of a graph. The reverse mapping $(O, \rho) \to (V, E)$ is called a *realization* of G. Then G is called an (O, ρ)-*graph*. Examples are *string graphs* defined by (open) curves and intersection [18, 6], *map graphs* [5] defined by regions and touching, and *contact graphs* [10] defined by open curves, where the end of one curve touches the other curve in its interior. Visibility is an important topic in computational geometry. Two points p and q *see one another* if there is an unobstructed line of sight between the points. Two segments (curves) are *visible* to one another if they contain points that see one another. Several types of visibility graphs graphs have been introduced, notably, *point visibility graphs* [7, 3], in which the vertices correspond to points in the plane; *polygon visibility graphs* [7], whose vertices correspond to the points of a polygon and whose edges represent lines of sight entirely contained in the interior of the polygon; *visibility graphs of polygons* [7, 3], in which two points belonging to two polygons are connected by an edge if they see one another; and bar visibility graphs [15, 20, 21], where the vertices are horizontal lines (bars) and visibility is restricted to vertical lines of sight.

Our Contribution In this note, we introduce *general visibility graphs* whose vertices are regions or curves in the plane and whose edges are defined by their mutual visibility. We establish some graph theoretic properties, similar to studies of intersection, touching, and contact graphs. Properties on the representation of the graph classes, the complexity of the recognition problem, the containment of (triangulated) planar graphs and cliques, and the closure under some graph operations are summarized in Table 1.

* Supported by DFG grant Br835/20-1.

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

property	intersect	touch	contact	see
name	string graph [18]	map graph [5]	contact graph [10]	GVG
size of polyline	exponential [13, 17]	$O(n)$	$O(n)$	?
recognition	NP-C [11, 16]	?	NP-C	$\exists\mathbb{R}$-hard [3]
polynomial cases	interval graphs	4-map [1]	2-contact	bar visibility
planar graphs	✓	✓	no	no (Lem. 3.1)
triangulat. planar	✓	✓	✓	✓
n-cliques K_n	✓	✓	✓	✓
1-subdivision (K_5)	no [18, 6]	no	no	no (Lem. 3.1)
edge removal	no [11, 18]	no	no	no
vertex removal	✓	✓	✓	no

Table 1 realizable graph classes and selected properties

2 Basic Definitions

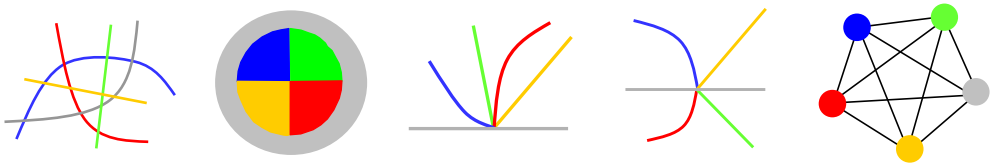


Figure 1 Representations of K_5 by the intersection of strings (curves), touching of regions (maps), one-sided and two-sided contact of strings, and visibility of points.

A *region* is a closed disc-homeomorph set of points in the plane or on the sphere. Its *boundary* is a closed Jordan curve. An open Jordan curve $c : [0, 1] \to \mathbb{R}^2$ is called a *string* [18, 6, 11]. The points $c(x), 0 < x < 1$ are called *interior points* and $c(0)$ and $c(1)$ are the *ends* of c. Two strings c and c' *intersect* at point p if $p = c(x) = c'(y)$ for some $0 \leq x, y \leq 1$ and the strings alternate in a cyclic order at p, see Fig. 1 (left). They *touch* at p if $p = c(x) = c'(y)$ and the pieces of each curve are consecutive at p. Two (or more) strings are in *contact* at p [9] if p is an end of c and an interior point of c'. Touching or contact at a point may be *one-sided* or *two-sided*, see Fig. 1. We assume that two strings intersect (touch) only finitely many times. Clearly, two regions intersect (touch) if their boundaries intersect (touch).

Let $C = c_1, \ldots, c_n$ be a finite set of non-intersecting curves in the plane, called a *configuration*. Each curve is the boundary of a *region* (closed curve) or a string (open curve). Two curves c_i and c_j *see one another* if there are points p on c_i and q on c_j such that the straight segment $L(p, q)$ is unobstructed, that is $(L(p, q) - \{p, q\}) \cap c_k(z) = \emptyset$ for every $k = 1, \ldots, n$ and every $z \in [0, 1]$. Then $L(p, q)$ is an *unobstructed line of sight* and p and q are *strongly visible*. Clearly, two lines of sight intersect at most once, whereas each line of sight can be intersected several times. A graph G is called a *general visibility graph*, *GVG* for short, if there is a one-to-one correspondence between its vertices and a finite set of non-intersecting curves, such that there is an edge if and only if the corresponding curves see one another. Thus, G is realized by a set of curves using visibility for the edges.

The set of real functions $c : [0, 1] \to \mathbb{R}^2$ is uncountable and most functions c do not have a finite description. Compactness and finite sets imply that curves can be approximated by

polylines and regions by polygons [12, 17]. Moreover, closed curves can be cut and strings (polyline) can be inflated if there is no point p in which other curves touch or have a contact from both sides. That is, two-sided touching or contact is excluded.

▶ **Lemma 2.1.** *A graph is an (O, ρ)-graph if it is an (O', ρ)-graph, where O and O' are finite sets of strings, regions, polylines, and polygons, respectively. The relation ρ denotes intersection, one-sided touching, one-sided contact, and visibility, respectively.*

In consequence, we can use regions, strings, polygons, and polylines interchangeably for the realization of graphs. Note that the step from regions (curves) to polygons (polylines) is not constructive.

▶ **Corollary 2.2.** *Every general visibility graph admits a realization by polylines (polygons).*

3 Properties of General Visibility Graphs

Obviously, every general visibility graph is connected. It may be a path, so that it is not necessarily biconnected. The *strong subdivision of a graph* is obtained by replacing each edge by a path of length at least two.

▶ **Lemma 3.1.** *The strong subdivision of any 2-connected graph is not a GVG.*

Proof. Assume that each edge of a graph G is subdivided exactly once; the generalization by more subdivisions follows by induction. For any two vertices u and v of G, if edge $e = \{u, v\}$ is subdivided by the subdivision point q_e, then q_e has degree two, while edge e is replaced by $\{u, q_e\}$ and $\{v, q_e\}$. In consequence, if s_u, s_v and s_e are the polylines representing u, v and q_e, respectively, then s_e only sees s_u and s_v, whereas s_u and s_v do not see one another. Now, s_u and s_v must be *obstructed* by the polylines of other vertices, and s_u and s_v obstruct s_e from any other polyline. Therefore, s_u and s_v are on opposite sides of s_e and s_e obstructs s_u and s_v. Otherwise, s_e sees the polylines of other vertices or s_u and s_v see one another. By induction, if $v_1, \ldots, v_k$ are the vertices on a cycle of G and $q_1, \ldots, q_k$ are the subdivision points on consecutive edges, then the polylines of $v_1, q_1, v_2, \ldots, q_{k-1}, v_k$ are ordered (e.g., from left to right), such that the polylines corresponding to v_i and v_{i+1} are obstructed by the polyline of q_i, and, similarly, the polyline of v_{i+1} obstructs those of q_i and q_{i+1}. Now, the cycle cannot be closed without a violation, since the polylines of v_1 and v_k cannot see another, as the polyline of any point q_i for $i = 1, \ldots, k$–1 obstructs their mutual visibility. ◀

By the same arguments we obtain:

▶ **Corollary 3.2.** *n-cycle graphs for $n \geq 4$ are not general visibility graphs.*

Clearly, every n-clique is a general visibility graph. Now n-cliques and n-cycle graphs restrict the comparison (inclusion relation) of general visibility graphs with many graph classes, e.g., the planar graphs.

A *map* M is a partition of the sphere into internally disjoint regions [4, 5], which are labeled as countries or holes. It is *hole-free* if all regions are labeled as countries. Two countries are *adjacent* if they meet in at least one point, that is, they touch. A graph is a (hole-free) *map graph* if there is a one-to-one correspondence between its vertices and the countries of a (hole-free) map, such that there is an edge if and only if the countries are adjacent. Alternatively, a map graph is the touching graph of polylines and a hole-free map graph is the extension of the dual of a plane graph with touching faces.

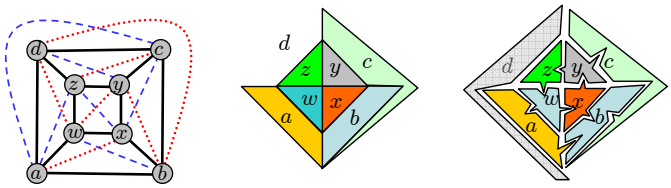


Figure 2 Illustration for the proof of Theorem 3.3. A (1-planar) graph (left), its representation by a hole-free map (middle) and as a GVG (right).

▶ **Theorem 3.3.** *Every hole-free map graph is (a) an intersection graph, (b) or a touching graph, (c) or a one-sided contact graph or (d) a general visibility graph of polylines (or polygons) whose size in total is $O(n)$.*

Proof. Every n-vertex map graph admits a representation by a map with regions of size $O(n)$. This follows from the representation of map graphs by so called witnesses of size $O(n)$, as shown by Chen et al. [5]. Now (a)-(c) are obvious. For (d), we shrink each polygon P by some $\epsilon > 0$ to a polygon P', such that a common segment on the boundary between two regions becomes a channel of two parallel lines and width 2ϵ, and each meeting point of $k \geq 3$ regions becomes an ϵ-ball, see Fig. 2. This preserves all adjacencies from touching to visibility. To avoid a further adjacency by a visibility through a channel, each pair of parallel edges e and e' has a dent, which prevents new adjacencies. ◀

▶ **Corollary 3.4.** *Every triangulated planar graph and every triangulated 1-planar graph (which admits a drawing in which each edge is crossed at most once) is a general visibility graph with polygons of size $O(n)$, and similarly for intersection and touching graphs.*

To the best of our knowledge, the hole-free map graphs are the largest class of graphs that is contained in the classes of intersection, touching, and general visibility graphs. We refer to [9] for relationships between planar and contact graphs.

Kratochvíl and Matousek [13] have shown there are string graphs with $O(n)$ vertices for which every string representation contains at least 2^n intersections. In contrast, general visibility graphs need only a quartic number of crossings, which is best possible due to the quartic crossing number of K_n.

▶ **Lemma 3.5.** *Every general visibility graph has a drawing with $O(n^4)$ many crossings.*

Proof. Construct a drawing of G as follows: Suppose P_v is the polygon corresponding to vertex v, and $L(u, v)$ is a line of sight between vertices p_{uv} and p_{vu} on the boundaries of P_u and P_v if there is an edge $\{u, v\}$. For the drawing, place a point p_v in the interior of P_v. For every edge $\{u, v\}$ of G, route a curve between p_u and p_v first from p_u to p_{uv} in the interior of P_u, then use $L_{u,v}$ from p_{uv} to q_{uv}, and finally route the curve from p_{vu} to p_v in the interior of P_v. The edges can be routed such that any two edges do not cross in the interior of polygons, since they are adjacent. For every edge of G, there is a segment $\overline{p_{uv}p_{vu}}$. Only segments may cross in the drawing, and two segments cross at most once. Hence, there are $\frac{m(m-1)}{2}$ crossings if G has m edges, and $m \leq \frac{n(n-1)}{2}$ for an n-vertex graph G. ◀

The recognition problem for general visibility graphs (of polylines) can be solved by using *visibility polygons of segments* [14, 19]. Let P_0 be an outer triangle and let $\mathcal{S} = \{s_1, \ldots, s_n\}$ be a set of disjoint segments $s_i = \overline{p_i p'_i}$ between points p_i and p'_i in the interior of P_0. The

visibility region $VR(s)$ of a segment $s \in \mathcal{S}$ consists of all points q in the interior of P_0 that are visible from any point on s. It is a polygon with holes, denoted by $Vis(s)$, that consists of the facial cycles of its outer boundary and of the holes. Each facial cycle is described by a cyclic sequence of segments $\sigma_1, \ldots, \sigma_k$. Each segment σ_i is determined by at most four points (a, b, c, d), such that two consecutive points are unobstructed by any other segment. The interior points b and c are two points on a segment or the endpoints of two segments of $\mathcal{S}$ which specify an unobstructed straight line L. Then the endpoints a and d are intersection points of L and another segment or a side of P_0 which is intersected first in either direction, so that a, b, c and d are collinear. The visibility region $Vis(s)$ has size $O(n^4)$ and can be computed in $O(n^4)$ time, which is even worst case optimal [14, 19].

▶ **Theorem 3.6.** *Recognizing general visibility graphs is decidable. It is decidable in polynomial space if the size of each polyline (polygon) is polynomally bounded in the order of G, and it is hard for the existential theory of the reals.*

Proof. The lower bound is due to the complexity of the point visibility problem [3]. The upper bound can be show as follows. (Sketch) Consider the general visibility graph of a set $S = \{s_1, \ldots, s_k\}$ of k segments. The GVG for polylines is obtained by merging the segments of each polyline corresponding to a vertex. We will use the following facts:

- The existential theory of the reals is decidable in polynomial space [2].
- The point visibility problem can be described in the existential theory of the reals [8].
- The computation of the visibility polygon $Vis(s)$ for every segment $s \in S$ [14, 19].

The task is to describe visibility polygons in the existential theory of the reals. Let T_s be the set of segments of the visibility polygon $Vis(s)$ for $s \in S$ described in terms pairs of real variables for the endpoints and let $T = \cup_s T_s$. Each $t \in T_s$ either corresponds a piece of a segment $s \in S$ or to a line of sight between the endpoints of two segments that determine a segment of $Vis(s)$ [14, 19]. For every segment in $S \cup T$ there exist four real variables for its endpoints, and there exists a real variable z describing an interior point of the segment or on its extension to a line. There are boolean variables $b_{s,s'}$ and $b_{s,t}$ describing that two segments $s, s' \in S$ do not intersect and that each segment of T is unobstructed by segments of S. For every $s \in S$ there is a set of boolean variables B_s corresponding to the other segments, such that $b_{s'} \in B_s$ shall be true if and only if a point of s' is in $Vis(s)$. Set b_s to be true. For $s', s'' \in \{s\} \cup B_s$ let $b_{s'} \Leftrightarrow b_{s''}$ if s' and s'' are in a quadruple specifying a segment $t \in T_s$. As visibility is strong, three collinear endvertices of segments in a quadruple are treated as in [8]. Now the description of each visibility polygon $Vis(s)$ can be expressed by a formula which says that each segment $t \in T_s$ is valid. Then there is an edge $\{u, v\}$ in a generalized visibility graph G if and only if $b_v \in B_u$ is true (where u and v mean the vertices and the corresponding segments, respectively). The number of real and boolean variables and the length of the formula describing the visibility regions of all segments is polynomial in the number of segments. ◀

Our major open questions are:
(1) Does every n-vertex general visibility graph have a realization by polygons of size $O(n)$?
(2) Generalize GVGs to three dimensions by replacing polygons by polyhedra that are grounded on the sphere. The sphere is taken as the only obstacle.

4 Acknowledgement

I wish to thank the reviewers for their constructive suggestions.

References

1 Franz J. Brandenburg. Characterizing and recognizing 4-map graphs. *Algorithmica*, 81(5):1818–1843, 2019. doi:10.1007/s00453-018-0510-x.

2 John F. Canny. Some algebraic and geometric computations in PSPACE. In Janos Simon, editor, *Proceedings of the 20th Annual ACM Symposium on Theory of Computing*, pages 460–467. ACM, 1988. doi:10.1145/62212.62257.

3 Jean Cardinal and Udo Hoffmann. Recognition and complexity of point visibility graphs. *Discrete & Computational Geometry*, 57(1):164–178, 2017. doi:10.1007/s00454-016-9831-1.

4 Zhi-Zhong Chen, Michelangelo Grigni, and Christos H. Papadimitriou. Planar map graphs. In *Proc. 30th STOC'98*, pages 514–523, 1998. doi:10.1145/276698.276865.

5 Zhi-Zhong Chen, Michelangelo Grigni, and Christos H. Papadimitriou. Map graphs. *J. ACM*, 49(2):127–138, 2002. doi:10.1145/506147.506148.

6 Gideon Ehrlich, Shimon Even, and Robert Endre Tarjan. Intersection graphs of curves in the plane. *J. Comb. Theory B*, 21(1):8–20, 1976. doi:10.1016/0095-8956(76)90022-8.

7 Subir Kumar Ghosh and Partha P. Goswami. Unsolved problems in visibility graphs of points, segments, and polygons. *ACM Comput. Surv.*, 46(2):22, 2013. doi:10.1145/2543581.2543589.

8 Subir Kumar Ghosh and Bodhayan Roy. Some results on point visibility graphs. *Theor. Comput. Sci.*, 575:17–32, 2015. doi:10.1016/j.tcs.2014.10.042.

9 Petr Hliněný. Contact graphs of curves. In Franz-Josef Brandenburg, editor, *Proceedings Symposium on Graph Drawing, GD '95*, volume 1027 of *Lecture Notes in Computer Science*, pages 312–323. Springer, 1995. doi:10.1007/BFB0021814.

10 Petr Hliněný. Classes and recognition of curve contact graphs. *J. Comb. Theory B*, 74(1):87–103, 1998. doi:10.1006/JCTB.1998.1846.

11 Jan Kratochvíl. String graphs I. the number of critical nonstring graphs is infinite. *J. Comb. Theory B*, 52(1):53–66, 1991. doi:10.1016/0095-8956(91)90090-7.

12 Jan Kratochvíl, M. Goljan, and P. Kucera. String graphs. *Academia Prague*, 1986.

13 Jan Kratochvíl and Jirí Matousek. String graphs requiring exponential representations. *J. Comb. Theory B*, 53(1):1–4, 1991. doi:10.1016/0095-8956(91)90050-T.

14 Joseph O'Rourke. *Art Gallery Theorem and Algorithms*. Oxford University Press, 1987.

15 Pierre Rosenstiehl and Robert Endre Tarjan. Rectilinear planar layouts and bipolar orientations of planar graphs. *Discrete & Computational Geometry*, 1:343–353, 1986. doi:10.1007/BF02187706.

16 Marcus Schaefer, Eric Sedgwick, and Daniel Štefankovič. Recognizing string graphs in NP. *J. Comput. Syst. Sci.*, 67(2):365–380, 2003. doi:10.1016/S0022-0000(03)00045-X.

17 Marcus Schaefer and Daniel Štefankovič. Decidability of string graphs. *J. Comput. Syst. Sci.*, 68(2):319–334, 2004. doi:10.1016/J.JCSS.2003.07.002.

18 F.W. Sinden. Topology of thin film RC circuits. *Bell System Tech. Journal*, pages 1639–1662, 1966.

19 Subhash Suri and Joseph O'Rourke. Worst-case optimal algorithms for constructing visibility polygons with holes. In Alok Aggarwal, editor, *Proceedings of the Second Annual ACM SIGACT/SIGGRAPH Symposium on Computational Geometry*, pages 14–23. ACM, 1986. doi:10.1145/10515.10517.

20 Roberto Tamassia and Ioannis G. Tollis. A unified approach a visibility representation of planar graphs. *Discret. Comput. Geom.*, 1:321–341, 1986. doi:10.1007/BF02187705.

21 S. Wismath. Characterizing bar line-of-sight graphs. In *Proc. 1st ACM Symp. Comput. Geom.*, pages 147–152. ACM Press, 1985. doi:10.1145/323233.323253.

Fréchet Distance in the Imbalanced Case*

Lotte Blank[1]

1 **University of Bonn, Germany**
lblank@uni-bonn.de

Abstract

We study the problem of approximating the discrete Fréchet distance between two 1-dimensional polygonal curves when the complexities of the two curves is imbalanced. We present a 2-approximation algorithm with almost optimal running time. Further, in the extended version we show that this approximation factor is optimal and this result cannot be extended to higher dimensions unless the Orthogonal Vectors Hypothesis fails.

Related Version *Full Version*: https://arxiv.org/abs/2602.09551

1 Introduction

The Fréchet distance is a distance metric for polygonal curves, where each curve is represented as a piecewise-linear interpolation between its vertices. There are different variants of this metric and the two most common variants are the discrete and the continuous Fréchet distance. Since Alt and Godau [2] introduced the Fréchet distance to the computational geometry community, its computational complexity has been extensively studied. Recently, Blank and Driemel [4] showed that the continuous Fréchet distance between 1-dimensional curves with n and m vertices ($n \geq m$) can be computed in $\widetilde{\mathcal{O}}(n + m^2)$ time[1]. No algorithm with the same running time can exist in higher dimensions unless SETH fails [5]. Further, Bringmann and Mulzer [6] showed that this running time cannot be achieved for the discrete Fréchet distance in 1D even with an approximation factor up to $1.4 - \varepsilon$ for any $\varepsilon > 0$[2]. In this paper, we show that the discrete Fréchet distance between 1-dimensional curves can be 2-approximated in $\widetilde{\mathcal{O}}(n + m^2)$ time. Moreover, in the full version of this paper we show that this approximation factor is optimal and that this result cannot be extended to curves in higher dimensions by reducing from the Orthogonal Vectors Hypothesis. Table 1 summarizes related work and the results of this (full version [3]) paper.

2 Preliminaries

For any two points $p_1, p_2 \in \mathbb{R}^d$, $\overline{p_1 p_2}$ denotes the directed line segment connecting p_1 with p_2. A polygonal curve $P = \langle P(1), \ldots, P(n) \rangle$ of complexity n is formed by ordered line segments $\overline{P(i)P(i+1)}$ of points $P(1), P(2), \ldots, P(n)$, where $P(i) \in \mathbb{R}^d$. We call the points $P(i)$ vertices of P. Further, $P[s,t]$ is the subcurve of P obtained from restricting the domain to the interval $[s,t]$. Let $P : [1,n] \to \mathbb{R}^d$ and $Q : [1,m] \to \mathbb{R}^d$ be two polygonal curves. A *discrete matching* M between P and Q is an ordered sequence that starts in $(1,1)$, ends in (n,m) and for any consecutive tuples (i,j), (i',j') in M it holds that

* This work was funded by 390685813 (Germany's Excellence Strategy – EXC-2047/1: Hausdorff Center for Mathematics); 416767905; and the Deutsche Forschungsgemeinschaft (DFG, German ResearchFoundation) – 459420781 (FOR AlgoForGe)

[1] $\widetilde{\mathcal{O}}(\cdot)$ hides polylogarithmic factors in n and m.

[2] In [6], this result is stated only for $n = m$, but the same construction works for the case $n \neq m$ as well.

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

Table 1 Upper and Lower Bounds for the Fréchet distance Computation

Upper Bounds						
	Discrete			Continuous		
1D	$\mathcal{O}(mn)$	exact	[11, 1]	$\widetilde{\mathcal{O}}(mn)$	exact	[2, 7, 9]
	$\widetilde{\mathcal{O}}(n+m^2)$	2-approx.	Cor.3.8	$\widetilde{\mathcal{O}}(n+m^2)$	exact	[4]
dD	$\mathcal{O}(mn)$	exact	[11, 1]	$\widetilde{\mathcal{O}}(mn)$	exact	[2, 7, 9]
	$\widetilde{\mathcal{O}}(n+m^2)$	$(3+\varepsilon)$-approx.	[3]	$\widetilde{\mathcal{O}}(n+m^2)$	$(3+\varepsilon)$-approx.	[3]
	$\widetilde{\mathcal{O}}(nm^{0.99})$	$(7+\varepsilon)$-approx.	[10]	$\widetilde{\mathcal{O}}(nm^{0.99})$	$(7+\varepsilon)$-approx.	[10]
Lower Bounds						
	Discrete			Continuous		
1D	$\nexists\ \mathcal{O}(m^{2-\delta})$	$(3-\varepsilon)$-approx.	[8]	$\nexists\ \mathcal{O}(m^{2-\delta})$	$(3-\varepsilon)$-approx.	[8]
	$\nexists\ \widetilde{\mathcal{O}}((mn)^{1-\delta})$	$(1.4-\varepsilon)$-approx.	[6]			
	$\nexists\ \widetilde{\mathcal{O}}((mn)^{1-\delta})$	$(2-\varepsilon)$-approx.	[3]			
2D		$\nexists\ \mathcal{O}((mn)^{1-\delta})$	(1.001)-approx.			[5]
		$\nexists\ \mathcal{O}((mn)^{1-\delta})$	$(1+\sqrt{2}-\varepsilon)$-approx. in L_2			[3]
		$\nexists\ \mathcal{O}((mn)^{1-\delta})$	$(3-\varepsilon)$-approx. in L_∞			[3]

$(i', j') \in \{(i+1, j), (i, j+1), (i+1, j+1)\}$. Then, the *discrete Fréchet distance* between P and Q is

$$d_{\mathrm{dF}}(P, Q) = \min_{\text{discrete matching } M} \max_{(i,j)\in M} \|P(i) - Q(j)\|.$$

The *free space matrix* M_δ is an $n \times m$ matrix, where $M_\delta[i, j] = 1$ if $|P(i) - Q(j)| \leq \delta$ and otherwise $M_\delta[i, j] = 0$. We say that (i, j) is *reachable* if $d_{\mathrm{dF}}(P[1, i], Q[1, j]) \leq \delta$.

3 2-Approximation for the discrete Fréchet distance in 1D

Given a value m and a 1-dimensional curve P of complexity $n \geq m$. We want to built a data structure that preprocesses P and can approximate the discrete Fréchet distance between P and any 1-dimensional query curve Q of complexity at most m.

Preprocessing. Define $\delta_m = \min\{d_{\mathrm{dF}}(P, Q) \mid Q \text{ has complexity } m\}$. Then, we want to compute a curve P', called *compressed simplification*, such that $d_{\mathrm{dF}}(P, P') \leq \delta_m/2$. The complexity of P' might be n, but it should be possible to store a compact description of P' in $\mathcal{O}(m)$ space (see Figure 1). To compute P', we first compute the *optimal m-simplification* P^* of P, which is a curve of complexity at most m minimizing the discrete Fréchet distance to P among all curves of complexity at most m. The curve P^* is computed via searching over the possible values δ for δ_m and then computing the *optimal δ-simplification*, which is a curve of minimum complexity among all curves that have Fréchet distance at most δ to P. We compute the optimal δ-simplification of P using Algorithm 1. It begins by finding the longest prefix $P[1, i_1]$ that is contained in a 2δ-interval, whose center becomes the first vertex of the δ-simplification. Then, it continues by finding the largest index i_2 such that $P[i_1 + 1, i_2]$ is contained in a 2δ-interval and again the center of this interval is added to the δ-simplification. This process is repeated until the final vertex of P is reached.

▶ **Lemma 3.1.** *Algorithm 1 computes an optimal δ-simplification P_δ of P using $\mathcal{O}(n)$ time.*

Algorithm 1: Optimal δ-simplification of P

1 $\ell \leftarrow 1$, $a \leftarrow P(1)$, $b \leftarrow P(1)$
2 **for** $i = 1$ *to* n **do**
3 **if** $P(i) \leq a + 2\delta$ *and* $b - 2\delta \leq P(i)$ **then**
4 $a \leftarrow \min\{a, P(i)\}$, $b \leftarrow \max\{b, P(i)\}$
5 **else**
6 $p_\ell \leftarrow \frac{a+b}{2}$, $a \leftarrow P(i)$, $b \leftarrow P(i)$, $i_\ell \leftarrow i - 1$, $\ell \leftarrow \ell + 1$
7 Return $P_\delta = \langle p_1, \ldots, p_\ell \rangle$, $i_0 = 0, i_1, \ldots, i_\ell = n$

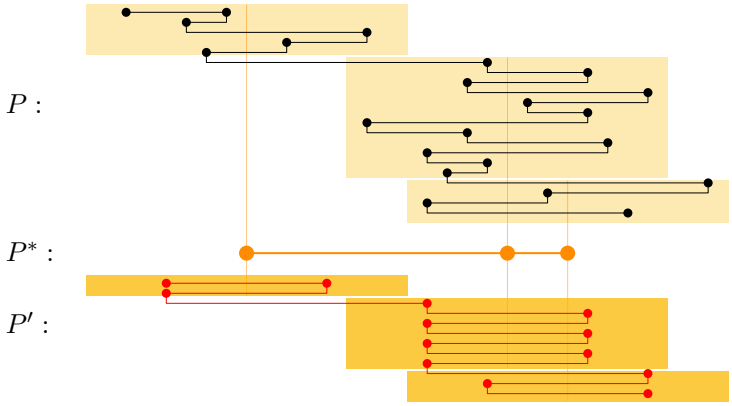


Figure 1 The curve P^* is an optimal 3-simplification for P and P' is the compressed curve. Here, $k_1 = 3$, $k_2 = 7$, $k_3 = 3$ and $z_1 = -1$, $z_2 = -1$, $z_3 = 1$.

Proof. First, observe that $d_{\mathrm{dF}}(P, P_\delta) \leq \delta$, since $((1,1),(2,1)\ldots,(i_1,1),(i_1+1,2),(i_1+2,2),\ldots,(i_2,2),(i_2+1,3),\ldots,(i_\ell,\ell))$ is a discrete matching realizing this distance. Assume that there exists a curve P' of complexity $k < \ell$ such that $d_{\mathrm{dF}}(P, P') \leq \delta$. Then, there exists a discrete matching M between P and P' realizing the discrete Fréchet distance and a tuple (x, y) in M with $x > i_y$. Let (x, y) be the first such tuple. Then, by line 3–4 of Algorithm 1 there exist $P(x_1) \leq P(x_2)$ such that (x_1, y) and (x_2, y) in M and $P(x_1)$, $P(x_2)$, and $P(x)$ are not contained in a 2δ interval. Hence, $d_{\mathrm{dF}}(P, P') > \delta$. ◀

If the complexity of the δ-simplification is greater than m, it holds that $\delta_m > \delta$. Otherwise, $\delta_m \leq \delta$. In contrast to higher dimensions, the optimal m-simplification for any m of a 1-dimensional curve P has distance $\delta_m \in \{|P(i) - P(j)|/2 \mid i, j \in \{1, \ldots, n\}\}$ to P, where the distance is only defined by two vertices of P. By Frederickson and Johnson [12], we can search over all possible values for δ_m using $\mathcal{O}(\log n)$ calls of Algorithm 1 and get the following lemma.

▶ **Lemma 3.2.** *We can compute an optimal m-simplification P^* of P together with an discrete matching that realizes the discrete Fréchet distance between P^* and P in $\mathcal{O}(n \log n)$ time.*

Let $P^* = \langle p_1, \ldots, p_\ell \rangle$ be an optimal m-simplification of P and $M = ((1,1),(2,1)\ldots,(i_1,1),(i_1+1,2),(i_1+2,2),\ldots,(i_2,2),(i_2+1,3),\ldots,(i_\ell,\ell))$ be a discrete matching between P and P^* realizing the distance. For every $(x, y) \in M$, define $r_x = p_y - \delta_m/2$ if $P(x) < p_y$ and $r_x = p_y + \delta_m/2$ otherwise. Then, it holds that $|r_x - P(x)| \leq \delta_m/2$. The compressed simplification P' is $\langle r_1, \ldots, r_n \rangle$ after deleting the vertices with identical predecessor, so that P' contains r_i only if $r_i \neq r_{i-1}$. For every j, we count the total number k_j that P'

contains $p_j \pm \delta_m/2$ and define $z_j \in \{-1, 1\}$ such that the first time P' contains $p_j + z_j\delta_m/2$ is before P' contains $p_j - z_j\delta_m/2$. Therefore, the values $p_1, \ldots, p_\ell$, $k_1, \ldots, k_\ell$, $z_1, \ldots, z_\ell$, and δ_m fully describe P'. The next lemma follows directly by construction and $\ell \leq m$.

▶ **Lemma 3.3.** *Given P and its optimal m-simplification P^*, we can compute in $\mathcal{O}(n)$ time a curve P' such that $d_{dF}(P, P') \leq \delta_m/2$ and P' can be stored using $\mathcal{O}(m)$ space.*

The Query Algorithm. In the following, we give a 2-approximation query algorithm for the discrete Fréchet distance between P and any query curve Q of complexity m. We use the precomputed compressed simplification P' and compute $d_{\mathrm{dF}}(P', Q)$ if it is at least $\frac{3}{2}\delta_m$.

▶ **Lemma 3.4.** *If $d_{dF}(P', Q) \leq \frac{3}{2}\delta_m$, then $d_{dF}(P, Q) \in [\delta_m, 2\delta_m]$. Otherwise, define δ such that $\frac{3}{2}\delta = d_{dF}(P', Q)$. Then, $d_{dF}(P, Q) \in [\delta, 2\delta)$.*

Proof. By definition of δ_m, it holds that $d_{\mathrm{dF}}(P, Q) \geq \delta_m$ for every curve Q of complexity m. Further, by Lemma 3.3 it holds that $d_{\mathrm{dF}}(P, P') \leq \delta_m/2$. Hence, if $d_{\mathrm{dF}}(P', Q) \leq \frac{3}{2}\delta_m$, then

$$\delta_m \leq d_{\mathrm{dF}}(P, Q) \leq d_{\mathrm{dF}}(P', Q) + d_{\mathrm{dF}}(P, P') \leq \frac{3}{2}\delta_m + \frac{1}{2}\delta_m \leq 2\delta_m.$$

Otherwise, it holds that $\frac{3}{2}\delta = d_{\mathrm{dF}}(P', Q) > \frac{3}{2}\delta_m$. Hence, $\delta > \delta_m$. Then,

$$\delta < \frac{3}{2}\delta - \frac{1}{2}\delta_m \leq d_{\mathrm{dF}}(P', Q) - d_{\mathrm{dF}}(P', P) \leq d_{\mathrm{dF}}(P, Q) \leq d_{\mathrm{dF}}(P', Q) + d_{\mathrm{dF}}(P', P) < 2\delta. \quad \blacktriangleleft$$

Using Lemma 3.4, we could simply compute $d_{\mathrm{dF}}(P', Q)$ to get a 2-approximation algorithm for $d_{\mathrm{dF}}(P, Q)$. However, computing $d_{\mathrm{dF}}(P', Q)$ with known discrete Fréchet distance algorithms takes roughly $\mathcal{O}(m \cdot \sum_{i=1}^{\ell} k_i)$ time and $\sum_{i=1}^{\ell} k_i$ could be n. In the following, we make use of the special structure of P' to get a faster algorithm that decides whether $d_{\mathrm{dF}}(P', Q) \leq \frac{3}{2}\delta$ for any value $\delta \geq \delta_m$.

We use the concept of the free space matrix $M_{(3/2)\delta}$, but do not compute all reachable entries explicitly. Define $s_i = \sum_{j=1}^{i-1} k_j + 1$ to be the index of the first vertex with value $p_i \pm \delta_m/2$ in P' and $s_{\ell+1} = |P'| + 1$. Algorithm 2 iterates over the sub-curves $P'[s_i, s_{i+1} - 1]$ of P' from $i = 1$ to ℓ and stores in the set F_{pre} the indices j of the vertices of Q such that $(s_i - 1, j)$ is reachable (see the hatched cells in Figure 2). Using F_{pre}, in iteration i we successively compute the set F_{new} of indices such that $(s_{i+1} - 1, j)$ is reachable and make use of the following property.

▶ Observation 1. For any point $q \in \mathbb{R}$, it holds that exactly one of the following is true:

- $|q - p_i| \leq \delta$ and $|q - P'(s_i + x)| \leq \frac{3}{2}\delta$ for all $x \in \{0, \ldots, k_i - 1\}$, or
- $|q - p_i| \in (\delta, 2\delta]$ and $|q - P'(s_i + x)| \leq \frac{3}{2}\delta$ for all even (resp. odd) $x \in \{0, \ldots, k_i - 1\}$ and $|q - P'(s_i + x)| > \frac{3}{2}\delta$ for all odd (resp. even) $x \in \{0, \ldots, k_i - 1\}$, or
- $|q - p_i| > 2\delta$.

By Observation 1, it holds that whenever $|Q(j) - p_i| \leq \delta$, there exists an number a such that $(s_i + x, j)$ with $x \in \{0, \ldots, k_i - 1\}$ is reachable if and only if $x \geq a$ and whenever $|Q(j) - p_i| > 2\delta$, then $(s_i + x, j)$ is not reachable for any integer $x \in \{0, \ldots, k_i - 1\}$ and we set $a = \infty$. When $|Q(j) - p_i| \in (\delta, 2\delta]$, things get slightly more complicated. However, we can still store a number a such that $(s_i + x, j)$ is reachable for all $x \in \{a, a + 2, \ldots\}$ with $x \leq k_i - 1$. Since there might be more reachable pairs $(s_i + x, j)$, we store a set A of size at most m and a number u which can be used to reconstruct all other reachable pairs $(s_i + x, j)$ with $x \in \{0, \ldots, a\}$. Then, let $B \subset \{0, \ldots, \min\{k_i - 1, a\}\}$ be the set such that for all

$x \leq \min\{k_i - 1, a\}$ it holds that $(s_i + x, j-1)$ is reachable if and only if $x \in B$. It holds that $B \neq \emptyset$ only if $|Q(j-1) - p_i| \in (\delta, 2\delta]$. In that case, define $z = 0$ (resp. $z = 1$) if $Q(j-1)$ and $Q(j)$ lie on the same side (resp. different sides) of p_i. Then for any $x + z > 0$, it holds that that $(s_i + x + z, j-1)$ is reachable if and only if $x \in B$. Using this property, we store a counter u that counts the sum of the values z until we reach again a vertex with $|Q(j') - p_i| \notin (\delta, 2\delta]$. If $x + z = 0$, we additionally check whether $(s_i - 1, j-1)$ or $(s_i - 1, j)$ is reachable by looking at the set F_{pre}. If one of them is reachable, then $(s_i, j-1) = (s_i + (u-u), j)$ is reachable and we add the current number u to the stored set A. In short, we intend to maintain the following invariant. During iteration i of the outer for-loop and at the end of iteration j of the inner for-loop, it holds that $(s_i + x, j)$ is reachable with $x \in \{0, \ldots, k_i - 1\}$ if and only if

a) $x \in \{u - z \mid z \in A\} \cup \{a\}$ or **b)** $x > a$ and $|Q(j) - P'(s_i + x)| \leq (2/3)\delta$.

See Figure 2 for an example. Here, during iteration $i = 3$ of the outer for-loop and at the end of iteration $j = 7$ (resp. $j = 8$) of the inner for-loop, it holds that $A = [0, 2]$, $u = 2$, and $a = \infty$ (resp. $A = [\,]$, $u = 0$, and $a = 0$).

Algorithm 2: Decision Algorithm for $d_{\text{dF}}(P', Q) \leq \frac{3}{2}\delta$

```
1  F_new ← {0}
2  for i = 1 to ℓ do
3      A ← [ ], a ← ∞, u ← 0, F_pre ← F_new, F_new ← [ ]
4      for j = 1 to m do
5          if |Q(j) − p_i| > 2δ then
6              A ← [ ], a ← ∞, u ← 0
7          if |Q(j−1) − Q(j)| > 2δ and |Q(j) − p_i|, |Q(j−1) − p_i| ∈ (δ, 2δ] then
8              a ← a + 1, u ← u + 1
9          if (j ∈ F_pre or j − 1 ∈ F_pre) and |P'(s_i) − Q(j)| ≤ (3/2)δ then
10             Append u to A
11         if |Q(j) − p_i| ≤ δ then
12             a ← min{a, u − min{{x | x ∈ A}} ∪ {∞}}
13             A ← [ ], u ← 0
14         Delete all x ∈ A with x ≤ u − k_i
15         if |P'(s_{i+1} − 1) − Q(j)| ≤ (3/2)δ and (a < k_i or u − k_i + 1 ∈ A) then
16             F_new ← j
```

▶ **Lemma 3.5.** *Assume that at the beginning of iteration i of the outer for-loop after setting $F_{pre} \leftarrow F_{new}$, it holds $(s_i - 1, j)$ is reachable if and only if $j \in F_{pre}$. Then, at the end of iteration i it holds that $(s_{i+1} - 1, j)$ is reachable if and only if $j \in F_{new}$.*

The proof of Lemma 3.5 is contained in the full version of this paper. It shows via induction that the invariant stated above is true. To prove the induction step, we distinguish between the three cases in Observation 1.

▶ **Lemma 3.6.** *For any value $\delta \geq \delta_m$, we can decide whether $d_{dF}(P', Q) \leq \frac{3}{2}\delta$ in $\mathcal{O}(m\ell)$ time.*

Proof. We run Algorithm 2 and output $d_{\text{dF}}(P', Q) \leq \frac{3}{2}\delta$ if and only if $m \in F_{\text{new}}$ at the end of Algorithm 2. The running time of Algorithm 2 is in $\mathcal{O}(m\ell)$, since the array A is sorted and

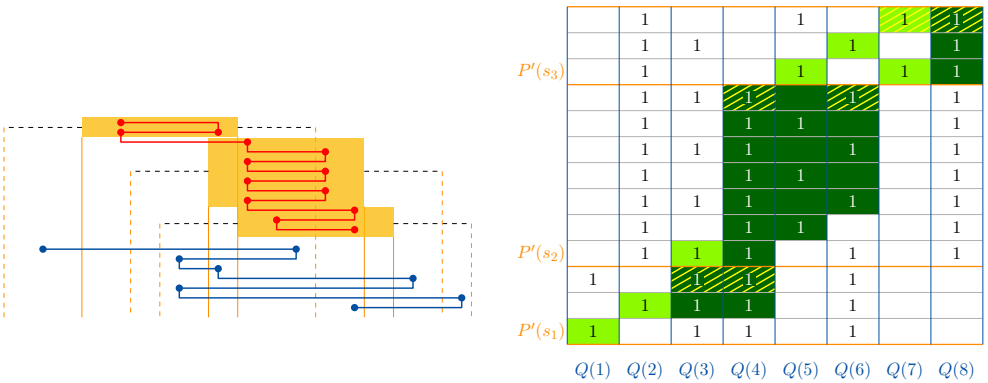


Figure 2 The compressed curve P' and the query curve Q are on the left. On the right is the free space matrix of P' and Q, where only 1-entries are added. The dark green cells visualize the parameter a and the light green cells the set $\{u - z \mid z \in A\}$. The hatched cells are the cells that have been in F_{pre} and F_{new}.

in every iteration of the inner for loop at most one point gets appended to A. By induction on the index i of s_i and Lemma 3.5, it holds that $(|P'|, m) = (s_{\ell+1} - 1, m)$ is reachable if and only if $d_{\text{dF}}(P', Q) \leq \frac{3}{2}\delta$. ◀

It remains to optimize the δ value. First, observe that the discrete Fréchet distance between P' and Q gets realized by a point to point distance between two vertices and that there are only $\mathcal{O}(m^2)$ such values. Hence, searching over all these distances that are at least $\frac{3}{2}\delta_m$ together with Lemma 3.4 and Lemma 3.6 results in the following theorem:

▶ **Theorem 3.7.** *For a given value m, there exists a data structure that preprocesses a 1-dimensional curve P of complexity n in $\mathcal{O}(n \log n)$ time using $\mathcal{O}(m)$ space such that for any 1-dimensional query curve Q of complexity at most m, we can approximate the discrete Fréchet distance between P and Q within a factor of 2 in $\mathcal{O}(m^2 \log m)$ time.*

The next corollary is a direct consequence of the theorem above.

▶ **Corollary 3.8.** *There is a 2-approximation algorithm to compute the discrete Fréchet distance between 1-dimensional curves of complexity n and $m \leq n$ in $\mathcal{O}(n \log n + m^2 \log m)$ time.*

In the full version we prove that there does not exist an algorithm using $\mathcal{O}(n \log n + m^2 \log m)$ time with a better approximation factor. Further, the lower bound in [8] shows the almost optimality of the running time.

References

1 Pankaj K. Agarwal, Rinat Ben Avraham, Haim Kaplan, and Micha Sharir. Computing the discrete Fréchet distance in subquadratic time. *SIAM Journal on Computing*, 43(2):429–449, 2014. doi:10.1137/130920526.

2 Helmut Alt and Michael Godau. Computing the Fréchet distance between two polygonal curves. *Int. J. Comput. Geom. Appl.*, 5:75–91, 1995. doi:10.1142/S0218195995000064.

3 Lotte Blank. Fréchet distance in the imbalanced case, 2026. URL: https://arxiv.org/abs/2602.09551, arXiv:2602.09551.

4 Lotte Blank and Anne Driemel. A faster algorithm for the Fréchet distance in 1d for the imbalanced case. In Timothy M. Chan, Johannes Fischer, John Iacono, and Grzegorz

Herman, editors, *32nd Annual European Symposium on Algorithms, ESA 2024, September 2-4, 2024, Royal Holloway, London, United Kingdom*, volume 308 of *LIPIcs*, pages 28:1–28:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2024. URL: https://doi.org/10.4230/LIPIcs.ESA.2024.28, doi:10.4230/LIPICS.ESA.2024.28.

5 Karl Bringmann. Why walking the dog takes time: Fréchet distance has no strongly subquadratic algorithms unless seth fails. In *2014 IEEE 55th Annual Symposium on Foundations of Computer Science*, pages 661–670. IEEE, 2014.

6 Karl Bringmann and Wolfgang Mulzer. Approximability of the discrete Fréchet distance. *Journal of Computational Geometry*, 7(2):46–76, 2016.

7 Kevin Buchin, Maike Buchin, Wouter Meulemans, and Wolfgang Mulzer. Four soviets walk the dog: Improved bounds for computing the Fréchet distance. *Discrete Comput. Geom.*, 58(1):180–216, jul 2017. doi:10.1007/s00454-017-9878-7.

8 Kevin Buchin, Tim Ophelders, and Bettina Speckmann. SETH says: Weak Fréchet distance is faster, but only if it is continuous and in one dimension. In Timothy M. Chan, editor, *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, San Diego, California, USA, January 6-9, 2019*, pages 2887–2901. SIAM, 2019. doi:10.1137/1.9781611975482.179.

9 Siu-Wing Cheng and Haoqiang Huang. Fréchet distance in subquadratic time. In Yossi Azar and Debmalya Panigrahi, editors, *Proceedings of the 2025 Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2025, New Orleans, LA, USA, January 12-15, 2025*, pages 5100–5113. SIAM, 2025. doi:10.1137/1.9781611978322.173.

10 Siu-Wing Cheng, Haoqiang Huang, and Shuo Zhang. Constant approximation of Fréchet distance in strongly subquadratic time. In Michal Koucký and Nikhil Bansal, editors, *Proceedings of the 57th Annual ACM Symposium on Theory of Computing, STOC 2025, Prague, Czechia, June 23-27, 2025*, pages 2329–2340. ACM, 2025. doi:10.1145/3717823.3718157.

11 Thomas Eiter and Heikki Mannila. Computing discrete Fréchet distance. 05 1994.

12 Greg N. Frederickson and Donald B. Johnson. Generalized selection and ranking: Sorted matrices. *SIAM J. Comput.*, 13(1):14–30, 1984. doi:10.1137/0213002.

Solving the Contiguous Art Gallery Problem using Few Starting Points

Sarita de Berg[1], Jacobus Conradi[2], Ivor van der Hoog[1], and Eva Rotenberg[1]

1 **IT University of Copenhagen, Denmark**
debe@itu.dk, ivva@itu.dk, erot@itu.dk
2 **University of Copenhagen, Denmark**
jaco@di.ku.dk

Abstract

Recently, a natural variant of the Art Gallery problem, known as the Contiguous Art Gallery problem, was proposed. Given a simple polygon P with n vertices, the goal is to partition its boundary ∂P into the smallest number of contiguous segments such that each segment is completely visible from some point in P. Unlike the classical Art Gallery problem, which is NP-hard, this variant is polynomial-time solvable as show by three independent works at SoCG '25. Interestingly, these results were obtained using entirely different approaches, yet all led to roughly the same high asymptotic complexity.

In this abstract, we show a simple but strong structural theorem that defines a linear-size set starting locations X, such that for at least one $u \in X$, a greedy algorithm that starts at u finds an optimal solution. Together with an $O(n \log n)$ time algorithm to compute X, this directly improves the running time of previous algorithms by a quadratic factor, resulting in an $O(n^3 \log n)$ algorithm for the Contiguous Art Gallery problem.

Related Version Full version: arXiv:2511.02960

Funding This work was supported by the the VILLUM Foundation grant (VIL37507) "Efficient Recomputations for Changeful Problems" and by the Carlsberg Foundation, grant CF24-1929.

1 Introduction

The art gallery problem has enjoyed much attention since its introduction in 1973 [5]. In its classical form, it asks, given a polygon P, for a smallest set of points (or guards) in P, such that every point in P is visible to some guard. A point p sees a point q, if the line segment $\overline{p\,q}$ is contained in P. This problem is known to be NP-hard [6], and $\exists\mathbb{R}$-complete [1]. Many variants have been considered. This ranges from restricting guards to the boundary of the polygon, which is also $\exists\mathbb{R}$-complete [1], to restricting guards to the vertices of P, which is NP-hard [6]. Alternatively, one can restrict the points that must be guarded. If asked to guard only the *boundary* ∂P of the polygon P, then the problem is also $\exists\mathbb{R}$-complete [9]. If the guards are also restricted to the boundary, it is at least NP-hard [6]. In short: most variants and restrictions of the art gallery problem are hard to solve.

Recently, at SoCG'25 [3], the Contiguous Art Gallery problem was introduced. In this version of the problem, a *contiguous guard* $(g, [u, v])$ is a point g in P that is assigned a contiguous visible portion $[u, v]$ of ∂P. The Contiguous Art Gallery problem is then: Given a simple polygon P with n vertices, compute a minimum-size set of contiguous guards G such that their corresponding visible portions cover the entire boundary of P.

Three simultaneous works show that this problem is polynomially solvable. Let P have n vertices and let the optimal solution have k guards. Merrild, Rysgaard, Schou, and Svenning [7] show an $O(kn^5 \log n)$-time algorithm via a greedy procedure. Starting from an

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

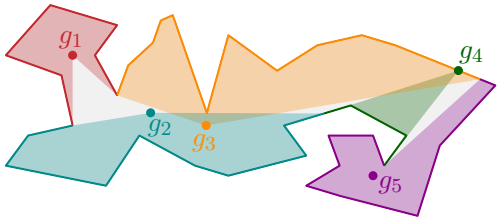


Figure 1 A simple polygon P and five contiguous guards that guard the entire boundary of P.

arbitrary point u on ∂P, they compute the maximal portion $[u, v]$ along ∂P such that there exists a contiguous guard $(g, [u, v])$, and recurse on v. This procedure returns a solution of at most $k + 1$ contiguous guards. They restart their procedure with a starting point that is not already in any of the previous solutions and after $O(kn^3)$ restarts they find solution of size k (using $O(kn^5 \log n)$ total time). Biniaz, Bose, Maheshwari, Mitchell, Odak, Polishchuk and Shermer [4] compute $O(n^4)$ possible guard locations. From this set, they apply an intricate combinatorial argument to combine these candidate contiguous guards into the optimal solution in $O(kn^5 \log n)$ time. Robson, Spalding-Jamieson, and Zheng [8] reduce the problem to an abstract problem that they call the Analytic Arc Cover Problem (AACP). Using results from real algebraic geometry and quantifier elimination, they show that AACP (and thereby Contiguous Art Gallery) is polynomially solvable.

We revisit the Contiguous Art Gallery problem. We give a simple but strong structural theorem that defines a set of $O(n)$ starting locations X, such that for at least one $u \in X$, a greedy algorithm that starts at u finds an optimal solution. Together with an $O(n \log n)$ time algorithm to compute X, this directly improves the running time of [7] by a factor $\Theta(kn^2)$ [3, Theorem 3], resulting in an $O(n^3 \log n)$ algorithm for the Contiguous Art Gallery problem. In the full version of this paper [2], we further improve the running time to $O(n \log n)$ and prove that this running time is optimal. Omitted proofs can be found in the full version [2].

2 Preliminaries

Our input is a simple polygon P with n edges. Formally, P is the closed region bounded by a simple closed curve consisting of n vertices defining n edges. A vertex of P is a *reflex vertex* if its interior angle exceeds π. We denote by ∂P the boundary of P and assume that ∂P is given in counter-clockwise order. Each edge is an ordered pair $\overline{u\,v}$ following this counter-clockwise ordering, i.e., it is directed from u to v. A point lies *strictly to the right* of $\overline{u\,v}$ if it lies in the open half-plane to the right of the directed line through u and v. A point *to the right* of $\overline{u\,v}$ may lie on this line. We make the standard general position assumptions that no three vertices are collinear.

Parameterizing ∂P. We assume that for any integer $i \in [1, n-1]$, the consecutive vertices v_i and v_{i+1} of P appear counter-clockwise. Consequently, the interior of P lies immediately left of $\overline{v_i\,v_{i+1}}$. For convenience, we define a continuous surjective function $[1, 2n+1) \to \partial P$ such that for every vertex v_j, both j and $n + j$ map to v_j. Thus, each point on ∂P can be represented by two real values: one in $[1, n+1)$ and one in $[n+1, 2n+1)$.

Using this parametrization, we define (open) chains as follows. A *chain* is a sequence of edges. For two points $u, v \in \partial P$, we denote by $[u, v]$ the chain obtained by traversing ∂P from u to v in counter-clockwise order. We denote by (u, v) the *open chain*, consisting of all

points $x \in [u, v]$ with $x \neq u$ and $x \neq v$; that is, $(u, v) := \{ x \in [u, v] \mid x \neq u,\ x \neq v \}$. Observe that the chain between any two points on ∂P can be described as $[u, v]$ with $u \in [1, n+1)$ and $v \in [u, 2n+1)$. We say that a guard $(g, [u, v])$ *dominates* another guard $(g', [u', v'])$ if $[u', v'] \subseteq [u, v]$. It *strictly dominates* $(g', [u', v'])$ if $[u', v'] \subsetneq [u, v]$.

Visibility core. For any chain $[u, v]$ between two points $u, v \in \partial P$, we define the *visibility core* $\mathcal{E}[u, v]$ as the set of points $p \in \mathbb{R}^2$ that lie left of all edges in $[u, v]$. We can use the visibility core to decide whether a single guard suffices.

▶ **Observation 2.1.** *If $(g, [u, v])$ is a guard, then g lies left of all edges of P intersecting the open chain (u, v). In particular, g lies in $\mathcal{E}[u, v]$. Moreover, there exists a guard that can see all of ∂P if and only if $\mathcal{E}[1, n+1]$ is non-empty.*

3 A linear number of starting guards

We first partition the set of all guards into *good* and *bad* guards. In Figure 1, g_2, g_3, and g_4 are good, while g_1 and g_5 are bad.

▶ **Definition 3.1.** Let u and v be points on ∂P with $u < v$. A contiguous guard $(g, [u, v])$ is *good* if $g \neq u$ and $g \neq v$ and the angle $\sphericalangle(u, g, v) \leq \pi$, and *bad* otherwise.

We will define a collection of guards called *good dominators*. We prove that all good guards are dominated by a good dominator, which will induce a set X of $O(n)$ points along ∂P such that there exists an optimal solution that includes a guard $(g, [x, v])$ for $x \in X$. To this end, we first make two observations:

▶ **Lemma 3.2.** *Let $(g, [u, v])$ be a guard where, for the fixed vertex u, v is maximal. Then either v is a reflex vertex of P or $\overline{vg}$ contains a reflex vertex of P in its interior.*

▶ **Observation 3.3.** *For any fixed $g \in P$ and indices i and j, with $i \leq j$, there exists at most one maximal chain $[u_{max}, v_{max}]$ visible from g that contains $[i, j]$.*

3.1 The good dominators

We show that there exists a set $\mathcal{D}$ of $O(n)$ guards, called the *reduced good dominators*, such that every good guard is dominated by one of them. We begin by defining two categories of *good dominators*, each of size $O(n^2)$.

▶ **Definition 3.4.** Consider an index pair (i, j) with $i \in [2, n+1]$ and $j \in [i, 2n]$. We define for (i, j) two types of guards that we call *good dominators*. Specifically, for each (i, j) either a segment-intersecting dominator (**A**) or an angle-maximizing dominator (**B**):

- **A (segment-intersecting dominator):** See Figure 2(a). If $\mathcal{E}[i-1, j+1]$ intersects the interior of $\overline{ij}$, we define the guard $(g, [u_{\max}, v_{\max}])$, where g is the last point of $\overline{ij}$ intersecting $\mathcal{E}[i-1, j+1]$, and $[u_{\max}, v_{\max}] \subseteq [i-1, j+1]$ is the maximal chain containing $[i, j]$ that is visible to g.
- **B (angle-maximizing dominator):** See Figure 2(b). If $\mathcal{E}[i-1, j+1]$ lies left of $\overline{ij}$, we define the guard $(g, [u_{\max}, v_{\max}])$, where g is i (or j) if $\mathcal{E}[i-1, j+1]$ intersects i (or j), or g is the point in $\mathcal{E}[i-1, j+1]$ that maximizes the angle $\sphericalangle(i, g, j)$, otherwise. We define $[u_{\max}, v_{\max}] \subseteq [i-1, j+1]$ as its maximal visible chain containing $[i, j]$.

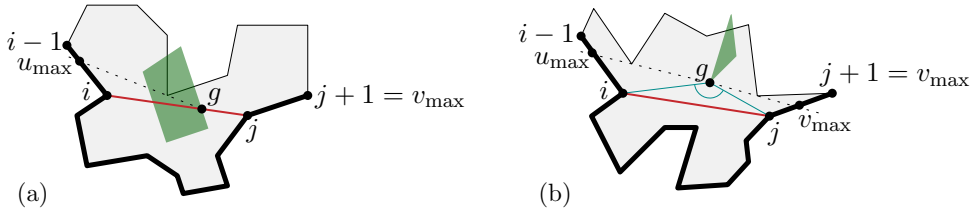


Figure 2 The visibility core $\mathcal{E}[i-1, j+1]$ in green and the (a) segment-intersecting dominator and the (b) angle-maximizing dominator $(g, [u_{\max}, v_{\max}])$ for (i, j).

For $i = j$, we define $(g, [u_{\max}, v_{\max}]) = (i, [i-1, i+1])$, which is a dominator of both type **A** and **B**. If there does not exist a type **A** and **B** guard for (i, j) that adheres to the stated conditions, then there is no good dominator for (i, j).

Note that by general position if $\mathcal{E}[i-1, j+1]$ is left of $\overline{i\,j}$, then it can intersect $\overline{i\,j}$ in at most a single point, so the angle-maximizing dominators are well-defined.

▶ **Definition 3.5.** The *reduced good dominators* $\mathcal{D}$ are the good dominators not strictly dominated by another good dominator.

If for the pair (i, j) the corresponding good dominator $(g, [u, v])$ is a reduced good dominator, then there is no pair (i', j') with $i' > i$ and $j' < j$ for which there is a guard in the reduced good dominators, as such a guard would be dominated by $(g, [u, v])$. This implies that the size of $\mathcal{D}$ is linear.

Next, we show that any good guard is dominated by a reduced good dominator.

▶ **Lemma 3.6.** *For every good guard $(g, [u, v])$ there exists a reduced good dominator $(g', [u', v']) \in \mathcal{D}$ that dominates it.*

Proof sketch. The guard $(g, [u, v])$ is good and thus $g \neq u, v$ and $\sphericalangle(v, g, u) \leq \pi$ by Definition 3.4. We prove that a good dominator (and thus a reduced good dominator) dominates $(g, [u, v])$. Let $u \in [i-1, i)$ and $v \in (j, j+1]$. If $i = j$ or $i - 1 = j$, then the dominator for (i, j), which is the guard $(i, [i-1, j+1])$, dominates $(g, [u, v])$. So, assume $i < j$.

We define by P_g the polygon formed by the segments $\overline{v\,g}$, $\overline{g\,u}$ and the edges in $[u, v]$. Since $(\overline{u\,g}, \overline{v\,g})$ forms a convex wedge, and P_g contains no points of ∂P in its interior, all points in $\mathcal{E}[u, v] \cap P_g$ see all of $[u, v]$. By general position, $\mathcal{E}[u, v]$ either is left of $\overline{i\,j}$ and intersects $\overline{i\,j}$ in at most one point, or contains at least one point strictly right of $\overline{i\,j}$. We define g^* as i (or j) if $\mathcal{E}[u, v]$ contains i (or j) and is left of $\overline{i\,j}$, or as the point in $\mathcal{E}[u, v]$ that maximizes the angle $\sphericalangle(i, g^*, j)$, otherwise.

Case 1: $\sphericalangle(i, g^*, j) > \pi$. The point g^* must lie strictly right of the segment $\overline{i\,j}$ and, in particular, cannot serve as an angle-maximising dominator. Instead, we construct the segment-intersecting dominator (**A**) and show that it dominates $(g, [u, v])$. Because $\sphericalangle(u, g, v) \leq \pi$, we also have $\sphericalangle(i, g, j) \leq \pi$, which, together with $\sphericalangle(i, g^*, j) > \pi$, implies that $\overline{i\,j}$ lies in P_g. In particular, g^* also lies in P_g, with $\overline{i\,j}$ separating g and g^*. By definition, both g and g^* lie in $\mathcal{E}[u, v]$. Since $\mathcal{E}[u, v]$ is convex, and $\overline{i\,j}$ separates g and g^*, $\mathcal{E}[u, v]$ must intersect $\overline{i\,j}$. Moreover, every edge of $\mathcal{E}[u, v]$ supports an edge in $[i-1, j+1]$, and as no three vertices of P are collinear, there must exist an intersection point in the interior of $\overline{i\,j}$.
Any point of intersection between $\overline{i\,j}$ and $\mathcal{E}[u, v]$ lies in both $\mathcal{E}[u, v]$ and P_g, and thus sees all of $[u, v]$. It follows that the last point of intersection g' between $\overline{i\,j}$ and $\mathcal{E}[u, v]$ sees

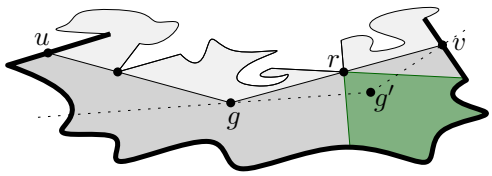


Figure 3 Illustration of Theorem 3.7 which shows a bad contiguous guard $(g, [u, v])$.

$[u, v]$ and so the segment-intersecting dominator $(g', [u_{\max}, v_{\max}])$ corresponding to (i, j) is well-defined and it dominates $(g, [u, v])$.

Case 2: $g^* = i$ or $g^* = j$ or $\triangleleft(i, g^*, j) \leq \pi$. In the full version, we show that the angle-maximizing dominator $(g^*, [u_{\max}, v_{\max}])$ (**B**) corresponding to (i, j) dominates $(g, [u, v])$. ◀

3.2 The set of starting guards

In this section we show that there is always an optimal solution to the Contiguous Art Gallery problem where either: the solution contains a good guard, or the solution contains a guard $(g, [u, v])$ where u is a vertex of P. Together with Lemma 3.6 this then implies that there exists an optimal solution where either: the solution contains a good *dominator*, or the solution contains a guard $(g, [u, v])$ with u a vertex of P.

▶ **Theorem 3.7.** *Let G be a minimal set of contiguous guards guarding ∂P with $|G| > 1$. There exists a set of contiguous guards G' with $|G| = |G'|$ where at least one $(g, [u, v]) \in G'$ is either in $\mathcal{D}$, or has u a vertex of P.*

Proof. We show how to transform G into such a set G' (see also Figure 3). We start by replacing every guard $(g, [u, v]) \in G$ by a guard $(g, [u', v'])$ such that $[u, v] \subseteq [u', v']$ and the interval $[u', v']$ is maximal to form G'. Next, we show that we can adapt G' such that either there is a good guard in G' or a guard whose visibility ends at a vertex of P. Lemma 3.6 implies that this good guard is dominated by a guard in $\mathcal{D}$, concluding the proof.

Suppose that G' contains only bad guards and let $(g, [u, v]) \in G'$ be such a bad guard. If (u, v) does not contain a vertex of P then $[u, v] \subseteq [i, i + 1]$ for some vertex i. We now replace $(g, [u, v])$ by $(i, [i, i + 1])$ and conclude the proof. Similarly, if v is a vertex of P then, because $|G| > 1$, there has to be at least one other guard $(g', [u', v']) \in G'$ such that $v \in [u', v']$. We replace this guard by $(g', [v, v'])$ so that the start of its chain is a vertex of P which concludes the proof. Lastly, note that it is not possible for g to be u (resp. v), as in that case, g sees all edges of P containing $g = u$ (resp. $g = v$), and in particular, strictly more than $[u, v]$, contradicting the maximality of $[u, v]$.

It follows that for all bad guards $(g, [u, v]) \in G'$, the open chain (u, v) contains a vertex of P, while neither u nor v is a vertex of P. By Lemma 3.2, the segment $\overline{v\,g}$ contains a reflex vertex r of P with $r \notin [u, v]$. Hence, there exists a guard $(g', [u', v'])$ such that $r \in [u', v']$. If $u' = r$ or $v' = r$, we conclude the proof as above. Otherwise, if $r \in (u', v')$, then g' lies left of both supporting lines of the edges incident to r. This implies that g' lies left of $\overline{v\,g}$. Since $\triangleleft(v, g, u) > \pi$, any point that sees r and lies left of $\overline{v\,g}$ must also lie within the polygon P_g defined by $[u, v]$, $\overline{v\,g}$, and $\overline{g\,u}$ (see Figure 3). Observe that the points in $[u', v'] \setminus (u, v)$ form a single closed chain; otherwise, $[u, v] \subseteq [u', v']$, contradicting the minimality of G'. Define $[a, b] = [u', v'] \setminus (u, v)$. If we replace $(g', [u', v'])$ by $(g', [a, b])$, the resulting set of guards still

covers ∂P. Crucially, $[a, b]$ lies left of $\overline{g' v}$ and left of either the supporting line of $\overline{u\, g'}$ or $\overline{g\, g'}$. As g' lies in P_g and $\sphericalangle(v, g, u) > \pi$, this implies that, unless $g' = a$ or $g' = b$, $\sphericalangle(a, g', b) \leq \pi$, and thus G' contains a good guard. Finally, if $g' = a$ (resp. $g' = b$), then g' is on ∂P and in particular, sees the entirety of the edge containing a (resp. b), hence we can extend $[a, b]$ to start (or end) in a vertex, which concludes the proof. ◀

Define by X a set of points along ∂P that contains all vertices of P and for all guards $(g, [u, v]) \in \mathcal{D}$ the point u. In the full version, we prove the following theorem, which states that the set X can be computed in $O(n \log n)$ time. Additionally, we further elaborate on the high-level algorithm to compute an optimal solution to the Contiguous Art Gallery problem by starting at a point in X, and greedily applying the 'next best' guard.

▶ **Theorem 3.8.** *For a simple polygon P of n vertices, we can compute a set X of $O(n)$ points in $O(n \log n)$ time such that there is a point $u \in X$ for which repeatedly applying the 'next best' guard will result in an optimal solution to the Contiguous Art Gallery problem.*

References

1 Mikkel Abrahamsen, Anna Adamaszek, and Tillmann Miltzow. The art gallery problem is ∃ℝ-complete. In *ACM SIGACT Symposium on Theory of Computing (STOC)*, STOC 2018, page 65–73, New York, NY, USA, 2018. Association for Computing Machinery. doi:10.1145/3188745.3188868.

2 Sarita de Berg, Jacobus Conradi, Ivor van der Hoog, and Eva Rotenberg. The contiguous art gallery problem is in Θ(n log n), 2026. arXiv:2511.02960.

3 Ahmad Biniaz, Anil Maheshwari, Magnus Christian Ring Merrild, Joseph S. B. Mitchell, Saeed Odak, Valentin Polishchuk, Eliot W. Robson, Casper Moldrup Rysgaard, Jens Kristian Refsgaard Schou, Thomas Shermer, Jack Spalding-Jamieson, Rolf Svenning, and Da Wei Zheng. Polynomial-Time Algorithms for Contiguous Art Gallery and Related Problems. In *Symposium on Computational Geometry (SoCG)*, volume 332 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 20:1–20:21, Dagstuhl, Germany, 2025. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.SoCG.2025.20.

4 Ahmad Biniaz, Anil Maheshwari, Joseph S. B. Mitchell, Saeed Odak, Valentin Polishchuk, and Thomas C. Shermer. Contiguous boundary guarding, 2024. arXiv:2412.15053.

5 Václav Chvátal. A combinatorial theorem in plane geometry. *Journal of Combinatorial Theory, Series B*, 18(1):39–41, 1975. doi:10.1016/0095-8956(75)90061-1.

6 D. Lee and A. Lin. Computational complexity of art gallery problems. *IEEE Transactions on Information Theory*, 32(2):276–282, 1986. doi:10.1109/TIT.1986.1057165.

7 Magnus Christian Ring Merrild, Casper Moldrup Rysgaard, Jens Kristian Refsgaard Schou, and Rolf Svenning. The contiguous art gallery problem is solvable in polynomial time, 2024. arXiv:2412.13938.

8 Eliot W. Robson, Jack Spalding-Jamieson, and Da Wei Zheng. The analytic arc cover problem and its applications to contiguous art gallery, polygon separation, and shape carving, 2024. arXiv:2412.15567.

9 Jack Stade. The Point-Boundary Art Gallery Problem Is ∃ℝ-Hard. In *Symposium on Computational Geometry (SoCG)*, volume 332 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 74:1–74:23, Dagstuhl, Germany, 2025. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. URL: https://drops.dagstuhl.de/entities/document/10.4230/LIPIcs.SoCG.2025.74, doi:10.4230/LIPIcs.SoCG.2025.74.

Garment numbers of bi-colored point sets in the plane

Oswin Aichholzer[1], Helena Bergold[2], Simon D. Fink[3], Maarten Löffler[4], Patrick Schnider[5,6], and Josef Tkadlec[7]

1 **Institute of Algorithms and Theory, TU Graz, Austria**
`oswin.aichholzer@tugraz.at`
2 **Freie Universität Berlin, Germany**
`helena.bergold@fu-berlin.de`
3 **Algorithms and Complexity Group, TU Wien, Austria**
`sfink@ac.tuwien.ac.at`
Vienna Science and Technology Fund (WWTF) grant [10.47379/ICT22029] and Austrian Science Fund (FWF) grant [10.55776/Y1329]
4 **Utrecht University, the Netherlands**
`m.loffler@uu.nl`
5 **Department of Mathematics and Computer Science, University of Basel, Switzerland**
`patrick.schnider@unibas.ch`
6 **Department of Computer Science, ETH Zürich, Switzerland**
`patrick.schnider@inf.ethz.ch`
7 **Computer Science Institute, Charles University, Prague, Czech Republic**
`josef.tkadlec@iuuk.mff.cuni.cz`
GAČR project 25-17221S and Charles Univ. projects UNCE 24/SCI/008, PRIMUS 24/SCI/012

Abstract

We consider colored variants of a class of geometric-combinatorial questions on k-gons and empty k-gons that have been started around 1935 by Erdős and Szekeres. In our setting we have n points in general position in the plane, each one colored either red or blue. A *structure* on k points is a geometric graph where the edges are spanned by (some of) these points and is called *monochromatic* if all k points have the same color. Already for $k = 4$ there exist interesting open problems. Most prominently, it is still open whether for any sufficiently large bichromatic set there always exists a *convex* empty, monochromatic quadrilateral. In order to shed more light on the underlying geometry we study the existence of five different monochromatic structures that all use exactly 4 points of a bichromatic point set. We provide several improved lower and upper bounds on the smallest n such that every bichromatic set of at least n points contains (some of) those monochromatic structures.

Related Version *Full Version*: `https://arxiv.org/abs/2603.05339` [1]

1 Introduction

Around 1935 Erdős and Szekeres [5] asked about the existence of a bound ES(k) such that any set of at least ES(k) points in general position in the plane contains a subset of k points that are the vertices of a convex k-gon. This problem is also known as the "Happy End Problem", see e.g. [3, 7] for further details. It started a whole area of geometric-combinatorial questions on k-gons and k-holes (k-gons which do not contain other points of the given set in their interior) in different settings and led to many beautiful results.

Variations when the points belong to different classes – that are usually described as *colors* – were introduced by Devillers, Hurtado, Károlyi, and Seara [4]. Here we say that a (not necessarily convex or empty) structure is *monochromatic* if all its points have the

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

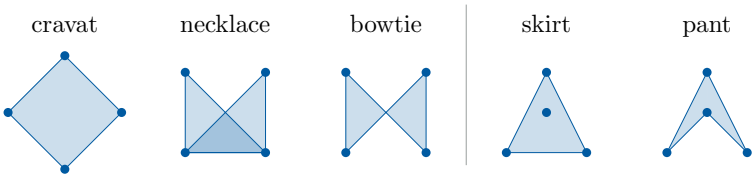


Figure 1 The five types of structures: cravat, necklace, bowtie, skirt, and pant.

same color. For sets of n points in the plane with two colors (called *bichromatic*), it has been shown that there are always at least $\lceil \frac{n}{4} \rceil - 2$ monochromatic triangles with pairwise disjoint interiors [4], and at least $\Omega(n^{4/3})$ [9] not-necessarily-disjoint empty monochromatic triangles. On the other hand, Devillers et al. show that for $k \geq 5$ and any n there are bichromatic sets of n points where no empty monochromatic convex k-gon exists (Theorem 3.4 in [4]). So somehow surprisingly the main remaining open problems in this setting are for sets of cardinality four. Moreover, it has been shown [2] that any sufficiently large (meaning with at least 2760 points) bichromatic set of points in the plane in general position determines an empty, monochromatic quadrilateral (which might not be convex). While this statement sounds rather innocent its proof is non-trivial, and at the time when this result was shown the corresponding lower bound was just 18 [2]. Moreover, the questions whether for a sufficiently large bichromatic set of points in the plane in general position there always exists a *convex* empty, monochromatic quadrilateral is still open. Here, the best known lower bound is 46 [8].

Motivated by these results and with the goal to shed more light on the underlying structures we study the existence of combinations of different monochromatic structures on subsets of 4 points. More precisely, we go beyond convex/non-convex polygons, and we introduce two new structures, the necklace and the bowtie; see Figure 1 and the next section for a definition. In total, we consider 5 different monochromatic structures that all use exactly 4 points. For combinations of these structures we study the following question: What is the smallest n such that every bichromatic set of n points contains at least one of those empty monochromatic structures?

2 Preliminaries

Throughout this paper, let P be a set of n points in the plane in general position (that is, no three points in a set are collinear). A *structure* is a (closed) subset of the plane that is induced by 4 different points from P. We consider five different structures, namely (see also Figure 1):

1. a *cravat*,[1] defined by 4 points in convex position, is the convex hull of these 4 points.
2. a *necklace*, defined by 4 points in convex position, is the union of two triangles that share exactly two points that are consecutive on the convex hull.
3. a *bowtie*, defined by 4 points in convex position, is a self-intersecting 4-gon with those 4 points as vertices.
4. a *skirt*, defined by 4 points in non-convex position, is the convex hull of these 4 points.
5. a *pant*,[2] defined by 4 points in non-convex position, is a simple 4-gon on these 4 points.

[1] As in French "à la cravate", commonly also called necktie.
[2] The singular "a pant" is actually a viable alternative to "pants", especially in the fashion industry, and combinatorially less confusing.

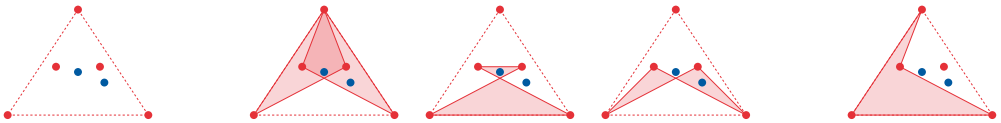


Figure 2 Left: A set of 7 points, 5 of them red and 2 blue. Middle: Each monochromatic skirt and bowtie contains a point of the other color, thus we have $\mathcal{G}(\text{skirt} \vee \text{bowtie}) > 7$. Right: There is an empty pant, so the point set does not imply anything about the setting pant $\vee$ bowtie.

Note that the first three structures have a convex hull of size 4, while the latter two have a convex hull of size 3. Also note that each set of 4 points in convex position, depending on how they are ordered, induces 1 cravat, 4 necklaces, and 2 bowties; while in non-convex position it induces 1 skirt and 3 pants.

For a bichromatic point set P, we only consider structures that are monochromatic and for which there is no further point in P of the same color within their boundary. A point of the other color *blocks* such a structure if it is within its boundary. For a given bichromatic point set (colored, say, red and blue) and type of structure, we say that the blue points *block* the red points if every red structure contains at least one blue point. A structure that does not contain a point of the other color (and is thereby not blocked) is called *empty*. For a given set S of structures, we say "r *red points require* b *blue points*" or shortly write $r \triangleright b$, if every set of r red points requires at least b blue points to block all red structures in S.

We study whether all large enough bichromatic point sets contain an empty monochromatic structure. Depending on the types of considered structure(s), we obtain different *settings*, see Figure 2 for an example. Formally, given a set S of structures, we define the *GuARanteed Monochromatic Empty Not-quite-order Type number* (shorthand: Garment number) of bichromatic point sets, to be the smallest integer n, such that every bichromatic point set of size n contains at least one empty monochromatic structure from S. (If no such integer exists, we set the garment number to $+\infty$.) Figure 2 illustrates that the Garment number of the set $S = \{\text{skirt, bowtie}\}$ is more than 7. We denote this by $\mathcal{G}(\text{skirt} \vee \text{bowtie}) > 7$.

3 Upper Bounds

We present upper bounds on Garment numbers in three different settings. They all include one structure with a convex hull of size 4, plus up to one structure with convex hull of size 3; see Table 1. The first two of our upper bounds follow a similar three-step approach:

1. By direct casework, we show that $r \triangleright b$ for certain pairs (r, b) of small integers.
2. Using an inductive argument, we show that if both $r-1 \triangleright b-1$ and $r \triangleright b$, then $r+1 \triangleright b+1$.
3. For large enough n, we locate a convex-like shape in P, and we find an empty structure there using $r+k \triangleright b+k$, for suitable $k \geq 1$.

In the rest of this section, we first present the inductive argument (Lemma 1). Then, we complete the proofs of the upper bounds for the settings {pant, necklace} (Theorem 4) and {pant, bowtie} (Theorem 6). Finally, we prove Theorem 8 for the setting {necklace}.

Statements with a $\star$ have their full proofs deferred to the full version [1].

▶ **Lemma 1** ($\star$). *Let* $S = \{\text{pant, bowtie}\}$ *or* $S = \{\text{pant, necklace}\}$. *For all positive integers red and blue, it holds that* $(r \triangleright b \ \wedge \ r-1 \triangleright b-1) \ \Rightarrow \ r+1 \triangleright b+1$.

	none	cravat	necklace	bowtie
none	∞	$\leq ?$ $>$**46**	$\leq$**1508** $>$**14**	$\leq$*1508* $>$*12*
skirt	∞	$\leq ?$ $>$~~*18*~~ $>$**35**	$\leq$*1508* $>$*12*	$\leq$*1508* $>$**12**
pant	∞	$\leq$**2760** $>$~~**18**~~ $>$**22**	$\leq$~~*2760*~~ $\leq$**21** $>$**12**	$\leq$~~*2760*~~ $\leq$**11** $>$**10**

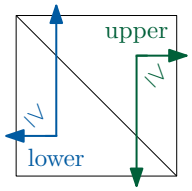
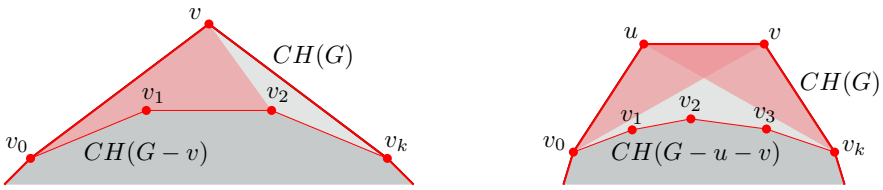


Table 1 The settings we consider in this paper, together with the upper and lower bounds on the number of points of a bichromatic point set in which an empty monochromatic structure always exists. Previously-known values are black, new results from this paper are orange. The results that are not a consequence of other results in this table are boldface and roman (i.e., non italic). Upper bounds are also upper bounds for settings below them or to their right, while lower bounds are also lower bounds for settings above them or to their left.



(a) Case 1, where the convex hull does not shrink. **(b)** Case 2, where the convex hull shrinks.

Figure 3 The two cases of Lemma 1.

Proof sketch. We distinguish two cases depending on whether the boundary of the convex hull of the red vertices contains a vertex whose removal yields a point set with at least that many red points on the convex hull (Case 1), or not (Case 2); see Figure 3. ◀

We now focus on the pant $\vee$ necklace setting. The proof of the first following lemma is a relatively straightforward casework (see Figure 4), while the second lemma can be shown by exhaustive enumeration on a computer.

▶ **Lemma 2** ($\star$). *For* $S = \{\text{pant}, \text{necklace}\}$ *we have* $4 \triangleright 2$ *and* $5 \triangleright 3$.

▶ **Lemma 3** ($\star$). *Any set of* 11 *points in general position contains either a convex 6-gon, or a convex 5-gon with at least 1 point inside of it.*

We note that Lemma 3 is minimal in the sense that there are sets of 10 points which contain neither a convex 6-gon nor a non-empty convex 5-gon (e.g. the double-chain with $5 + 5$ points). This now allows us to show our first result.

▶ **Theorem 4.** *We have* $\mathcal{G}(\text{pant} \vee \text{necklace}) \leq 21$.

Proof. Towards a contradiction, suppose there exists a bichromatic set of $n \geq 21$ points where each monochromatic pant and necklace are blocked. Consider the set G of the first

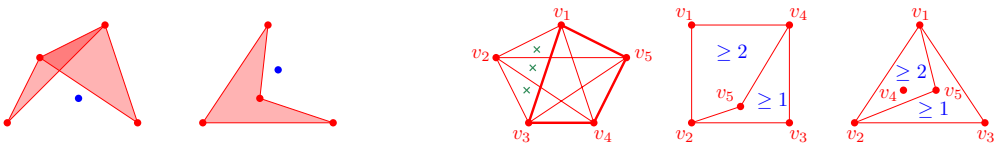


Figure 4 Left: We have $4 \triangleright 2$, regardless of whether the red quadrilateral is convex or not. Right: We have $5 \triangleright 3$, regardless of the size of the convex hull of the red points.

21 points in x-sorted order. Suppose G has r red and b blue points. W.l.o.g. assume $r > b$. By Lemma 2 we have $4 \triangleright 2$ and $5 \triangleright 3$, thus by Lemma 1 we have $12 \triangleright 10$. If $r \geq 12$ then $b \leq 21 - 12 = 9 < 10$, a contradiction. Thus, we can assume that $r = 11$ and $b = 10$.

By Lemma 3, the set of red points contains a subset I of size $|I| \geq 6$ that has at least $|\operatorname{CH}(I)| \geq 5$ points on the boundary of its convex hull. By Lemmas 1 and 2, I contains a set J of at least $|J| \geq |I| - 2 \geq 4$ blue points in the interior of its convex hull $\operatorname{CH}(I)$, and the set J in turn contains at least $|J| - 2 \geq |I| - 4$ red points in the interior of its convex hull $\operatorname{CH}(J)$. But this is a contradiction – since $|\operatorname{CH}(I)| \geq 5$, at most $|I| - 5$ red points may be in the interior of $\operatorname{CH}(I)$. ◀

To use induction for the upper bound in the pant $\vee$ bowtie setting, we again need a lemma for small point sets that uses some case work we defer to the full version [1].

▶ **Lemma 5** (⋆). *For $S = \{\text{pant}, \text{bowtie}\}$ we have $6 \triangleright 5$ and $7 \triangleright 6$.*

▶ **Theorem 6.** *We have $\mathcal{G}(\text{pant} \vee \text{bowtie}) \leq 12$.*

Proof. Given any bichromatic point set P of $n \geq 12$ points, consider the set P' of the 12 leftmost points. W.l.o.g. suppose $r \geq 6$ of them are red and $b = 12 - r$ are blue. If $r \geq 7$, then $b \leq 5$ thus there is an unblocked red pant or bowtie by Lemmas 1 and 5. So suppose $r = b = 6$. The boundary of $\operatorname{CH}(P')$ contains at least 3 vertices, w.l.o.g. at least 2 of them are blue. Removing them, we obtain a set of 6 red points and 4 blue points, thus again by Lemmas 1 and 5 there is a red unblocked pant or bowtie. ◀

Using exhaustive enumeration on a computer, one can actually show the stronger statement that all bichromatic point sets of size 11 contain an empty pant or bowtie. Together with Observation 9 from below, this establishes the following corollary.

▶ **Corollary 7** (⋆). *We have $\mathcal{G}(\text{pant} \vee \text{bowtie}) = 11$.*

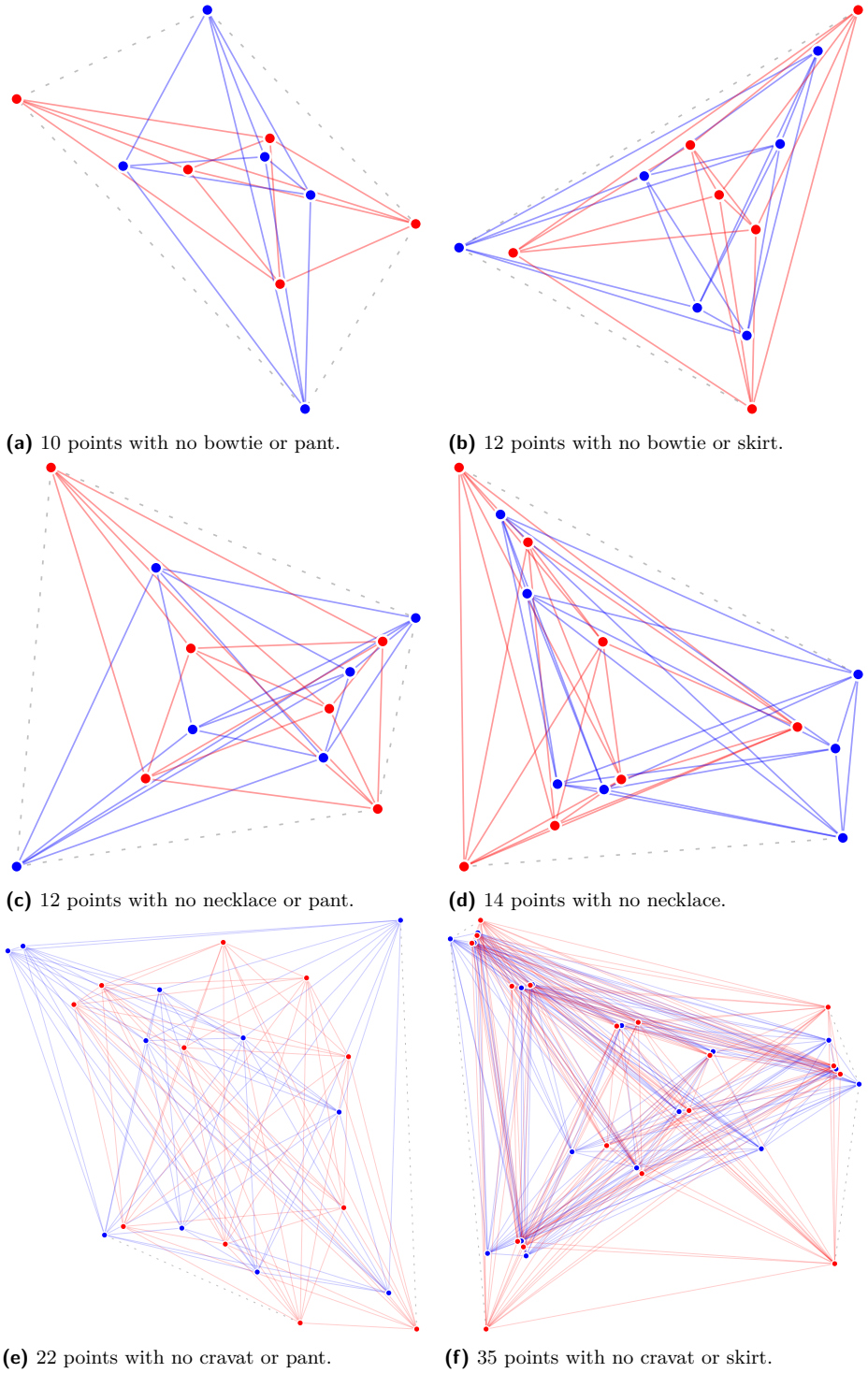
Our third upper bound also finds an empty structure within a large enough convex shape.

▶ **Theorem 8** (⋆). *We have $\mathcal{G}(\text{necklace}) \leq 1508$.*

Proof sketch. We first use the Erdős–Szekeres theorem [5] on convex k-gons to find a bicolored subset which consists of, say, at least five more blue than red points. Then we use a result on counting the number of interior disjoint convex blue 4-gons to show that there are not enough red points to double-block them all. Thus at least one blue convex 4-gon generates a blue necklace. ◀

4 Lower bounds

Our lower bounds follow from examples of point sets without empty monochromatic structures. More specifically, a lower bound of the form $\mathcal{G}(S) > x$ follows from a configuration of x points that contains no empty monochromatic S shown in Figure 5.



(a) 10 points with no bowtie or pant.

(b) 12 points with no bowtie or skirt.

(c) 12 points with no necklace or pant.

(d) 14 points with no necklace.

(e) 22 points with no cravat or pant.

(f) 35 points with no cravat or skirt.

Figure 5 Our lower bound constructions, all avoiding certain empty monochromatic structures.

▶ Observation 9. The following lower bounds hold:

(a) $\mathcal{G}(\text{bowtie} \vee \text{pant}) > 10$ (c) $\mathcal{G}(\text{necklace} \vee \text{pant}) > 12$ (e) $\mathcal{G}(\text{cravat} \vee \text{pant}) > 22$

(b) $\mathcal{G}(\text{bowtie} \vee \text{skirt}) > 12$ (d) $\mathcal{G}(\text{necklace}) > 14$ (f) $\mathcal{G}(\text{cravat} \vee \text{skirt}) > 35$

The exact point positions, together with code for verifying their correctness, can be found on GitHub [6]. The set of 22 points was found by students Severin Kann, Jürgen Pammer, and Matthias Platzer; the set of 35 points by Christian Payer and Karlheinz Wohlmuth.

5 Conclusion

We consider this work as a starting point of a more systematic exploration of blocking the various 4-point structures of bichromatic point sets. As seen in Table 1, many open questions remain. Next to the already-known open question of further closing the gap between lower and upper bounds for the pant and cravat setting, the main open part is finding any upper bounds for settings without pants. The main difference between settings with pants and those with only skirts, is that blocking a pant requires at least two points of the other color, while a skirt requires only one.

Acknowledgments

We would like to thank Philipp Kindermann as organizer of GG Week 2024 in Trier where this work was initiated.

References

1 Oswin Aichholzer, Helena Bergold, Simon D. Fink, Maarten Löffler, Patrick Schnider, and Josef Tkadlec. Garment numbers of bi-colored point sets in the plane, 2026. arXiv:2603.05339.

2 Oswin Aichholzer, Thomas Hackl, Clemens Huemer, Ferran Hurtado, and Birgit Vogtenhuber. Large Bichromatic Point Sets Admit Empty Monochromatic 4-Gons. *SIAM Journal on Discrete Mathematics*, 23(4):2147–2155, 2010. doi:10.1137/090767947.

3 Peter Brass, William O. J. Moser, and János Pach. *Research Problems in Discrete Geometry*. Springer, New York, 2005. doi:10.1007/0-387-29929-7.

4 Olivier Devillers, Ferran Hurtado, Gyula Károlyi, and Carlos Seara. Chromatic variants of the Erdős–Szekeres theorem on points in convex position. *Computational Geometry*, 26(3):193–208, 2003. doi:10.1016/S0925-7721(03)00013-0.

5 Paul Erdős and George Szekeres. A combinatorial problem in geometry. *Compositio Mathematica*, 2:463–470, 1935. URL: https://eudml.org/doc/88611.

6 Simon D. Fink. GuARanteed Monochromatic Empty Not-quite-order Type numbers of bichromatic point sets.
URL: https://github.com/N-Coder/garment-numbers-colored-point-sets, swh:1:dir:d2a56fdc1cb60a78113a85f7deada0430c5400b8.

7 Paul Hoffman. *The Man Who Loved Only Numbers: The Story of Paul Erdős and the Search for Mathematical Truth*. Hyperion Books, 1998.

8 Vitaliy Koshelev. On Erdös–Szekeres problem and related problems, 2009. arXiv:0910.2700.

9 János Pach and Géza Tóth. Monochromatic empty triangles in two-colored point sets. *Discrete Applied Mathematics*, 161(9):1259–1261, 2013. doi:10.1016/j.dam.2011.08.026.

Devil's Games and $\exists\mathbb{R}$ Continuous Games complete for the First-Order Theory of the Reals.

Lucas Meijer[1], Arnaud de Mesmay[2], Till Miltzow[3], Marcus Schaefer[4], and Jack Stade[5]

1 **Utrecht University**
2 **Gustave Eiffel University**
3 **Utrecht University**
4 **DePaul University**
5 **University of Copenhagen**

Abstract

We introduce the complexity class *Quantified Reals* abbreviated as $\exists\mathbb{R}$. Let FOTR be the set of true sentences in the first-order theory of the reals. A language L is in $\exists\mathbb{R}$, if there is a polynomial time reduction from L to FOTR. To the best of our knowledge, this is the first time this complexity class is studied. We show that $\exists\mathbb{R}$ can also be defined using real Turing machines or the real RAM. Interestingly, it is known that deciding FOTR requires at least exponential time unconditionally [Berman, *Theoretical Computer Science, 1980*].

We focus on so-called devil's games that have two defining properties: (1) Players alternate in taking turns and (2) each turn gives an infinite continuum of possible options. The two players are referred to as human and devil. Our paper presents four $\exists\mathbb{R}$-complete problems.

First, we show that FOTRINV is $\exists\mathbb{R}$-complete. FOTRINV has only inversion and addition constraints and all variables are contained in a bounded range. FOTRINV is a stepping stone for further reductions.

Second, we show that the Packing Game is $\exists\mathbb{R}$-complete. In the Packing Game we are given a container, as a polygonal domain and two sets of polygons, called pieces. One set of pieces for the human and one set for the devil. The human and the devil alternate by removing one of their pieces and placing it into the container. Both rotations and translations are allowed. The first player who cannot place a piece into the container loses. If all pieces can be placed the human wins.

Third, we show that the Planar Extension Game is $\exists\mathbb{R}$-complete. In this game, we are given a partially drawn plane graph and the human and the devil alternate by placing vertices and the corresponding edges in a straight-line manner. The vertices and edges to be placed are prescribed beforehand and known to both players. The first player who cannot place a vertex while respecting planarity of the drawing loses. If all the vertices can be placed the human wins.

Finally, we show that the Order Type Game is $\exists\mathbb{R}$-complete. In this game, we are given an order type together with a linear order of the abstract points. The human and the devil alternate in placing a point in the Euclidean plane $\mathbb{R}^2$ following the linear order. The first player who cannot place a point respecting the order type loses. If all the points can be placed the human wins.

Acknowledgments. We thank Jeremy Kirn for some helpful discussions to the machine model. T.M. and L.M. want to thank Netherlands Organisation for Scientific Research (NWO) Vidi grant VI.Vidi.213.150 for their generous support. All authors thank Dagstuhl to give them the wonderful opportunity to meet in a relaxed and enthusiastic environment at the seminar *Precision in Geometry* with seminar number 25372. This research project originated at the workshop. J.S. is supported by Independent Research Fund Denmark, grant 1054-00032B, and by the Carlsberg Foundation, grant CF24-1929.

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

1 Introduction

We introduce the complexity class *Quantified Reals* denoted by Q$\mathbb{R}$. This complexity class can be defined as all problems that are equivalent to deciding the First-Order Theory of the Reals under polynomial time reductions. We describe a framework to show Q$\mathbb{R}$-completeness of devil's games. Devil's games have two key properties.

- Players alternate in taking turns and
- each turn gives an infinite continuum of possible options.

1.1 Results and Definition

To state our results, we first need to define the first-order theory of the reals and its corresponding complexity class. The first-order theory of the reals, which is also called the theory of real closed fields in the literature, is the problem of deciding the truth of (first-order) logical sentences with any number of quantifiers in any order. The variables can take any real number, and the formula is built using standard arithmetic operations. For example, the following is such a first-order logical sentence

$$\forall x \exists y (x \cdot y = x + y) \vee (\exists z)(z > x + y) \rightarrow ((x \cdot y)^2 > z),$$

and deciding its truth is an instance of the first-order theory of the reals. An in-depth introduction to the theory of the reals can be found in standard references such as Basu, Pollack and Roy [2].

Next we define the class of the *quantified reals*, Q$\mathbb{R}$, of all problems reducible to FOTR under polynomial time reductions. Let us remark here that Q$\mathbb{R}$-complete problems take at least exponential time as was first shown by Fischer and Rabin [6]. To the best of our knowledge, no one has named the class before.

FOTRINV. In order to show hardness of any problem it is convenient to have an appropriate intermediate problem. Like, 3-SAT for the class NP. In our case, this intermediate problem is called FOTRINV and defined as follows.

▶ **Definition 1.1** (FOTRINV)**.** In the problem FOTRINV, we are given a quantified formula $\exists x_1 \forall x_2 \ldots Q_n x_n : \Phi(x_1, \ldots, x_n)$, where Φ consists of a conjunction between a set of equations of the form $x = 1, \quad x + y = z, \quad x \cdot y = 1$, for $x, y, z \in \{x_1, \ldots, x_n\}$. Each quantifier bounds exactly one variable and the quantifiers keep alternating between $\exists$ and $\forall$. The goal is to decide whether the system of equations has a solution when each existential variable is restricted to the range $[\frac{1}{2}, 2]$ and each universal variable is restricted to the range $[\frac{3}{4}, 1]$.

Indeed, if $x = 2$, then an addition constraint $x + y = z$ can never be satisfied. Similarly, if $z = \frac{1}{2}$, then $x + y = z$ can never be satisfied. We want to be able to involve universally-quantified variables in addition constraints without immediately causing the formula to become false, so we restrict the universal variables to a smaller range. We are now ready to state our first theorem.

▶ **Theorem 1.2.** *FOTRINV is Q$\mathbb{R}$-complete.*

We consider this our most technical result. The main difficulty lies in showing that we can bound (compactify to be more precise) the range of all the variables conveniently. The underlying idea is that if there is an assignment that makes a sentence true then we never had to pick very large values for any of the variables. If we have a bound on the values

of all variables involved, we can replace each variable by a scaled variable giving us small range bounds for FOTRINV. The issue is that there is no simple bound on the value of the variables. To see this consider the sentence

$$\forall x \neq 0 \exists y : x \cdot y = 1.$$

The sentence is clearly correct, but we cannot give an upper bound on y without having some lower bound on the distance of x to 0.

The main benefit of Theorem 1.2 is that it allows us to use FOTRINV as a starting point for future reductions. Note that ETRINV played an important role to show $\exists\mathbb{R}$-hardness of a host of problems, for example [1, 3, 7], so we expect that FOTRINV can play a similar role to show $Q\mathbb{R}$-hardness for other interesting problems.

Packing. In our next result we apply Theorem 1.2 to show that the Packing Game is $Q\mathbb{R}$-hard and **??** to show that Packing Game is in $Q\mathbb{R}$. Before we state the theorem formally, we define the Packing Game. See also Figure 1 for an illustration.

▶ **Definition 1.3** (Packing Game)**.** We are given two set of simple polygons H and D, called pieces, and a container C which is a polygonal domain. (A polygonal domain is a polygon that is allowed to have polygonal holes.) The human and the devil alternate in selecting one piece from their set and placing it in the container C without overlapping with any of the previously placed pieces. It is allowed that vertices and edges overlap. Just no two interior points should overlap. Furthermore, when placing the pieces it is allowed to rotate and translate the pieces. The player that cannot place one of their pieces anymore (without repetition) loses. If all pieces are successfully placed the human wins.

We are now ready to present our third theorem.

▶ **Theorem 1.4.** *The Packing Game is $Q\mathbb{R}$-complete.*

The theorem is established by a reduction from FOTRINV using mostly standard techniques from similar geometric reductions. One technical strength is that both players can choose from a set of pieces and are not predetermined which piece they use. We think the main value of Theorem 1.4 is that this is a very natural game. One can easily imagine this being an actual computer game.

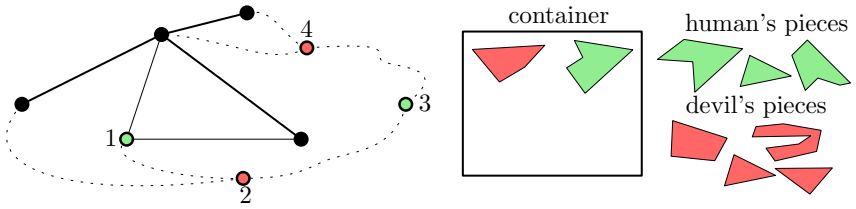


Figure 1 Left: The human and the devil place alternatingly vertices in the plane and the incident straight edges. The black vertices and thick edges were part of the input. The vertices $2, 3, 4$ still need to be placed. Right: We see a square container and a set of pieces both for the human and the devil, where both players have already placed some pieces.

Graph Drawing. Another very natural game is the Planar Extension Game. The idea is that two players have to draw a straight line planar graph together. And part of the graph is already drawn. Below is a more precise definition, see Figure 1 for an illustration.

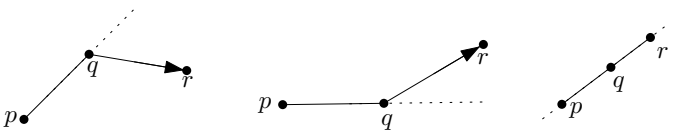


Figure 2 The orientation of a triple of points is either positive, negative, or zero.

▶ **Definition 1.5** (Planar Extension Game)**.** We are given a planar graph G, a planar straight-line drawing of a subgraph G' in the plane, and a linear ordering on the vertices of G not in G'. The human and the devil take turns in placing vertices according to the given linear order. All edges must be drawn as straight line segments without intersections. That is, no two edges are allowed to share a point in their interior, no two vertices are at the same location, and no vertex is allowed to lie in the interior of an edge. The game ends when the first player cannot place a vertex without creating an intersection. If the graph is drawn completely in the plane the human wins.

We are now ready to state our fourth theorem.

▶ **Theorem 1.6.** *The Planar Extension Game is Q$\mathbb{R}$-complete.*

The main strength of the theorem is that it is a very natural game to consider. Theorem 1.6 follows the ideas of the reduction to show $\exists\mathbb{R}$-hardness of the Graph in Polygon problem [9], combining it with new insights described below.

One highlight here is that we can show Q$\mathbb{R}$-completeness also for the variant that requires the drawing to be simple. That is, no edge has a vertex in its interior or overlaps with another edge. For the non-game variant only NP-hardness is known [12] and $\exists\mathbb{R}$-hardness is an intriguing open problem [9].

Notably, we develop a technique to enforce points to be placed collinearly with a line segment, despite the fact that we have no closed geometric constraints. Intuitively, we let the human and the devil check each other whether points are placed collinearly. We leverage the interaction between the human and the devil to prove that when a player does not place a vertex collinear to a specific intended line segment, the opponent can construct an obstruction that forces the player to create a crossing later and thus lose the game.

Order Types. Finally, we consider the Order Type Game. For completeness, we first define order types. An *(abstract) order type* $O = (E, \chi)$ of size n is a combinatorial abstraction of the possible locations of n points in $\mathbb{R}^2$. Here, E is a set of n elements called the ground set and χ is a map

$$\chi : \binom{E}{3} \to \{-1, 0, 1\}$$

called a chirotope satisfying a set of constraints derived from the geometric interpretation. We say that a point set $P \subset \mathbb{R}^2$ *represents* a given order type $O = (E, \chi)$ if there is a bijection φ between E and P such that the following holds. Let $p = \varphi(e)$, $q = \varphi(f)$, and $r = \varphi(g)$ then the orientation of p, q, r is given by $\chi(e, f, g)$. If $\chi(e, f, g) = 1$ then (p, q, r) form a triangle in counter-clockwise order. If $\chi(e, f, g) = -1$ then (p, q, r) form a triangle in clockwise order. And if $\chi(e, f, g) = 0$ then (p, q, r) are on a line. See Figure 2 for an illustration. For E' a subset of E, we denote by $O_{|E'}$ the restricted order type on E', where the chirotope is only defined on $\binom{E'}{3}$.

The Order Type Game is defined as follows.

▶ **Definition 1.7** (Order Type Game)**.** We are given an order type O and a linear ordering on the points of O. The human and the devil take turns in placing points in $\mathbb{R}^2$ according to the given order. At each step, the points must be placed so that the order type of the placed set of points $(p_1, \ldots, p_k)$ must represent $O_{|p_1,\ldots,p_k}$. The first player who cannot place a point loses, while the other player wins. If the entire order type is realized, the human wins.

The following theorem summarizes our findings on the game.

▶ **Theorem 1.8.** *Order Type Game is Q$\mathbb{R}$-complete.*

We find Theorem 1.8 interesting for several reasons.

- For $\exists\mathbb{R}$-completeness, order type realizability is one of the most important problem both conceptually and as a tool to show hardness for a host of other problems [10, 4, 5, 11, 8]. This leads us to hope that Order Type Game can also serve as a convenient starting point for reductions in the future.
- The techniques used to show Q$\mathbb{R}$-completeness are distinct to the techniques used in the previous two theorems.
- We strengthen the claim that devil's games are usually Q$\mathbb{R}$-complete.

References

1. Mikkel Abrahamsen, Anna Adamaszek, and Tillmann Miltzow. The art gallery problem is $\exists\mathbb{R}$-complete. *J. ACM*, 69(1):Art. 4, 70, 2022.
2. Saugata Basu, Richard Pollack, and Marie-Francoise Roy. *Algorithms in real algebraic geometry*. Springer Berlin, 2006.
3. Daniel Bertschinger, Christoph Hertrich, Paul Jungeblut, Tillmann Miltzow, and Simon Weber. Training fully connected neural networks is $\exists\mathbb{R}$-complete. *CoRR*, abs/2204.01368, 2022.
4. Jean Cardinal. The complexity of intersection graphs of lines in space and circle orders. *CoRR*, abs/2406.17504, 2024.
5. Jean Cardinal, Stefan Felsner, Tillmann Miltzow, Casey Tompkins, and Birgit Vogtenhuber. Intersection graphs of rays and grounded segments. *Journal of Graph Algorithms and Applications*, 22(2):273–295, 2018.
6. Michael J. Fischer and Michael O. Rabin. Super-exponential complexity of presburger arithmetic. In Bob F. Caviness and Jeremy R. Johnson, editors, *Quantifier Elimination and Cylindrical Algebraic Decomposition*, pages 122–135, Vienna, 1998. Springer Vienna.
7. Miika Hannula, Juha Kontinen, Jan Van den Bussche, and Jonni Virtema. Descriptive complexity of real computation and probabilistic independence logic. In *Proceedings of the 35th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS 2020)*, page 14. ACM, New York, 2020.
8. Michael Hoffmann, Tillmann Miltzow, Simon Weber, and Lasse Wulf. Recognition of Unit Segment and Polyline Graphs is $\exists\mathbb{R}$-complete. *Arxiv*, abs/2401.02172(2401.02172):1–18, 2024.
9. Anna Lubiw, Tillmann Miltzow, and Debajyoti Mondal. The complexity of drawing a graph in a polygonal region. In *Graph Drawing and Network Visualization: 26th International Symposium, GD 2018, Barcelona, Spain, September 26-28, 2018, Proceedings*, page 387–401, Berlin, Heidelberg, 2018. Springer-Verlag.
10. Jiri Matousek. Intersection graphs of segments and $\exists\mathbb{R}$. *arXiv preprint arXiv:1406.2636*, 2014.

11 Colin McDiarmid and Tobias Müller. The number of bits needed to represent a unit disk graph. In *Graph-theoretic concepts in computer science*, volume 6410 of *Lecture Notes in Comput. Sci.*, pages 315–323. Springer, Berlin, 2010.

12 Maurizio Patrignani. On extending a partial straight-line drawing. *Internat. J. Found. Comput. Sci.*, 17(5):1061–1069, 2006.

Signotopes Induce Unique Source Orientations on Grids*

Sandro M. Roch[1]

1 **Technische Universität Berlin**
roch@math.tu-berlin.de

Abstract

A unique sink orientation (USO) is an orientation of the edges of a polytope in which every face contains a unique sink. For a product of simplices $\Delta_{m-1} \times \Delta_{n-1}$, Felsner, Gärtner and Tschirschnitz (2005) characterize USOs which are induced by linear functions as the USOs on a $(m \times n)$-grid that correspond to a *two-colored arrangement of lines*. We generalize some of their results to products $\Delta^1 \times \cdots \times \Delta^r$ of r simplices, USOs on r-dimensional grids and $(r+1)$-*signotopes*.

1 Introduction

In the typical setting of linear programming, one has given a polytope $P \subset \mathbb{R}^n$ and a linear objective function $f : \mathbb{R}^n \to \mathbb{R}$. Assuming that f is sufficiently generic, f induces an orientation $v \to w$ of any edge $\{v, w\}$ of P s.t. $f(v) < f(w)$. This is an acyclic orientation of the skeleton of P with the property that on any face $F \subseteq P$ it has a unique source and a unique sink. A *source* resp. *sink* on F is a vertex on which all edges of F are oriented outgoing resp. incoming. The unique source and the unique sink of f correspond to the minimum and maximum of f on F.

A (not necessarily linear) function f on the vertices of P that on each face induces a unique source and a unique sink is called an *abstract objective function* by Kalai [6]. If f is a linear function, Holt and Klee [5] showed an even stronger property: On any face F, there exist $\dim F$ many internally disjoint dipaths from the source to the sink. This property is called the *Holt-Klee condition*.

In [1], Felsner, Gärtner and Tschirschnitz treat the special case in which P is a simple d-polytope with $d+2$ facets. Recall that two polytopes are *combinatorially equivalent* if their face lattices are isomorphic. A simple d-polytope with $d+2$ facets is combinatorially equivalent to the Cartesian product $\Delta_{m-1} \times \Delta_{n-1}$ of two simplices, with $m + n = d + 2$; see the example in Figure 1. Every face of $\Delta_{m-1} \times \Delta_{n-1}$ is the Cartesian product of a face of Δ_{m-1} and a face of Δ_{n-1}. Hence, if V and W denote the sets of vertices of Δ_{m-1} and Δ_{n-1}, then the $m \cdot n$ vertices of $\Delta_{m-1} \times \Delta_{n-1}$ can be identified with $V \times W$, and for every $V' \subseteq V$, $W' \subseteq W$, the vertices $V' \times W'$ form a face.

It is because of this simple fact that an orientation of a simple d-polytope with $d+2$ facets can be easily described as an orientation of a two-dimensional grid. For later purposes, we directly define a *grid* G *of size* $(n_1, \cdots, n_r)$ as the graph on the r-tuples $[n_1] \times \cdots \times [n_r]$, $n_i \geq 1$, where two of them form an edge if they differ in exactly one coordinate. Note that in the literature, this is often called a *rook's graph* instead of a grid. The *dimension* $\dim G$ of a grid is the number of i for which $n_i > 1$.

For subsets $V_i \subseteq [n_i]$, $V_i \neq \emptyset$, the graph induced on $V_1 \times \cdots \times V_r$ is called a *subgrid*. Any face $F \neq \emptyset$ of $\Delta_{m-1} \times \Delta_{n-1}$ corresponds uniquely to a subgrid $S = S_1 \times S_2 \subseteq [m] \times [n]$.

* **Acknowledgements:** Special thanks to Michaela Krüger for the stimulating discussions on this topic.

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

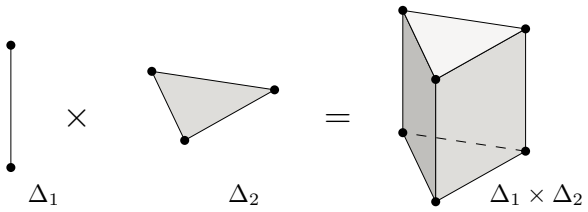


Figure 1 As a Cartesian product of simplices, the 3-polytope $\Delta_1 \times \Delta_2$ has 5 facets.

One should not confuse its dimension $\dim S$ with the polytope dimension $\dim F$, which is equal to $|S_1| + |S_2| - 2$. A *grid orientation* is an orientation of the edges of a grid. A grid orientation naturally induces a grid orientation on any subgrid. We say a grid orientation $\mathcal{O}$ contains a grid orientation $\mathcal{O}'$ if there is a subgrid S on which the orientation induced by $\mathcal{O}$ is *isomorphic* to $\mathcal{O}'$.

Let us make precise what we consider as isomorphic orientations: An edge $\{v, w\}$ in a grid is *of dimension* i if v and w differ in the i-th coordinate. Two grid orientations $\mathcal{O}$ and $\mathcal{O}'$ are *isomorphic* if there is a graph isomorphism $\varphi : \mathcal{O} \to \mathcal{O}'$ that preserves orientation, and in which a pair of edges $\{v, w\}$ and $\{v', w'\}$ is of the same dimension if and only if $\{\varphi(v), \varphi(w)\}$ and $\{\varphi(v'), \varphi(w')\}$ are of the same dimension. Such an isomorphism is a permutation of the r coordinates combined with a permutation of each of the sets $[n_i]$.

Given a grid orientation $\mathcal{O}$ and a subgrid $S \subseteq \mathcal{O}$, a *sink* of S is a vertex $v \in S$ whose incident edges in S are all incoming. In the case $S = \mathcal{O}$, we call v a *(global) sink* of $\mathcal{O}$. A *unique sink (grid) orientation (USO)* is a grid orientation in which each subgrid contains a unique sink. This is equivalent to saying that any subgrid has a unique *source*, where a source is defined analogously to a sink [4, Theorem 1]. Although abstract objective functions were previously studied, the term unique sink orientation was coined by Szabo and Welzl [7] in 2001. First introduced specifically for hypercubes, they were later generalized for grids [4].

A USO is called *admissible*, if in any subgrid of size $(n'_1, \cdots, n'_r)$ there exist $n'_1 + \cdots + n'_r - r$ many internally disjoint dipaths from the unique source to the unique sink[1]. This is exactly the Holt-Klee condition for faces. One can show that a USO of a two-dimensional grid is admissible if and only if it does not contain the *double-twist* DT in Figure 2 [1]. Indeed, the double twist violates the Holt-Klee condition, as there are only two instead of three internally disjoint paths from the source to the sink. Figure 3(a) shows an example of an admissible USO.

Furthermore, a grid orientation is called *realizable* if it is the orientation induced by a linear function on a polytope P which is combinatorially equivalent to a product of simplices. The reason why we only ask for a combinatorially equivalent polytope is this: A precise Cartesian product of simplices contains many classes of parallel edges. For instance, in $\Delta_1 \times \Delta_2$ (see Figure 1), there are four classes of parallel edges. In a linear orientation, parallel edges must

[1] In difference to [1], our definition of admissibility does not require acyclicity. However, two-dimensional USOs are always acyclic.

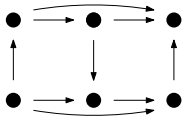


Figure 2 Double twist DT; acyclic but non-admissible USO

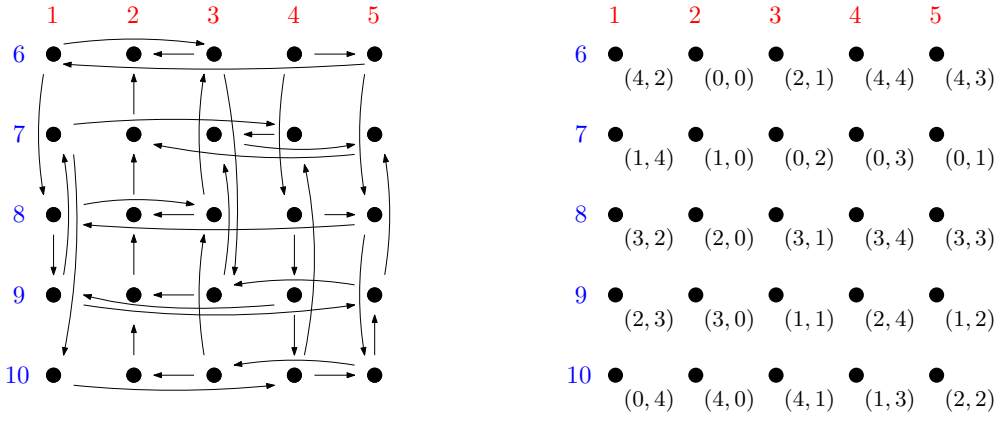


Figure 3 (a): Example of an admissible USO. Transitive arrows are not drawn. (b): Refined index of the orientation in (a).

have the same orientation. For our analysis, this is too strict. We allow the faces to be perturbed as long as the face lattice is still isomorphic to a product of simplices.

Every realizable orientation is admissible; and every admissible orientation is, by definition, a unique sink orientation. However, for both implications, the reverse is not true. It was one of the original motivations for the study of USOs to characterize realizable orientations. Admissibility is necessary but not sufficient.

For any vertex $x = (x_1, \cdots, x_r)$ in a grid orientation, let $\mathrm{rf}_i(x)$ be the out-degree in dimension i, i.e. the number of neighbors $y = (y_1, \cdots, y_r)$ with orientation $x \to y$ that have $x_i \neq y_i$ and $x_j = y_j$ for all $j \neq i$. The *refined index* of x is defined as

$$\mathrm{rf}(x) := (\mathrm{rf}_1(x), \cdots, \mathrm{rf}_r(x)).$$

It is easy to see that a USO is uniquely determined by its refined index. As an example, Figure 3(b) shows the refined index of the orientation on the left. Observe that all tuples $[4]_0 \times [4]_0$ (where $[n]_0 := \{0, \cdots, n\}$) appear as a refined index exactly once. This is not a coincidence:

▶ **Lemma 1.1** (Theorem 2 in [4]). *The refined index of a USO is a bijection*

$$\mathrm{rf} : [n_1] \times \cdots \times [n_r] \to [n_1 - 1]_0 \times \cdots \times [n_r - 1]_0 \ .$$

2 Block Colored Pseudoline Arrangements and Admissible USOs

An *arrangement of pseudolines* is a finite family of simple curves $f_1, \cdots, f_n : \mathbb{R} \to \mathbb{R}^2$ with

$$\lim_{t\to\infty} \|f_i(t)\| = \lim_{t\to-\infty} \|f_i(t)\| = \infty \ ,$$

called *pseudolines*, which fulfill the property that each two of them intersect at exactly one point where they cross. We label the pseudolines from 1 to n in counterclockwise order as in Figure 4. In this article, we only consider *simple* arrangements, i.e. at each crossing exactly two pseudolines meet. They are in correspondence with 3-*signotopes*, which we define next.

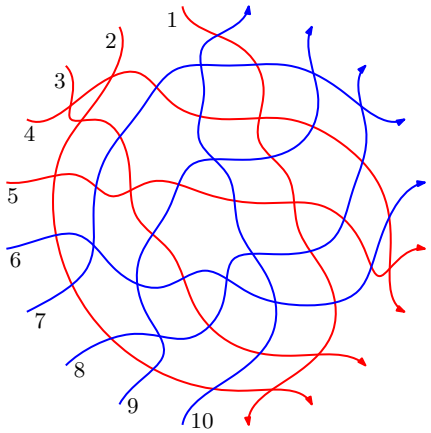
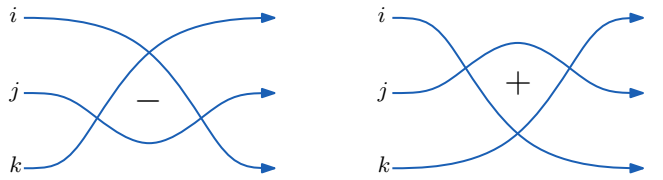


Figure 4 (a): Block colored pseudoline arrangement $\mathcal{A}$ of $r = 5$ red and $b = 5$ blue pseudolines.

For an $(r+1)$-subset $A \subseteq [n]$ and $1 \leq i \leq n$, let $A^{\lfloor i \rfloor}$ be the r-subset of A that contains all elements except the i-th element in increasing order. For example, $\{2,5,7,8\}^{\lfloor 2 \rfloor} = \{2,7,8\}$. For a map $\chi : \binom{[n]}{r} \to \{-,+\}$, we define the notation $\chi(A) := \left(\chi(A^{\lfloor 1 \rfloor}), \cdots, \chi(A^{\lfloor r+1 \rfloor})\right)$ as the sequence of signs that is obtained by successively omitting an element and applying χ.

Now, for $1 \leq r \leq n$, an *r-signotope* is a map $\chi : \binom{[n]}{r} \to \{-,+\}$ such that for each $(r+1)$-subset $A \subseteq [n]$ the sequence $\chi(A)$ contains at most one sign change. We call this the *monotonicity property* of signotopes.

In a pseudoline arrangement, every triple of pseudolines $1 \leq i < j < k \leq n$ forms exactly one of the two configurations shown in Figure 5. The triple orientations of pseudoline arrangements can be precisely characterized as 3-signotopes (cf. [2, Theorem 7]).



Negative triple, $\chi(i,j,k) = -$ Positive triple, $\chi(i,j,k) = +$

Figure 5 Each triple of pseudolines $i < j < k$ is either negative or positive.

A *block colored pseudoline arrangement* is a simple arrangement of n pseudolines that consists of two consecutive blocks of r red and b blue pseudolines, $n = r + b$. Figure 4 shows an example. Felsner, Gärtner and Tschirschnitz [1] associate a block colored pseudoline arrangement $\mathcal{A}$ with a two-dimensional USO. Assume that we have given a block colored arrangement $\mathcal{A}$ and want to assign to it a USO. Let p be a red and q a blue pseudoline. With x being the number of blue pseudolines that p crosses before crossing q, and y being the number of red pseudolines that q crosses before crossing p, we call (x, y) the *crossing index* $\mathrm{cri}_{\mathcal{A}}(p, q)$. The crossing indices $\mathrm{cri}_{\mathcal{A}}(p, q)$ for all pairs of a red pseudoline p and a blue pseudoline q form the refined index of a two-dimensional USO of size (b, r). For example, the USO in Figure 3 corresponds to the arrangement in Figure 4.

▶ **Theorem 2.1** (Felsner, Gärtner & Tschirschnitz [1]). *Any two-dimensional USO is admissible if and only if it is induced by a block colored pseudoline arrangement. Moreover, it is realizable if and only if it is induced by an arrangement of straight lines.*

3 Block Signotopes and USOs

Before, we described a way to obtain a USO $\mathcal{O}$ from a block colored arrangement $\mathcal{A}$. Let $\chi_{\mathcal{A}}$ be the 3-signotope corresponding to $\mathcal{A}$. Observe that we can read off the orientation $\mathcal{O}$ directly from $\chi_{\mathcal{A}}$: Say, pseudolines i, j are red and pseudoline k is blue, $i < j < k$. If $\chi_{\mathcal{A}}(i,j,k) = +$, pseudoline k first crosses i before crossing j, hence, $(\mathrm{cri}_{\mathcal{A}}(i,k))_2 < (\mathrm{cri}_{\mathcal{A}}(j,k))_2$, and in the resulting grid orientation $\mathcal{O}$ we must have $(k,i) \leftarrow (k,j)$. So, a sign of triples consisting of two elements of one color and one of the other color determines the orientation of exactly one edge. This is precisely what we aim to generalize for grids of arbitrary dimension r and $(r+1)$-signotopes.

We call a partition $[n] = C_1 \dot\cup \cdots \dot\cup C_r$ a *block partition*, if $c < c'$ for all $c \in C_i, c' \in C_j$ whenever $i < j$. For example, $(C_1, C_2, C_3) = (\{1,2\}, \{3,4,5\}, \{6,7\})$ is a block partition of [7]. A signotope $\chi : \binom{[n]}{r+1} \to \{-,+\}$ of rank $r+1$ together with a block partition $[n] = C_1 \dot\cup \cdots \dot\cup C_r$ is called a *block signotope* of *rank* $(r+1)$. The previous subsection treated the special case $r = 2$; the two classes were the classes of red and blue pseudolines. We are now going to generalize this to a greater number of classes.

Let G be the grid on $C_1 \times \cdots \times C_r$. The block signotope χ naturally induces a grid orientation O_χ on G: Suppose $v, v' \in C_1 \times \cdots \times C_r$ differ exactly in the i-th component, so $v = (c_1, \cdots, c_{i-1}, c, c_{i+1}, \cdots, c_r)$ and $v' = (c_1, \cdots, c_{i-1}, c', c_{i+1}, \cdots, c_r)$ with $c, c' \in C_i$, and assume $c < c'$. If $\chi(c_1, \cdots, c_{i-1}, c, c', c_{i+1}, \cdots, c_r) = +$, then, in $\mathcal{O}_\chi$, orient $v \to v'$; otherwise, orient $v' \to v$. Note that, in the case $r = 3$, compared to the construction in the previous section, $\mathcal{O}_\chi$ is now exactly reversed and transposed. Indeed, if $i < j < k$ and $\chi(i,j,k) = +$, then, in $\mathcal{O}_\chi$, we have $(i,k) \to (j,k)$.

▶ **Theorem 3.1.** *For every block signotope $\chi : \binom{[n]}{r+1} \to \{-,+\}$ with block partition $[n] = C_1 \dot\cup \cdots \dot\cup C_r$, the orientation $\mathcal{O}_\chi$ is a USO.*

This is our main result. For an extended version including proofs we refer to the full version.

Consider a block signotope χ of rank $(r+1)$ with block partition $[n] = C_1 \dot\cup \cdots \dot\cup C_r$. For an element $(c_1, \cdots, c_r) \in C_1 \times \cdots \times C_r$ we define the *index in block i* as

$$\begin{aligned} \mathrm{rf}^i_\chi(c_1, \cdots, c_r) := \quad & \#\{c \in C_i : c > c_i,\ \chi(c_1, \cdots, c_i, c, c_{i+1}, \cdots, c_r) = +\} \\ & + \#\{c \in C_i : c < c_i,\ \chi(c_1, \cdots, c_{i-1}, c, c_i, \cdots, c_r) = -\}\ , \end{aligned}$$

and its *refined index* as $\mathrm{rf}_\chi(c_1, \cdots, c_r) := \big(\mathrm{rf}^1_\chi(c_1, \cdots, c_r), \cdots, \mathrm{rf}^r_\chi(c_1, \cdots, c_r)\big)$.

Clearly, $\mathrm{rf}_\chi(c_1, \cdots, c_r) \in \{0, \cdots, |C_1| - 1\} \times \cdots \times \{0, \cdots, |C_r| - 1\}$. In terms of $\mathcal{O}_\chi$, the index $\mathrm{rf}^i_\chi(c_1, \cdots, c_r)$ is the out-degree of vertex $(c_1, \cdots, c_r)$ in the i-th dimension. Thus, $\mathrm{rf}_\chi = \mathrm{rf}_{\mathcal{O}_\chi}$ coincide. The following result is an immediate consequence of Theorem 3.1.

▶ **Corollary 3.2.** *For any $(r+1)$-signotope χ and block partition $[n] = C_1 \dot\cup \cdots \dot\cup C_r$, the refined index is a bijection*

$$\mathrm{rf}_\chi : C_1 \times \cdots \times C_r \to \{0, \cdots, |C_1| - 1\} \times \cdots \times \{0, \cdots, |C_r| - 1\}\ .$$

We consider Corollary 3.2 to be a result of independent interest: As it holds for any block partition $[n] = C_1 \dot\cup \cdots \dot\cup C_r$, it provides general insight into the structure of signotopes.

3.1 Admissibility of 3-dimensional USOs induced by signotopes

As mentioned earlier, in the case of two-dimensional acyclic USOs, admissibility is equivalent to avoiding the double twist DT. Moreover, as a further characterization, an acyclic USO is admissible if and only if it is induced by an arrangement of pseudolines. Does this generalize to higher-dimensional grids? Gärtner [3] has determined that, up to isomorphy, there exist only two acyclic USOs of a three-dimensional cube that are not admissible. We use this characterization of non-admissible cube orientations for the following result. It shows that one direction of the first part of Theorem 2.1 can be generalized from 3-signotopes alias pseudoline arrangements to 4-signotopes: they also induce admissible grid orientations.

▶ **Theorem 3.3.** *Every USO $\mathcal{O}_\chi$ on a grid of dimension at most 3 that is induced by a block signotope χ is admissible.*

However, the converse of Theorem 3.3 is not true: Figure 6 shows an admissible orientation of a cube for which we can prove that it is not induced by any signotope.

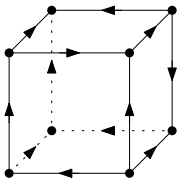


Figure 6 Admissible cube orientation that is not induced by a signotope.

4 Conclusion and Future Work

Our mission was to generalize the work from [1] to higher dimensions. For this purpose, we defined block signotopes and demonstrated in Theorem 3.1 that they induce a USO of a grid of corresponding dimension. Moreover, for dimension three they are admissible (Theorem 3.3). However, unlike in the two-dimensional case, not all higher-dimensional admissible USOs are induced by block signotopes. We conclude with the following question: Is the USO $\mathcal{O}_\chi$ always admissible for dimensions greater than three?

References

1 Stefan Felsner, Bernd Gärtner, and Falk Tschirschnitz. Grid orientations, $(d, d+2)$-polytopes, and arrangements of pseudolines. *Discrete Comput. Geom.*, 34(3):411–437, 2005. doi:10.1007/s00454-005-1187-x.

2 Stefan Felsner and Helmut Weil. Sweeps, arrangements and signotopes. *Discrete Appl. Math.*, 109(1-2):67–94, 2001. doi:10.1016/S0166-218X(00)00232-8.

3 Bernd Gärtner. The random-facet simplex algorithm on combinatorial cubes. *Random Struct. Algorithms*, 20(3):353–381, 2002. doi:10.1002/rsa.10034.

4 Bernd Gärtner, Walter D. Morris jun., and Leo Rüst. Unique sink orientations of grids. *Algorithmica*, 51(2):200–235, 2008. doi:10.1007/s00453-007-9090-x.

5 Fred Holt and Victor Klee. A proof of the strict monotone 4-step conjecture. In *Advances in Discrete and Computational Geometry*, volume 223 of *Contemp. Math.*, pages 201–216. AMS, 1999.

6 Gil Kalai. Linear programming, the simplex algorithm and simple polytopes. *Math. Program.*, 79(1-3 (B)):217–233, 1997. doi:10.1007/BF02614318.

7 Tibor Szabó and Emo Welzl. Unique sink orientations of cubes. In *Proc. 42nd IEEE Symposium on Foundations of Computer Science*, pages 547–555, 2001. doi:10.1109/SFCS.2001.959931.

Structural Properties of Shortest Flip Sequences Between Plane Spanning Trees*

Oswin Aichholzer[1], Joseph Dorfer[1], Peter Kramer[2], Christian Rieck[3], and Birgit Vogtenhuber[1]

1 **Institute of Algorithms and Theory, Graz University of Technology**
{oswin.aichholzer, joseph.dorfer, birgit.vogtenhuber}@tugraz.at
2 **Department of Computer Science, TU Braunschweig**
kramer@ibr.cs.tu-bs.de
3 **Institute of Mathematics, University of Kassel**
christian.rieck@mathematik.uni-kassel.de

Abstract

We study the reconfiguration of plane spanning trees on point sets in the plane in convex position, where a reconfiguration step (*flip*) replaces one edge with another, yielding again a plane spanning tree. The *flip distance* between two trees is then the minimum number of flips needed to transform one tree into the other. We study structural properties of shortest flip sequences.

The folklore *happy edge conjecture* suggests that any edge shared by both the initial and target tree is never flipped in a shortest flip sequence. The more recent *parking edge conjecture*, which would have implied the happy edge conjecture, states that there exist shortest flip sequences which use only edges of the start and target tree, and edges in the convex hull of the point set. Finally, another conjecture that is implicit in the literature is the *reparking conjecture* which states that no edge is flipped more than twice. Essentially all recent flip algorithms respect these three conjectures and the properties they imply. We study cases in which the latter two conjectures hold and disprove them for the general setting.

Related Version arXiv:2603.05205

1 Introduction

We investigate the reconfiguration of plane spanning trees on n-point sets P in convex position. A *flip* in a plane spanning tree removes one edge e from the tree and adds another edge f, resulting again in a plane spanning tree. The corresponding *flip graph* has as vertices all plane spanning trees on P and two vertices are connected by an edge if the trees differ in a single flip. A *flip sequence* between two trees is a path in the flip graph from one tree to the other, as shown in Figure 1, and the *flip distance* is the length of a shortest such path.

Recent research has focused on (1) the *diameter* of the flip graph, that is, the maximum flip distance between any pair of trees on convex point sets and (2) algorithmic aspects of the computation of the flip distance. For over two decades, the best bounds on the diameter were an upper and a lower bound of $2n - 4$ [4] and $\lfloor \frac{3n}{2} \rfloor - 5$ [8], respectively. The upper bound was later improved to $2n - \log(n)$ [1], and shortly after to $2n - \sqrt{n}$ [7]. In [6], the longstanding $2n$ barrier was surpassed and improved to $\approx 1.96n$. Most recently, the authors of [5] showed that the diameter of the flip graph lies between $\frac{14n}{9} - \mathcal{O}(1)$ and $\frac{5n}{3} - 3$.

* This work was initiated at the Reconfiguration Week hosted at TU Graz in March 2025. We thank the organizers and all participants for fruitful discussions and the very good atmosphere. Research was funded in part by the Austrian Science Fund (FWF) 10.55776/DOC183, a Fellowship of the German Academic Exchange Service (DAAD), and the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – 522790373.

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

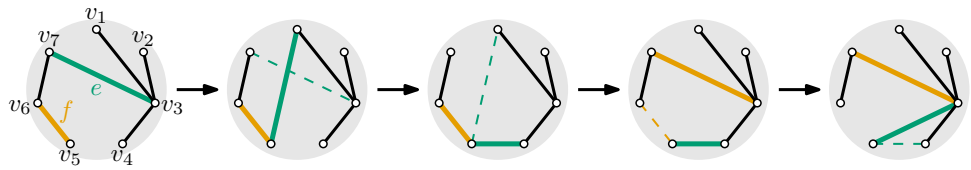


Figure 1 Flip sequence $\mathcal{S}$ with four flip types: crossing, rotation, compatible, slide (in order).

Flip types. A flip $e \to f$ that removes e and inserts f is *compatible* if the straight-line segments e and f are either disjoint or share a common endpoint, and *crossing* otherwise. For $u, v, w \in P$, a compatible flip of the shape $uv \to uw$ is called a *rotation*, and furthermore called a *slide* if $vw \in T$. Examples of all types of flips can be seen in Figure 1.

Conjectures. One tool for the investigation of diameter and complexity is the set of edges common to the start and target tree, known as *happy edges*. All other edges are referred to as *unhappy*. Building on this, the following long-standing conjecture is considered folklore.

▶ **Conjecture 1.1** (Happy edge conjecture). *Let T_I and T_F be two plane spanning trees, and let $H = T_I \cap T_F$ be the set of all happy edges. Then there exists a shortest flip sequence $T_I = T_0, T_1, \ldots, T_{k-1}, T_k = T_F$ such that $H \subset T_i$ for every $i \in [0, k]$.*

Most recent improvements to the diameter bounds respect this conjecture, that is, they never flip happy edges. Specifically, in [1], the $2n - \log(n)$ bound is presented as follows:

> *"...happy edges do not flip. An unhappy edge of T_I flips once or twice, and if twice, it flips to the convex hull and then to an edge of T_F."*

Also, they define as *parking edges* those that exist in intermediate trees of a flip sequence but not in T_I or T_F and conjecture that parking on edges of the convex hull (CH) is sufficient:

▶ **Conjecture 1.2** (Parking edge conjecture). *For any two plane spanning trees T_I and T_F, there exists a shortest flip sequence from T_I to T_F that uses only edges of $T_I \cup T_F \cup \mathsf{CH}$.*

Although it was not explicitly stated as a conjecture, the above quote suggests that there always exists a shortest flip sequence in which every unhappy edge is flipped at most twice:

▶ **Conjecture 1.3** (Reparking conjecture). *For any two plane spanning trees T_I and T_F, there exists a shortest flip sequence such that every parking edge is flipped to an edge of T_F. In particular, every edge is flipped at most twice.*

The latter conjecture is supported by the fact that, to date, every algorithm yielding a non-trivial upper bound on the diameter flips edges either directly to a final edge, or is first parked on the convex hull, from which it is then flipped to a final edge [1, 3, 5, 6, 7].

Preliminary results and structural insights towards the three conjectures include the following: (i) The parking edge conjecture implies the happy edge conjecture [1], and (ii) the happy edge conjecture does not hold when flips are restricted to edge slides [1] or rotations [3]. In contrast, (iii) the parking edge conjecture (and thus, the happy edge conjecture) holds for compatible flips [3]. For the case of unrestricted flips, the conjectures had remained open.

We make progress towards the conjectures by uncovering structural insights in the different settings. This, in turn, sheds new light on the properties of shortest flip sequences between plane spanning trees on convex point sets.

Our counterexamples to the parking and reparking properties for shortest flip sequences in the general setting show the limitations of the proposed algorithms that are based on these conjectures. The following table gives an overview of existing algorithms and their dependencies on these properties.

Table 1 Algorithms for upper bound constructions since 2022 and flip properties of conjectures they use. We highlight cells where algorithms use properties that are provably not always optimal.

Upper bound	Flip type	Conjecture 1.1	Conjecture 1.2	Conjecture 1.3
$2n - \log(n)$ [1]	general flips	Yes	Yes	Yes
$2n - \sqrt{n}$ [7]	general flips	No	Yes	Yes
$1.96n$ [6]	general flips	Yes	Yes	Yes
$\frac{5n}{3} - 3$ [5]	general flips	Yes	Yes	Yes
$\frac{5n}{3} - 2$ [3]	compatible flips	Yes	Yes	Yes
$\frac{7}{4}(n-1)$ [3]	rotations	No	No	Yes

Proofs for statements with an asterisk ($\star$) can be found in the arXiv version [2].

2 Notation and positive results on structural properties

To represent not only how flips affect a spanning tree's structure, but also individual edges, we introduce the concept of *traces*. For a flip sequence $\mathcal{S} = T_0, \ldots, T_k$, we define the *trace* of an edge $e \in T_0$ as the ordered sequence of positions that it is flipped to. Formally, this corresponds to a tuple of edges (Δ is the symmetric difference):

$$\operatorname{trace}(e, \mathcal{S}) = \operatorname{trace}(e, T_0, \ldots, T_k) = \begin{cases} \operatorname{trace}(e, T_1, \ldots, T_k) & \text{if } e \in T_0 \cap T_1, \text{ or} \\ (e) \circ \operatorname{trace}(f, T_1, \ldots, T_k) & \text{if } \{e, f\} = T_0 \Delta T_1. \end{cases}$$

For example in the flip sequence $\mathcal{S}$ in Figure 1 $\operatorname{trace}(e, \mathcal{S}) = (e,\ v_1v_5,\ v_4v_5,\ v_3v_5)$ and $\operatorname{trace}(f, \mathcal{S}) = (f,\ v_3v_7)$. A flip $e \to f$ is *final* with respect to a flip sequence exactly if the trace of f on the remaining flips is (f). We say that the *length of a trace* is the number of flips in it, which corresponds to the number of elements minus one. A happy edge $e \in T_I \cap T_F$ is *fixed* in a flip sequence $\mathcal{S}$ exactly if it has trace length zero, or equivalently $\operatorname{trace}(e, \mathcal{S}) = (e)$.

We generalize a property shown in [3, Lemma 20]. They showed that in every sequence of compatible flips, any parking flip that parks on a diagonal can be replaced with a parking flip that parks on the convex hull. It is in fact not required that the entire flip sequence is compatible, but only either the flip that inserts the parking edge or the flip that removes the parking edge has to be compatible. This already follows from their proof, but for the sake of self-containment, we include the adapted proof and rewrite the statement in terms of our definitions.

▶ **Lemma 2.1** ($\star$). *Let $\mathcal{S}$ be a flip sequence from T_I to T_F and e an edge such that $\operatorname{trace}(e, \mathcal{S}) = (e, f, e')$, where f is a diagonal and either $e \to f$ or $f \to e'$ is a compatible flip. Then there exists a flip sequence $\mathcal{S}'$ from T_I to T_F of the same length in which $\operatorname{trace}(e, \mathcal{S}') = (e, h, e')$ such that $h \in \mathsf{CH}$. The traces of all other edges remain unchanged.*

In order to study the reparking property, we start by observing sufficient properties for edges to have trace length 1 or, equivalently, necessary conditions for larger trace length.

▶ **Proposition 2.2** ($\star$ Final flip property). *For any two plane spanning trees T_I and T_F, there exists a shortest flip sequence such that every flip is either final, or the removed edge is crossed by another edge later on.*

We continue by showing that Proposition 2.2 is strong enough to settle the reparking property for compatible flips, as well as for convex hull edges of T_I in the general case.

▶ **Theorem 2.3** (⋆). *Consider any two plane spanning trees T_I and T_F of a point set P.*
(1) *There exists a shortest flip sequence such that any convex hull edge is flipped at most once.*
(2) *In the compatible case, all edges are flipped at most twice* (Compatible reparking property).

3 Counterexamples for general flips

In the previous section, we identified structural conditions that any counterexample to the parking and reparking conjectures must satisfy for general flips. With these conditions in hand, we now construct specific pairs of trees, T_I and T_F, that not only meet these requirements but also go further, yielding shortest flip sequences that violate the conjectures.

3.1 The parking edge property

In this section, we disprove Conjecture 1.2: there exist pairs of trees for which every shortest flip sequence parks on diagonal edges, and avoiding this requires significantly more flips.

▶ **Theorem 3.1** (⋆). *There exists a family of trees $(T_{I,k})_{k\in\mathbb{N}}$ and $(T_{F,k})_{k\in\mathbb{N}}$ on $10k+2$ vertices such that the length of the shortest flip sequence that only uses parking edges from the convex hull is at least by k flips longer than the shortest possible flip sequence.*

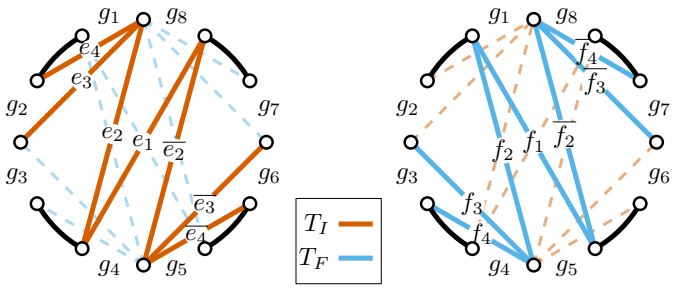
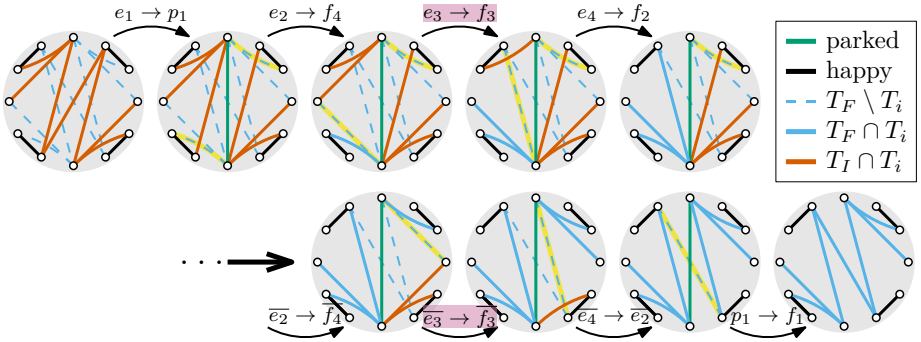


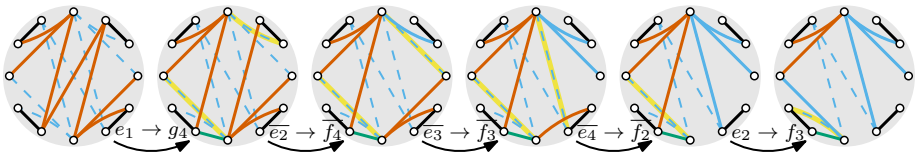
Figure 2 A counterexample to the parking edge conjecture using $n = 12$ points.

Proof sketch. We start with a counterexample for $n = 12$, as depicted in Figure 2. Since every edge in the symmetric difference $T_I \Delta T_F$ is crossed at least twice, the first flip of any sequence from T_I to T_F must necessarily be a parking flip. In Figure 3(a), we depict a flip sequence that uses a diagonal parking edge p_1 (green). The sequence contains exactly one parking flip, $e_1 \to p_1$, and all other flips are final. Hence this is a shortest flip sequence.

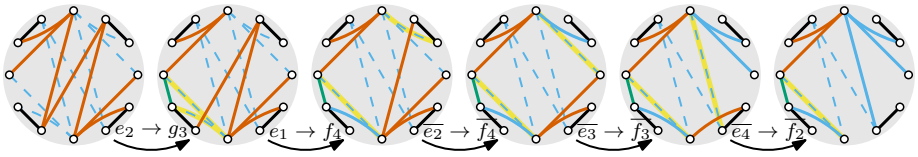
It remains to show that using a parking edge on the convex hull requires at least one additional flip. To this end, we show that no such flip sequence can reach T_F exclusively using final flips thereafter. In Figures 3(b) to 3(e), we enumerate all possible first flips (up to symmetry) that replace a diagonal of $T_I \setminus T_F$ with an edge on the convex hull. For each parking flip, we show a sequence of successive perfect flips until these are impossible. Since T_F is not reached, additional parking flips are inevitable, making such flip sequences longer than optimal. Note that Figure 3 does not represent every possible flip sequence: In some cases, multiple edges can be flipped to the same edge, or flips within the sequences could be reordered. To compute all relevant flip sequences, we employed a computer program [2]



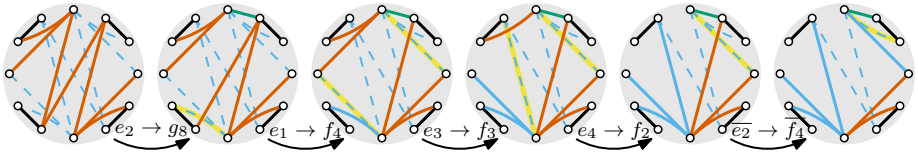
(a) A shortest flip sequence for the instance shown in Figure 2. The two highlighted flips are slides.



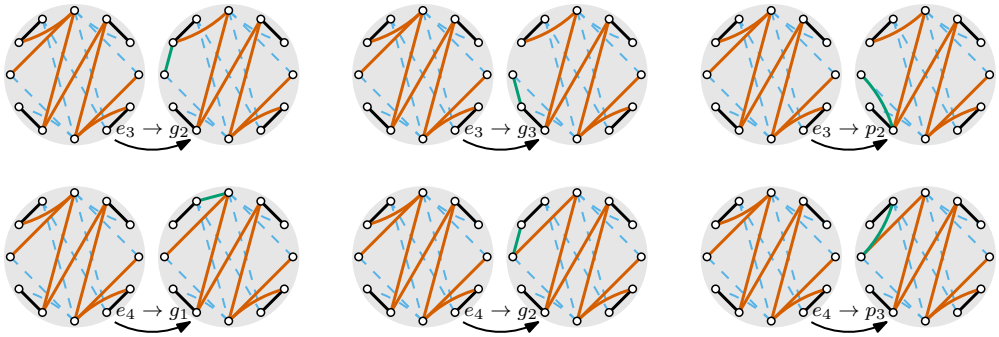
(b) A "maximal" flip sequence starting with $e_1 \to g_4$. Note that $e_1 \to g_8$ is symmetric.



(c) A "maximal" flip sequence starting with $e_2 \to g_3$. Note that $\overline{e_2} \to g_7$ is symmetric.



(d) A "maximal" flip sequence starting with $e_2 \to g_8$. Note that $\overline{e_2} \to g_4$ is symmetric.



(e) Parking either e_3 or e_4 leads to dead ends after a single flip. Note that $\overline{e_3}$ and $\overline{e_4}$ have symmetric flips.

Figure 3 We illustrate a shortest flip sequence (a) that parks the edge e_1 diagonally, alongside the maximal number of perfect flips possible if another edge pair is chosen for parking in the first flip. For clarity edges of T_F that are can currently be inserted are emphasized.

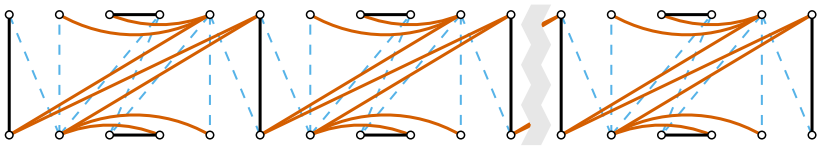


Figure 4 A counterexample to the parking edge conjecture using $n = 10k + 2$ points.

that systematically generates sequences and constructs the flip graph. This graph can be verified independently using our program and is depicted in the full version of this paper [2].

Restricting to parking edges on the convex hull can incur a single additional flip even for $n = 12$. As the second component of Theorem 3.1, we prove the existence of instances that require $\Omega(n)$ additional flips under the same constraint. We show this for instances as depicted in Figure 4, which we obtain by "gluing" together multiple copies from Figure 2. ◀

On the positive side, we can show that flip sequences that park solely on the convex hull remain close to an optimal flip sequence. More precisely, we prove that a diagonal parking flip can be replaced by two convex hull parking flips. Applying Proposition 3.2 once for every diagonal parking flip yields a flip sequence without any parking flip, but at the cost of one additional flip per iteration.

▶ **Proposition 3.2** (⋆). *A flip sequence that adds and removes a parking diagonal d by non-compatible flips can be replaced with a flip sequence that does not perform a diagonal parking flip that involves d and that is only one flip longer. In particular, the resulting flip sequence performs strictly fewer diagonal parking flips than the original one.*

3.2 The reparking property

From Lemma 2.1 and Proposition 2.2, we derive necessary conditions for a counterexample to the reparking property as follows:

▶ **Proposition 3.3** (⋆). *Suppose there exists a shortest flip sequence with an edge e with $trace(e) = (e, e_1, e_2, e_3)$ of length 3. If no other shortest flip sequence exists in which e has shorter trace, the following conditions must hold: (1) e and e_3 are diagonals, (2) e and e_1 cross, (3) e_1 and e_2 cross, and (4) e_2 and e_3 cross.*

▶ **Theorem 3.4.** *There exist pairs of plane spanning trees T_I and T_F, and T'_I and T'_F on 22 and 32 points, respectively, such that any shortest flip sequence from T_I to T_F, respectively T'_I to T'_F, flips the same edge 3, respectively 4, times.*

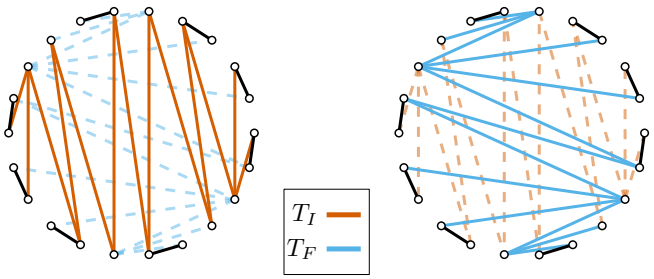


Figure 5 A counterexample to the reparking property.

Proof. Figure 5 provides a counterexample to the reparking property. It is constructed with the conditions of Proposition 3.3 in mind. Any shortest flip sequence from T_I to T_F contains an edge with trace length 3. A sketch of a shortest flip sequence is given in Figure 6. First, the central diagonal is flipped to the green diagonal. Then, six perfect flips insert edges from T_F. Subsequently, the green diagonal is flipped again. This again allows for six perfect flips before the green diagonal is flipped to its final position.

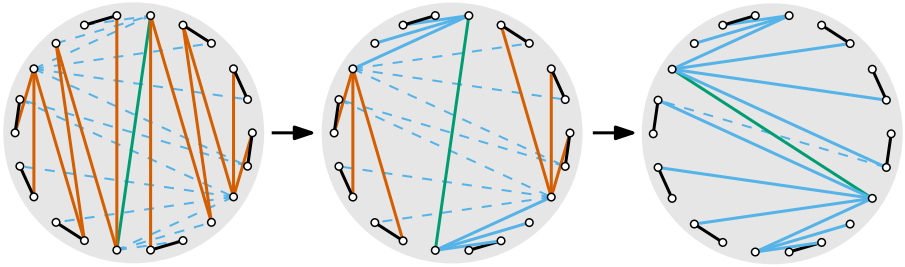


Figure 6 A sketch of a shortest flip sequence for the two trees shown in Figure 5.

We took this construction one step further and provide a pair of trees in which for every shortest flip sequence there exists an edge that has trace length four, that is, it is reparked twice. The example is depicted in Figure 7. To verify this, we again used our program [2]. ◀

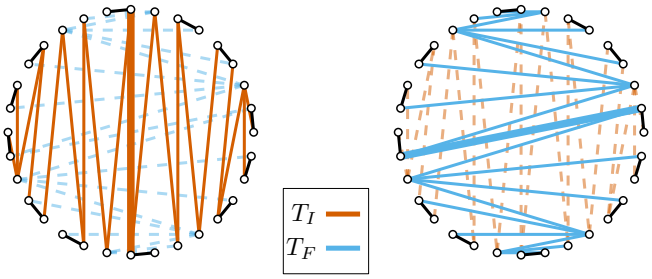


Figure 7 An example with 32 vertices in which every shortest flip sequence contains an edge that has trace length four (highlighted as the bold edge).

4 Conclusions and future work

We constructed explicit counterexamples demonstrating that the parking and reparking properties fail for shortest flip sequences in the general setting. In contrast, we prove that the reparking property holds when flips are restricted to be compatible. This highlights the limits of existing assumptions and algorithms.

By disproving the parking conjecture for general flips, we have eliminated a previously promising approach of establishing the happy edge conjecture. Consequently, we think that entirely new methods are required to make further progress on that conjecture.

Moreover, we provided a construction of initial and final trees where every shortest flip sequence for these pairs has an edge of trace length 3 or 4, respectively. We expect that this construction can be iterated without limit. Thus, we raise the following conjecture:

▶ **Conjecture 4.1.** *For any constant c there exists a pair of trees T_I and T_F such that any shortest flip sequence between T_I and T_F has an edge with trace length larger than c.*

References

1 Oswin Aichholzer, Brad Ballinger, Therese Biedl, Mirela Damian, Erik D. Demaine, Matias Korman, Anna Lubiw, Jayson Lynch, Josef Tkadlec, and Yushi Uno. Reconfiguration of non-crossing spanning trees. *Journal of Computational Geometry*, 15(1):224–253, 2024. doi:10.20382/jocg.v15i1a9.

2 Oswin Aichholzer, Joseph Dorfer, Peter Kramer, Christian Rieck, and Birgit Vogtenhuber. Structural properties of shortest flip sequences between plane spanning trees. 2026. arXiv:2603.05205.

3 Oswin Aichholzer, Joseph Dorfer, and Birgit Vogtenhuber. Constrained flips in plane spanning trees. In *Symposium on Graph Drawing and Network Visualization (GD)*, pages 5:1–5:18, 2025. doi:10.4230/LIPIcs.GD.2025.5.

4 David Avis and Komei Fukuda. Reverse search for enumeration. *Discrete Applied Mathematics*, 65(1):21–46, 1996. doi:10.1016/0166-218X(95)00026-N.

5 Håvard Bakke Bjerkevik, Linda Kleist, Torsten Ueckerdt, and Birgit Vogtenhuber. Flipping non-crossing spanning trees. In *Symposium on Discrete Algorithms (SODA)*, pages 2313–2325, 2025. doi:10.1137/1.9781611978322.77.

6 Nicolas Bousquet, Lucas De Meyer, Théo Pierron, and Alexandra Wesolek. Reconfiguration of plane trees in convex geometric graphs. *Discrete & Computational Geometry*, pages 1–34, 2025. doi:10.1007/s00454-025-00785-8.

7 Nicolas Bousquet, Valentin Gledel, Jonathan Narboni, and Théo Pierron. A note on the flip distance between non-crossing spanning trees. *Computing in Geometry and Topology*, 2023. doi:10.57717/cgt.v2i1.36.

8 M. Carmen Hernando, Ferran Hurtado, Alberto Márquez, Mercè Mora, and Marc Noy. Geometric tree graphs of points in convex position. *Discrete Applied Mathematics*, 93(1):51–66, 1999. doi:10.1016/S0166-218X(99)00006-2.

On t-colorable k-plane drawings

Miriam Goetze[1], Michael Kaufmann[2], and Soeren Terziadis[3]

1 **Karlsruhe Institute of Technology**
`miriam.goetze@kit.edu`
2 **University of Tübingen**
`michael.kaufmann@uni-tuebingen.de`
3 **TU Munich and TU Eindhoven**
`soeren.terziadis@ac.tuwien.ac.at`

Abstract

In this work, we introduce *t-colorable k-plane drawings*, that is, drawings of graphs with a t-edge-coloring where every edge is crossed by at most k edges of each color. We give tight upper bounds on the edge- and crossing density for small values of t and k and show that the recognition of such drawings is NP-complete if $t \geq 2$ and $k \geq 1$.

1 Introduction

Beyond planar graphs have been studied extensively over the last decades. A particular focus has been *k-plane drawings*, that is drawings of graphs where every edge is crossed at most k times. For $k \in \{1, 2, 3\}$ the structure of such drawings is well-understood [4] and bounds on the edge density [6, 12–15] and crossing density [3, 9] have been obtained. While it is easy to check whether a given drawing is k-plane (for given k), it is NP-hard to recognize graphs which admit such drawings [16].

In this work, we consider similar questions for *t-colorable k-plane drawings*, that is drawings that admit a t-edge-coloring such that each edge has at most k crossings with edges of each color, see Fig. 1 for an illustration. Such edge-colorings correspond to frugal colorings (see Sec. 2 for a definition) of string graphs. Upper bounds on the minimum number of colors needed in proper k-frugal colorings have been obtained previously for planar graphs [2] and graphs of bounded maximum degree [10, Theorem 3.5].

We give upper bounds on the edge density for small t and k (see Table 1a for an overview) and on the crossing density of 2-colorable 1-plane drawings (cf. Sec. 4). Further, we investigate the complexity of recognizing t-colorable k-plane drawings, see Table 1b for an overview. Results marked with ★ are proven in the full version of the paper.

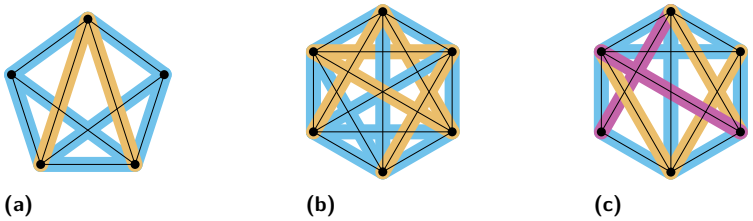


Figure 1 Examples of t-colorable k-plane drawings for (a) $t = 2, k = 1$, (b) $t = k = 2$, as well as (c) $t = 3, k = 1$.

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

(a)

	1-colorable	2-colorable	3-colorable
$k = 1$	$4(n-2)$ [6]	$4.\overline{3}(n-2)$ ⟨3.2,3.3⟩	$5(n-2)$ ★
$k = 2$	$5(n-2)$ [14, 15]	$6(n-2)$ [1, 7]	Open

(b)

	1-colorable	2-colorable	3-colorable	$(t > 3)$-colorable
$k = 1$	trivial	$\mathcal{O}(n+m)$ ⟨5.1⟩	NPC ★	NPC ★
$k = 2$	trivial	NPC ★	NPC ⟨5.3⟩	NPC ⟨5.3⟩
$k > 2$	trivial	NPC ⟨5.3⟩	NPC ⟨5.3⟩	NPC ⟨5.3⟩

Table 1 Overview of results for t-colorable k-plane drawings. previously known implied by Thm. 5.3. (a) Maximum number of edges for drawings on n vertices. (b) Recognition complexity.

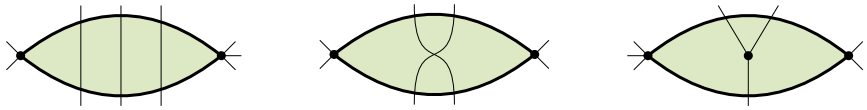


Figure 2 Examples of lenses (lightgreen) with their boundary (thick). An empty lens (a) cannot appear in a non-homotopic drawing, while lenses that contain a crossing or a vertex (b-c) may.

2 Preliminaries and notation

We consider drawings in the plane, where vertices are represented by points and edges by simple Jordan curves. Edges only cross in internal points, do not share an infinite number of points or touch (sharing a point without crossing), edges do not self-intersect and no more than two edges cross in a point, neither may adjacent edges cross.

The drawings we consider may contain parallel edges (but no loops). In general, the number of edges of such drawings may be arbitrarily large. However, we only consider *non-homotopic* drawings. That is, drawings where every region whose boundary is formed by (parts) of exactly two edges (called a *lens*) contains a vertex or a crossing in its interior, see Fig. 2 for examples. We call the subdrawing of a drawing Γ consisting of all plane edges the *plane skeleton* of Γ.

t-colorable k-plane drawings. We call a graph G *t-colorable k-planar* if it admits a drawing Γ and a (not necessarily proper) t-edge-coloring $\varphi\colon E(G) \to [t]$ such that each edge is crossed by at most k edges in Γ for each of the t colors. (As usual $[t] = \{1, \ldots, t\}$.) The drawing Γ is called *t-colorable k-plane* and its witness t-edge-coloring *k-plane*.

▸ **Observation 1.** *Every t-colorable k-plane drawing is kt-plane.*

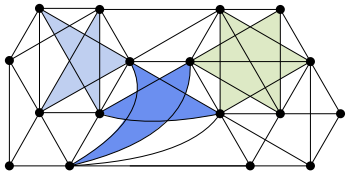


Figure 3 A drawing that contains a -graph (■) and a -configuration (■). While another region (■) contains a subdrawing isomorphic to a -graph, it is no -configuration.

Segments, cells and configurations. An edge with ℓ crossings is split into $\ell+1$ parts. The parts incident to a vertex are called *outer segments*, the other *inner* segments. The labeled embedded graph obtained by replacing each crossing with a vertex is the *planarization* $\Lambda(\Gamma)$ of drawing Γ. The labels indicate whether a vertex of $\Lambda(\Gamma)$ corresponds to a vertex or a crossing of Γ. Regions of Γ that correspond to faces of $\Lambda(\Gamma)$ are *cells*, denoted by $C(\Gamma)$. The boundary ∂c of a cell $c \in C(\Gamma)$ consists of a set of closed walks. A walk may contain the same segment twice. A *configuration* is a labeled embedded subgraph of the planarization $\Lambda(\Gamma)$. Intuitively, it is an arrangement of one or several adjacent cells of Γ. Two configurations are of the same *type* if they are isomorphic as labeled embedded subgraphs (up to reflection). We denote different types by small pictograms such as or , see Fig. 3 for an illustration.

***k*-frugal colorings.** For a drawing Γ, consider the *conflict graph* H with vertices $V(H) = E(\Gamma)$ corresponding to the edges of Γ. Two vertices in $V(H)$ are joined by an edge in $E(H)$ if the corresponding edges e, e' cross in Γ. If Γ is t-colorable k-plane, its edge-coloring corresponds to a vertex-coloring $\Phi\colon V(H) \to [t]$ where each color class appears at most k times in the neighborhood of each vertex, that is $\left|N(v) \cap \Phi^{-1}(i)\right| \le k$ for every $v \in V(H)$ and each color $i \in [t]$. Note that Φ need not be proper. Such a coloring of a graph H is called *k-frugal*. The minimum number of colors $\varphi_k(H)$ among all k-frugal colorings of H is the *k-frugal chromatic number*. A k-plane t-edge-coloring of a drawing Γ now corresponds to a k-frugal t-vertex-coloring of its conflict graph H. Hence, we are interested in (not necessarily proper) k-frugal colorings of *string graphs*, i.e., intersection graphs of Jordan curves.

▸ **Observation 2.** *For a graph H of maximum degree $\Delta(H)$, we have $\varphi_k(H) \ge \frac{\Delta(H)}{k}$.*

3 Edge density of t-colorable k-planar graphs for small t and k

In this section, we take a closer look at the maximum number of edges in t-colorable k-planar graphs for $kt \le 4$. Recall that every such graph is in particular kt-planar (cf. Obs. 1). For $kt \le 3$, some configurations appear in every sufficiently dense kt-plane drawing [8]. Yet, they may not be part of any t-colorable k-plane drawing due to the coloring restriction. For $t = 2, k = 1$, this holds for - and -configurations (cf. Sec. 3.1), and for $t = 3, k = 1$ for all -configurations (★). The densest 4-plane drawings turn out to be 2-colorable 2-plane (★). In each setting, this yields an upper bound on the edge density, with a matching lower bound based on modifications of the densest kt-plane drawings.

▸ **Theorem 3.1.** *Let Γ be a non-homotopic drawing of a graph on n vertices.*

(i) If Γ is 2-colorable 1-plane, then $|E(\Gamma)| \le 4.\overline{3}(n-2)$.
(ii) If Γ is 3-colorable 1-plane, then $|E(\Gamma)| \le 5(n-2)$.
(iii) If Γ is 2-colorable 2-plane, then $|E(\Gamma)| \le 6(n-2)$.

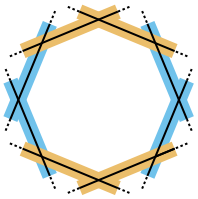


Figure 4 The coloring of an inner cycle (of size 8) in a 2-colorable 1-plane drawing.

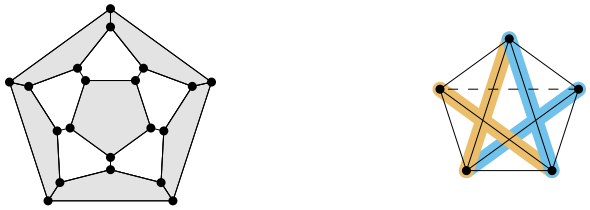


Figure 5 A 2-colorable 1-plane drawing based on [15, Figure 3]. (a) A plane drawing with pentagonal faces. (b) To each pentagonal face all but one diagonal (dashed) are added.

In each case, there exists an infinite family of drawings which matches the upper bound.

3.1 2-colorable 1-plane drawings

For a cell $c \in C(\Gamma)$ of a 2-colorable 1-plane drawing Γ, we call a closed walk w of ∂c an *inner cycle* if the component of the conflict graph containing the crossings of w is a cycle, see Fig. 4. The size of w is the number of incident segments. In a 2-colorable 1-plane drawing, each edge e that belongs to an inner cycle w is crossed by two edges e' and e'' of different colors.

▸ **Observation 3.** *Let Γ be a connected 2-colorable 1-plane drawing (in blue and orange) with an inner cycle w of size s. The segments of w (in the order on the boundary) are colored with the pattern blue, blue, orange, orange; see Fig. 4. In particular, $s \equiv 0 \mod 4$.*

Büngener and Kaufmann showed that every non-homotopic 2-plane drawing of an n-vertex graph with $4.\overline{3}(n-2)+x$ edges for $x \in [0, 0.\overline{6}(n-2)]$ contains at least x configurations of type and [8, Corollary 2]. Yet, and contain an inner cycle of size 5 and 6 respectively and thus do not appear in any 2-colorable 1-plane drawing.

▸ **Corollary 3.2.** *Every non-homotopic 2-colorable 1-plane drawing Γ of a graph on n vertices contains at most $4.\overline{3}(n-2)$ edges.*

A small modification of the known densest 2-plane drawings [15] yields a matching lower bound on the edge density of 2-colorable 1-plane drawings, see Fig. 5 for an illustration.

▸ **Proposition 3.3.** *For every $n = 2+3\ell$ with $\ell \in \mathbb{N}^+$, there exists a non-homotopic 2-colorable 1-plane drawing of a graph on n vertices with $4.\overline{3}(n-2)$ edges and $2(n-2)$ crossings.*

4 Crossing density of 2-colorable 1-plane drawings

The aim of this section is to prove the following:

Figure 6 (a) A face f with $\mathrm{cc}(f) = 10$. (b) A face g with $\mathrm{cc}(g) = 10$.

▸ **Theorem 4.1.** *Every non-homotopic 2-colorable 1-plane drawing of a graph on $n \geq 5$ vertices contains at most $2.\overline{6}n$ crossings.*

In fact, it is enough to prove Thm. 4.1 for what we call *filled* drawings, a concept used in previous work, e.g. [9]. A drawing Γ is *filled* if for every cell c of Γ, every pair of (distinct) vertices u, v on the boundary ∂c is connected with a plane edge uv on ∂c.

▸ **Observation 4.** *Every non-homotopic 2-colorable 1-plane drawing Γ is contained in a filled non-homotopic 2-colorable 1-plane drawing Γ'.*

Yet, even if Γ is a drawing of a simple graph, Γ' may contain parallel edges. In order to prove Thm. 4.1, we first provide an upper bound on the number of crossings within faces of the plane skeleton Γ_0. This bound is based on the function $\mathrm{cross}\colon \mathbb{N} \to \mathbb{N}_0$ with

$$\mathrm{cross}(x) = \begin{cases} 0, & \text{if } x \leq 3, \\ 2, & \text{if } x = 4, \\ x-1, & \text{if } x \geq 5 \text{ and } x \not\equiv 0 \mod 4, \\ x, & \text{if } x \geq 5 \text{ and } x \equiv 0 \mod 4, \end{cases}$$

for all x and what we call the *circumference* of faces. Each edge of Γ_0 has two *sides*, both of which may be incident to the same face $f \in C(\Gamma_0)$. The *circumference* $\mathrm{cc}(f)$ of a face $f \in C(\Gamma_0)$ is the number of sides incident to f, see Fig. 6 for examples.

▸ **Lemma 4.2 (★).** *In a filled non-homotopic 2-colorable 1-plane drawing Γ on $n \geq 5$ vertices, every face f of the plane skeleton Γ_0 contains at most $\mathrm{cross}(\mathrm{cc}(f))$ crossings.*

For filled drawings, we now obtain an upper bound on the number of crossings.

▸ **Theorem 4.3.** *Every filled non-homotopic 2-colorable 1-plane drawing Γ of a graph on n vertices with $n \geq 5$ has at most $2.\overline{6}(n-2)$ crossings.*

Proof. Let a_i be the number of faces f in the plane skeleton Γ_0 with $\mathrm{cc}(f) = i$ and let X be the set of crossings in Γ. For $i \geq 3$, we define $t_i = i - 2$. If f is connected, t_i corresponds to the number of triangular faces the face f would contain in a plane triangulation of Γ_0. Otherwise, the number t_i provides a lower bound on this number. As a plane triangulation of Γ_0 consists of $2n-4$ faces, we obtain $\sum_{i=3}^{\infty} t_i \cdot a_i \leq 2n-4$. By Lem. 4.2, we have

$$|X| \leq \sum_{i=3}^{\infty} \mathrm{cross}(i) \cdot a_i = \sum_{i=3}^{\infty} \frac{1}{t_i} \cdot \mathrm{cross}(i) \cdot t_i \cdot a_i \leq \sum_{i=3} \frac{4}{3} \cdot t_i \cdot a_i \leq \frac{8}{3} \cdot (n-2) = 2.\overline{6}(n-2).$$

where we used in the second step, that $\frac{1}{t_i}\mathrm{cross}(i) \leq \frac{1}{t_8}\mathrm{cross}(8) = \frac{4}{3}$ for all $i \geq 3$. Indeed, for $i \geq 5$ we have $\frac{i-1}{i-2} \leq \frac{4}{3}$ and for $i \geq 8$ we obtain $\frac{i}{i-2} \leq \frac{4}{3}$. ◂

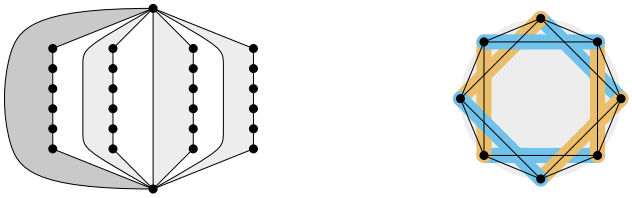


Figure 7 Construction of a 2-colorable 1-plane drawing on n vertices with $2.\overline{6}(n-2)$ crossings. (a) A plane drawing on $n = 26$ vertices where each face has size 8. (b) To each face f all diagonals connecting two vertices at distance 2 on the boundary of f are added.

As every 2-colorable 1-plane drawing can be augmented to a filled one by only adding plane edges, Thm. 4.1 follows. A drawing that matches the upper bound given in Thm. 4.1 can be obtained from a tiling of 8-gons by inserting a -configuration in each face, see Fig. 7a.

▸ **Proposition 4.4.** *For every $n = 2+6\ell$ with $\ell \in \mathbb{N}^+$, there exists a non-homotopic 2-colorable 1-plane drawing of a graph on n vertices with $2.\overline{6}(n-2)$ crossings and $4(n-2)$ edges.*

There are 3-plane drawings matching both the maximum crossing- and edge density among all 3-plane drawings [9]. Yet, while the upper bounds for edge and crossing density for 2-colorable 1-plane drawings are tight (cf. Prop. 3.3 and 4.4), no drawings matching both bounds are known.

5 Recognizing t-colorable k-plane drawings

Here we consider the recognition problem of t-colorable k-plane drawings, which we assume to be given as a planarization of the drawing. For $t = 1$, it suffices to verify that no edge is crossed more than k times. For $t = 2$ and $k = 1$, the problem still lies in P.

▸ **Theorem 5.1.** *Given a 2-plane drawing Γ of a graph G on n vertices and m edges, we can compute in time $\mathcal{O}(n+m)$ a 1-plane 2-coloring $\varphi\colon E(G) \to [2]$, or conclude that none exists.*

Proof. The conflict graph of the m edges of Γ consists of cycles and paths. All vertices of a path colored with the pattern orange, orange, blue, blue have a blue and a orange neighbor. Cycles admit such a coloring if and only if they are of size $s \equiv 0 \mod 4$ (cf. Obs. 3). This can be checked and (if applicable), such a coloring can be computed in $O(n+m)$ time. ◂

For all remaining values of t and k, the problem is NP-complete, shown using reductions from an intermediate problem called Consistent Segment t-coloring Problem (t-CSC). An instance I of t-CSC is an integer t, a set of non-crossing segments S and a set of gadgets G; there are 5 types of gadgets. A gadget $g \in G$ *connects* a set of segments $S' \subset S$ and prevents certain relative colorings between segments in S', depending on the type of g. Then I is a yes-instance if there exists a coloring $\varphi : S \to [t]$ consistent with restrictions imposed by the gadgets. For segments $s_i \in S$, a gadget g can enforce that (i) $\varphi(s_1) = \varphi(s_2)$, (ii) $\varphi(s_1) \neq \varphi(s_2)$, (iii) $\varphi(s_1) = \varphi(s_2) = \varphi(s_3)$, (iv) $\varphi(s_1) = \varphi(s_3)$ and $\varphi(s_2) = \varphi(s_4)$ or (v) that we do not have $\varphi(s_1) = \varphi(s_2) = \varphi(s_3)$. For a detailed overview we refer to (★).

▸ **Theorem 5.2 (★).** *The t-CSC problem is NP-complete for any $t \geq 2$.*

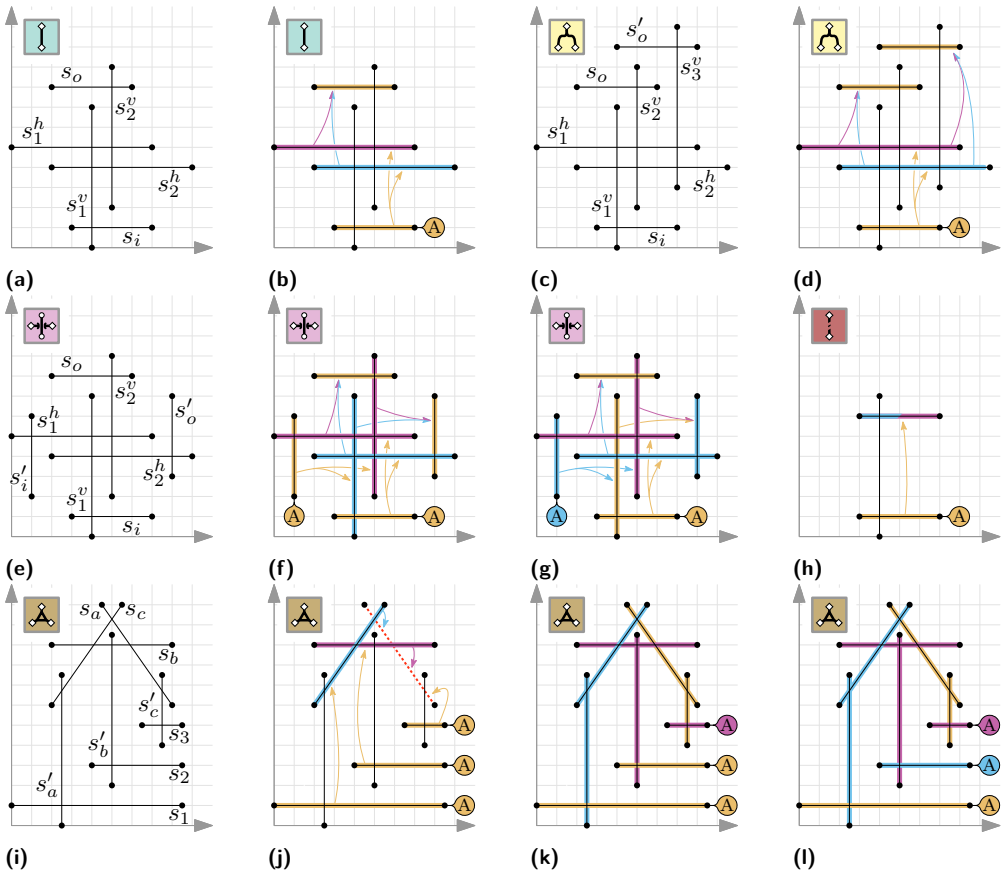


Figure 8 3-colorable 1-plane drawings for the equality (a)-(b), splitter (c)-(d), crossing (e)-(g), inequality (h), and not-all-equal (i)-(l) gadgets, together with edge-colorings. A coloring marked with A is assumed. Arrows show when the coloring of one edge forces another one. Exact geometry is not crucial, only the intersection pattern. The red dashed edge in (j) cannot be colored validly.

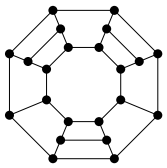


Figure 9 A planar 3-regular graph with no 2-frugal 2-coloring. Found by automatically checking examples provided by House of Graphs. The graph is taken from [5, p. 88].

When reducing t-CSC to recognizing t-colorable k-plane drawings, segments are simply represented as edges in a drawing and it is sufficient to provide five drawings which mimic the coloring restrictions of the gadgets in a k-plane t-edge-coloring. An example of these constructions for $t = 3$ and $k = 1$ is shown in Fig. 8. We can construct gadgets for any $t > 3, k = 1$ and $t = 2, k = 2$. Additionally, given a drawing Γ, we can construct one that is t-colorable $(k+1)$-plane if and only if Γ is t-colorable k-plane (★). This results in

▸ **Theorem 5.3 (★).** *Deciding if a given drawing is t-colorable k-plane is* NP*-complete for any $t = 2, k \geq 2$ and $t \geq 3, k \geq 1$.*

6 Future Work

Our research gives rise to structural questions with regard to the difference between fixed drawings and graphs, as well as complexity theoretic questions.

Every t-colorable k-plane drawing is in particular tk-plane. However, there are 4-plane drawings, which are not 2-colorable 2-plane. In fact, the 3-regular graph H represented in Fig. 9 is the conflict graph of a 3-plane drawing as H is planar (i.e., in particular a string graph). As H admits no 2-frugal 2-coloring, the corresponding drawing is not 2-colorable 2-plane. Yet, H might admit a different 3- or 4-plane drawings that is 2-plane 2-colorable. This raises the following question.

▸ Question 1. Is there a 4- or 3-planar graph which is not 2-colorable 2-planar?

While we constructed a 2-colorable 1-plane drawing of n-vertex graphs G that match the upper bound on the number of crossings, the graphs G might still admit a different 2-colorable 1-plane drawings with fewer crossings.

▸ Question 2. Are there n-vertex graphs where every 2-colorable 1-plane drawing contains $2.\overline{6}(n-2)$ crossings?

We only provided bounds on the number of crossings of 2-colorable 1-plane drawings.

▸ Question 3. What is the crossing density of 3-colorable 1-plane drawings?

In this work, we considered a colored variant of k-planarity. Yet, the same questions on edge and crossing density, as well as recognition complexity can also be applied to different notions of beyond planarity, such as fan-planar or quasi-planar graphs. In fact, for the uncolored variants many edge density results can be obtained using the density formula [11]. Can the density formula be generalized to colored drawings?

References

1 Eyal Ackerman. On topological graphs with at most four crossings per edge. *Computational Geometry*, 85:101574, 2019. doi:10.1016/j.comgeo.2019.101574.

2 Omid Amini, Louis Esperet, and Jan Van Den Heuvel. Frugal Colouring of Graphs. arXiv preprint, 2007. doi:10.48550/arXiv.0705.0422.

3 Michael A. Bekos, Prosenjit Bose, Aaron Büngener, Vida Dujmović, Michael Hoffmann, Michael Kaufmann, Pat Morin, Saeed Odak, and Alexandra Weinberger. On k-planar graphs without short cycles. In *32nd International Symposium on Graph Drawing and Network Visualization, (GD 2024)*, volume 320 of *LIPIcs*, pages 27:1–27:17, 2024. doi:10.4230/LIPIcs.GD.2024.27.

4 Michael A. Bekos, Michael Kaufmann, and Chrysanthi N. Raftopoulou. On Optimal 2- and 3-Planar Graphs. In *33rd International Symposium on Computational Geometry (SoCG 2017)*, volume 77 of *LIPIcs*, pages 16:1–16:16, 2017. doi:10.4230/LIPIcs.SoCG.2017.16.

5 Allan Bickle. Plane triangulations without spanning 2-trees. *Australasion Journal of Combinatorics*, 85(1):82–91, 2023.

6 Rainer Bodendiek, Heinz Schumacher, and Klaus Wagner. Bemerkungen zu einem Sechsfarbenproblem von G. Ringel. *Abhandlungen aus dem Mathematischen Seminar der Universität Hamburg*, 53:41–52, 1983. doi:10.1007/BF02941309.

7 Aaron Büngener. A simplified proof for the edge-density of 4-planar graphs. arXiv preprint, 2025. doi:10.48550/arXiv.2509.20999.

8 Aaron Büngener and Michael Kaufmann. Improving the crossing lemma by characterizing dense 2-planar and 3-planar graphs. In *32nd International Symposium on Graph Drawing and Network Visualization, (GD 2024)*, volume 320 of *LIPIcs*, pages 29:1–29:22, 2024. doi:10.4230/LIPICS.GD.2024.29.

9 Miriam Goetze, Michael Hoffmann, Ignaz Rutter, and Torsten Ueckerdt. Crossing Number of Simple 3-Plane Drawings. In *33rd International Symposium on Graph Drawing and Network Visualization (GD 2025)*, volume 357 of *LIPIcs*, pages 15:1–15:19, 2025. doi:10.4230/LIPIcs.GD.2025.15.

10 Ross J. Kang and Tobias Müller. Frugal, acyclic and star colourings of graphs. *Discrete Applied Mathematics*, 159(16):1806–1814, 2011. doi:10.1016/j.dam.2010.05.008.

11 Michael Kaufmann, Boris Klemz, Kristin Knorr, Meghana M. Reddy, Felix Schröder, and Torsten Ueckerdt. The Density Formula: One Lemma to Bound Them All. In *32nd International Symposium on Graph Drawing and Network Visualization (GD 2024)*, volume 320 of *LIPIcs*, pages 7:1–7:17, Dagstuhl, Germany, 2024. doi:10.4230/LIPIcs.GD.2024.7.

12 János Pach, Radoš Radoičić, Gábor Tardos, and Géza Tóth. Improving the Crossing Lemma by finding more crossings in sparse graphs. In *20th International Symposium on Computational Geometry (SoCG 2004)*, pages 68–75, 2004. doi:10.1145/997817.997831.

13 János Pach, Radoš Radoičić, Gábor Tardos, and Géza Tóth. Improving the Crossing Lemma by finding more crossings in sparse graphs. *Discrete Computational Geometry*, 36(4):527–552, 2006. doi:10.1007/s00454-006-1264-9.

14 János Pach and Géza Tóth. Graphs drawn with few crossings per edge. In *4th International Symposium on Graph Drawing and Network Visualization (GD 1996)*, volume 1190 of *LNCS*, pages 345–354, 1996. doi:10.1007/3-540-62495-3_59.

15 János Pach and Géza Tóth. Graphs drawn with few crossings per edge. *Combinatorica*, 17(3):427–439, 1997. doi:10.1007/BF01215922.

16 John C. Urschel and Jake Wellens. Testing gap k-planarity is NP-complete. *Information Processing Letters*, 169:106083, 2021. doi:10.1016/j.ipl.2020.106083.

On the Computation of Schrijver's Kernels

Vincent Delecroix[1], Oscar Fontaine[1], and Francis Lazarus[2]

1 **LaBRI, Université de Bordeaux, CNRS, France**
vincent.delecroix@u-bordeaux.fr, oscar.fontaine@u-bordeaux.fr
2 **G-SCOP/Institut Fourier, Université Grenoble Alpes, CNRS, France**
francis.lazarus@grenoble-inp.fr

Abstract

To understand the geometry of a graph G embedded on a closed surface, an interesting tool is the number of intersection of G with any closed curve c. A useful map is the length spectrum $\mu_G(c)$ counting the minimum number of intersections between G and any curve freely homotopic to c. The notion of kernel have been introduced by Schrijver in the 90's. It corresponds to an embedded graph G such that for any proper minor H of G, $\mu_H < \mu_G$. It is quite obvious that any embedded graph G has a minor K which is a kernel and with same length spectrum. Such a minor is called a minor kernel. We describe an algorithm to compute a minor kernel of a given graph in $O(n^3 \log n)$. Thanks to a correspondence with system of curves, this algorithm relies on a tight bound on the length of minimal bigon in system of curves. As a consequence of the computation of minor kernel, we are able to decide whether two graphs minors have the same spectrum and improve the state-of-the-art algorithm to compute $\mu_G(c)$.

Related Version arXiv:2510.18597

1 Introduction

Let G be a graph embedded on a closed surface S. For a closed curve c in S, we let $\mu_G(c)$ count the minimum number of intersections between G and any curve freely homotopic to c. Schrijver [14] calls G a **kernel** if it is minor minimal among all graphs determining the same map $c \mapsto \mu_G(c)$. In other words, contracting or deleting any edge of a kernel G changes $\mu_G(c)$ for at least one curve c.

In a beautiful series of papers, Schrijver [12, 13, 14], and then de Graff and Schrijver [3, 4], studied the maps μ_G and kernels in connection with systems of curves on S. In particular, they established that $\mu_G(c) = \frac{1}{2} \cdot \nu_{\mathcal{C}}(c)$ where $\mathcal{C}$ is a system of closed curves depending on G (but independent of c) and $\nu_{\mathcal{C}}(c)$ counts the minimum number of crossings between $\mathcal{C}$ and any curve freely homotopic to c in general position with respect to $\mathcal{C}$. Here, $\mathcal{C}$ can be viewed as a graph (its subdivision is irrelevant), so that $\nu_{\mathcal{C}}(c)$ can be interpreted as the **crossing number** of the system of closed curves $\mathcal{C}$ and c. This lead Schrijver [14] to make an explicit correspondence between kernels and systems of curves in minimal position in S. From this correspondence it follows that kernels are essentially unique: if G and H are two kernels on S with $\mu_G = \mu_H$, then H can be obtained from G by a series of ΔY-exchanges (see [14] for a precise definition) possibly taking the (surface) dual of the resulting graph.

The uniqueness of kernels is reminiscent of an analogous property stating that the geometry of Riemannian surfaces with negative scalar curvature is fully determined by their **(marked) length spectrum** proven by Otal [11]. Here, the length spectrum associates to every closed curve the infimum of the Riemannian length of any curve in its free homotopy class. The function μ_G associated to an embedded graph G can actually be seen as a length spectrum with respect to a discrete metric. The appropriate geometric notion that captures both the continuous and discrete settings is the definition of *geodesic current* from

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

Bonahon [1]. In accordance with this terminology we will also call μ_G the **μ-spectrum of G**.

For an embedded graph G, we say that H is a **minor kernel** of G if H is a minor of G, a kernel and has the same μ-spectrum as G. The main purpose of this paper is to propose an efficient algorithm that given an embedded graph G computes one of its minor kernels. We assume for all our complexity analyses the standard RAM model of computation with words of logarithmic size with respect to the input. More precisely, we show the following:

▶ **Theorem 1.1.** *Let G be a graph with n edges cellularly embedded in a closed oriented surface S of genus $g \geqslant 2$. A minor kernel of G can be computed in $O(n^3 \log n)$ time.*

We now state two applications of Theorem 1.1.

▶ **Theorem 1.2.** *Let G be a graph with n edges cellularly embedded in a closed oriented surface S of genus $g \geqslant 2$. Let H be a graph minor of G given as a list of minor operations on the edges of G. Then, we can decide whether G and H have the same μ-spectrum in $O(n^3 \log n)$ time.*

A similar but different question is to decide whether $\mu_G = \mu_H$ for two graphs G and H embedded on the same surface S. However doing so requires a way to compute an homotopy equivalence between G and H.

▶ **Theorem 1.3.** *Let G be a graph with n edges cellularly embedded in a closed oriented surface S of genus $g \geqslant 2$. After a preprocessing in $O(n^3 \log n)$ time, we can compute $\mu_G(c)$ for every closed walk c of length ℓ in the dual of G in $O(g(n+\ell)\log(n+\ell))$ time.*

It follows from previous work by Colin de Verdière and Erickson [2] that $\mu_G(c)$ can be computed in $O(gn\ell \log(n\ell))$ time after a precomputation in $O(gn \log ng)$ time and $O(gn)$ space. Our complexity improvement for the computation of $\mu_G(c)$ relies on the precomputation of a minor kernel of G in $O(n^3 \log n)$ time. We then take advantage on Schrijver's correspondence between kernels and system of closed curves together with more recent results for computing crossing numbers of systems of closed curves by Despré and Lazarus [5] as improved by Dubois [6]. Note, however, that [2] applies to a more general setting.

2 Preliminaries

We introduce in this section basic topological and combinatorial definitions that will be used throughout the text. For reference for topology of surfaces, see for example [10]

Let $G = (V, E)$ be an undirected graph with vertex set V and edge set E. We allow G to have loops and multiple edges. A **combinatorial surface** (S, G) is an embedding of G on some closed oriented surface S such that $S \setminus G$ is a collection of disjoint open disks.

A **curve** is a continuous map $c : [0, 1] \to S$. It is **closed** if $c(0) = c(1)$. Let $\mathcal{C} = (c_1, \ldots, c_k)$ be a system of closed curves on an oriented surface S, i.e. a sequence of maps $c_i : \mathbb{R}/\mathbb{Z} \to S$. The system of closed curves $\mathcal{C}$ is in *general position* if every intersection of the curves of $\mathcal{C}$ is crossed transversely by exactly two curves c_i and c_j (possibly $i = j$). A closed curve c is in *general position with respect to* $\mathcal{C}$ if $\mathcal{C} \cup \{c\}$ is in general position. An intersection point of a system of closed curves in general position is called a **crossing** to emphasize that the curves intersect themselves transversely.

A **smoothing** at a vertex v of the system of closed curves $\mathcal{C}$ is the system of closed curves obtained from $\mathcal{C}$ by removing the vertex v and reconnecting the incident strands in one of two ways as in Figure 1.

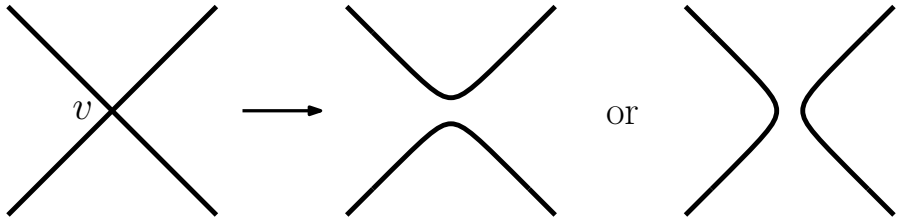


Figure 1 A smoothing at v

Let $\mathcal{C} = (c_1, \dots, c_k)$ be a system of closed curves in S and $\widetilde{\mathcal{C}}$ its preimage in the universal covering of S. A **bigon** of $\widetilde{\mathcal{C}}$ is a pair of two interior disjoint non-trivial compact segments $U \subset \widetilde{c_i^\infty}$ and $D \subset \widetilde{c_j^\infty}$ contained in some lifts of curves c_i and c_j in $\mathcal{C}$ (possibly with $i = j$) so that $U \cup D$ is the boundary of a closed disk in $\widetilde{S}$. A **monogon** of $\widetilde{\mathcal{C}}$ is a non-trivial compact segment of a lift of curve $U \subset \widetilde{c_i^\infty}$ with coincident endpoints bounding a disk in $\widetilde{S}$.

A monogon in $\widetilde{\mathcal{C}}$ is **empty** if the only vertex on its boundary is its corner. A bigon b in $\widetilde{\mathcal{C}}$ is **minimal** if the disk bounded by b does not contain another closed curve, a monogon or a bigon.

3 Length Spectra

There are several natural combinatorial notions of curves in a combinatorial surface S with underlying graph G. These different settings appear in the work of Schrijver [13, 14] under a continuous disguise and give rise to different length notions for curves and length spectra of graphs. In this paper, we define two notions of length spectra, one for graphs and another for system of curves.

Given a graph G, the **μ-spectrum** corresponding to this *vertex metric* is defined as the map

$$\mu_G : \pi_1(S)^{conj} \to \mathbb{N}, \gamma \mapsto \min_{c \in \gamma} \mathrm{cr}(c, G),$$

where c runs over all the curves representatives in the free homotopy class γ.

Given a system of curves $\mathcal{C}$, the length of a curve c is its number of crossings with $\mathcal{C}$ and is denoted $\mathrm{cr}(c, \mathcal{C})$. Thus the **$\nu$-spectrum** of $\mathcal{C}$ is the map

$$\nu_{\mathcal{C}} : \pi_1(S)^{conj} \to \mathbb{N}, \gamma \mapsto \min_{c \in \gamma} \mathrm{cr}(c, \mathcal{C}),$$

where c runs over all the curves representatives in the free homotopy class γ in general position with respect to $\mathcal{C}$. This time c may not pass through vertices of $\mathcal{C}$.

4 Schrijver's Correspondence

Recall that a kernel K on a closed oriented surface S is an embedded graph K such that for every proper minor H of K, there is a closed curve c satisfying $\mu_H([c]) < \mu_K([c])$. If G is an embedded graph, a **minor kernel** K of G is a minor of G which is a kernel and such that $\mu_G = \mu_K$. Moreover, any minor operation on a graph G may be obtained as a smoothing of a vertex of its medial $M(G)$ (See Figure 2).

Schrijver showed in [14] the following correspondence between system of closed curves and graphs.

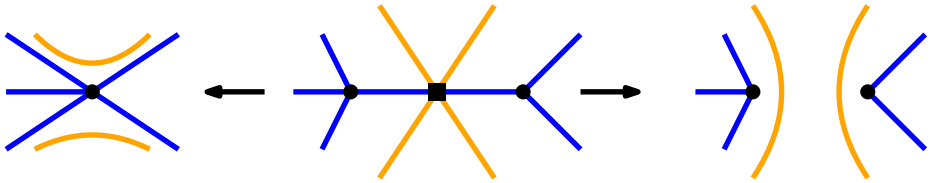


Figure 2 A graph minor operation in G (blue) and the corresponding smoothings in $M(G)$ (orange)

▶ Proposition 4.1. Let G be a graph embedded on a surface S. The following assertions are equivalent:

- G is a kernel,
- $M(G)$ is a tight system of primitive curves.

We thus define the analogue notion of kernel for a system of closed curves. If $\mathcal{C}$ is a system of closed curves on a closed surface S, a **smoothing minimal** system $\mathcal{K}$ for $\mathcal{C}$ is a system of closed curves $\mathcal{K}$ obtained from $\mathcal{C}$ by a sequence of smoothing such that $\mathcal{K}$ is a tight system of primitive curves and $\nu_{\mathcal{K}} = \nu_{\mathcal{C}}$. Proposition 4.1 together with the fact that any minor operation on G corresponds to a smoothing in $M(G)$ allows us to reformulate the computation of a minor kernel of a graph into the computation of a smoothing minimal system of closed curves. Moreover Schrijver caracterized the smoothing in the system of curves that may be made without changing the length spectrum.

▶ Proposition 4.2 (Schrijver [13, Th. 5]). Let $\mathcal{C}$ be a system of closed curves in general position on a closed surface S. Let b be a minimal bigon of $\widetilde{\mathcal{C}}$ and v be the projection on S of one of its corners. Then $\nu_{\mathcal{C}} = \nu_{\mathcal{C}_{v,b}}$.

▶ Proposition 4.3. Let $\mathcal{C}$ be a system of closed curves in general position on a closed surface S. Let m be an empty monogon of $\widetilde{\mathcal{C}}$ and v be the projection of its corner. Then $\nu_{\mathcal{C}} = \nu_{\mathcal{C}_{v,m}}$.

5 Minimal bigon length

The key result in our work is theorem 5.1. It bounds the length of a minimal bigon, i.e. the number of edges on its boundary, in a system of curves $\mathcal{C}$ by its number of crossings. This allow us to look for minimal bigon by an explicit search as described in section 6 to compute a smoothing minimal system.

▶ **Theorem 5.1.** *Let S be a closed orientable surface of genus $g \geqslant 2$. Let $\mathcal{C}$ be a system of closed curves on S. Then the length of every minimal bigon of $\widetilde{\mathcal{C}}$ is at most $8n$ where n is the number of crossings of $\mathcal{C}$.*

6 Minimal bigon search

The proof of Theorem 1.1 essentially relies on the existence of an algorithm to detect minimal bigons in $O(n^2 \log n)$ time. Thanks to Proposition 4.2, we can safely smooth any of the two corners of such bigons without changing the length spectrum. We repeat this operation, also taking care of empty monogons, until there is no more empty monogon or minimal bigon.

Now, the search for minimal bigons proceeds as follows. Thanks to Theorem 5.1, we can restrict the search to bigons of linear length. The boundary of such bigons projects on S to the concatenation of two homotopic subpaths of some curves in $\mathcal{C}$. We thus search among

the curves in $\mathcal{C}$ for pairs of homotopic subpaths of (the same) linear length and with same endpoints. We then check whether their concatenation lifts to a bigon in $\widetilde{S}$, i.e. to a simple closed walk. It remains to return a bigon with minimal area among the selected bigons.

The tests of homotopy and simplicity is an application of the work of Lazarus and Rivaud and then Erickson and Whittelsey [9, 8] that provide linear time contractibility test and linear time free homotopy test for walks in graphs embedded in surfaces. We extend this result by providing an online algorithm for geodesic tightening of walks. This allows us to perform efficient bigon detection.

The last tool is an algorithm to compute efficiently the area of a bigon, i.e. the number of faces circumscribed by the bigon. This area computation is a special case of an algorithm for computing the (signed) area circumscribed by the lift in the universal covering of S of any contractible closed walk on $\mathcal{C}$. The algorithm relies on the standard reduction of embedded graphs to one-vertex and one-face graphs via the tree-co-tree decomposition of Eppstein [7]. For these simpler graphs, we provide a discrete analogue of Stokes' theorem showing that the area we are interested in (a discrete 2-dimensional integral) is equal to a certain sum along the closed walk (a discrete 1-dimensional integral).

References

1 Francis Bonahon. The geometry of Teichmüller space via geodesic currents. *Invent. Math.*, 92(1):139–162, 1988. URL: https://eudml.org/doc/143562, doi:10.1007/BF01393996.

2 Éric Colin de Verdière and Jeff Erickson. Tightening non-simple paths and cycles on surfaces. *SIAM Journal on Computing*, 39(8):3784–3813, 2010.

3 Maurits de Graaf and Alexander Schrijver. Characterizing Homotopy of Systems of Curves on a Compact Surface by Crossing Numbers. *Linear Algebra and its Applications*, 226:519–528, 1995.

4 Maurits de Graaf and Alexander Schrijver. Decomposition of Graphs on Surfaces. *Journal of Combinatorial Theory, Series B*, 70(1):157–165, 1997.

5 Vincent Despré and Francis Lazarus. Computing the geometric intersection number of curves. *Journal of the ACM (JACM)*, 66(6):1–49, 2019.

6 Loïc Dubois. Making Multicurves Cross Minimally on Surfaces. In Timothy Chan, Johannes Fischer, John Iacono, and Grzegorz Herman, editors, *32nd Annual European Symposium on Algorithms (ESA 2024)*, volume 308 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 50:1–50:15, 2024. URL: https://drops.dagstuhl.de/entities/document/10.4230/LIPIcs.ESA.2024.50, doi:10.4230/LIPIcs.ESA.2024.50.

7 David Eppstein. Dynamic generators of topologically embedded graphs. *Proceedings of the Annual ACM-SIAM Symposium on Discrete Algorithms*, 2002. doi:10.1145/644108.644208.

8 Jeff Erickson and Kim Whittelsey. Transforming curves on surfaces redux. In *Proceedings 24th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1646–1655, 2013. URL: https://epubs.siam.org/doi/abs/10.1137/1.9781611973105.118, doi:10.1137/1.9781611973105.118.

9 Francis Lazarus and Julien Rivaud. On the homotopy test on surfaces. In *Proceedings of the 53rd Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 440–449, 2012. URL: http://arxiv.org/abs/1110.4573.

10 William S. Massey. *A Basic Course in Algebraic Topology*. Springer-Verlag, 1991.

11 Jean-Pierre Otal. Le spectre marqué des longueurs des surfaces à courbure négative. *Annals of Mathematics*, 131(1):151–162, 1990.

12 Alexander Schrijver. Homotopy and crossings of systems of curves on a surface. *Linear Algebra and its Applications*, 114-115:157–167, March 1989. Special Issue Dedicated

to Alan J. Hoffman. URL: https://www.sciencedirect.com/science/article/pii/0024379589904576, doi:10.1016/0024-3795(89)90457-6.

13 Alexander Schrijver. Decomposition of graphs on surfaces and a homotopic circulation theorem. *Journal of Combinatorial Theory, Series B*, 51(2):161–210, 1991.

14 Alexander Schrijver. On the uniqueness of kernels. *Journal of Combinatorial Theory, Series B*, 55(1):146–160, 1992.

The Witness Unit Disk Representability Problem*

Carolina Haase[1], Giuseppe Liotta[2], Maarten Löffler[3], Fabrizio Montecchiani[2], Alessandra Tappini[2], and Soeren Terziadis[4]

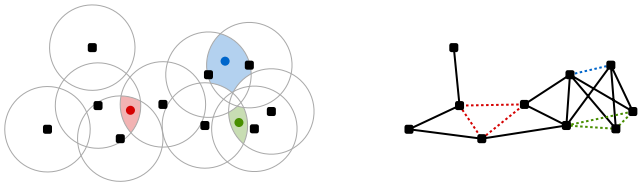
1 **Universität Trier**
m.haasec@uni-trier.de
2 **University of Perugia**
{giuseppe.liotta,fabrizio.montecchiani,alessandra.tappini}@unipg.it
3 **Utrecht University**
m.loffler@uu.nl
4 **TU Munich**
sterziadis@ac.tuwien.ac.at

Abstract

Witness unit disk representations (wUDRs), are unit disk representations, where an additional set of points (witnesses) is given and two disks are considered not adjacent, if their overlap contains a witness. We prove upper and lower bounds on the number of witnesses required to represent graphs of different graph classes and provide an algorithm to compute the graph based on its wUDR.

1 Introduction

Intersection representations of graphs map vertices to copies of geometric objects in the plane. Two such objects intersect if and only if their vertices are connected with an edge. Specifically a unit disk representation (UDR) U of a graph $G = (V, E)$ is a mapping $U : V \mapsto D$, where D is a set of disks, s.t. $uv \in E$ if and only if $U(v) \cap U(u) \neq \emptyset$. If such a mapping exists, we say that G *admits* a UDR.



(a) Example of a wUDR $\mathcal{R}$ **(b)** Implied graph G

Figure 1 A witness unit disk representation with three witnesses and the implied graph G with the prevented dashed edges. The color of a prevented edge matches the witness.

We investigate a representation, which uses an underlying unit disk intersection representation U and an additional set of objects, which prevent edges, which, in our case, is a set W of points (*witnesses*). The graph $G = (V, E')$ corresponding to a *witness unit disk representation* (wUDR) $\mathcal{R} = (U, W)$ is the graph represented by U without any uv where $U(v) \cap U(u)$ contains a witness $w \in W$. We say that w *prevents* the edge uv. See Fig. 1 for an example. We will use $|V| = n$, $|E| = m$, $|E'| = m'$ and $|W| = w$.

* This project was started during a research visit at the University of Perugia.

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

Let $\Psi(G)$ be the set of all wUDR representing a graph G. The *unit witness number* $\mathsf{w}(G)$ (the fewest witnesses any wUDR of G requires) is $\mathsf{w}(G) = \min_{(U,W)\in\Psi(G)} |W|$. For a class $\mathcal{G}$ of graphs, we define $\mathsf{w}(\mathcal{G}) = \max_{G\in\mathcal{G}} \mathsf{w}(G)$. Then $\mathcal{G}_k$ is the class of graphs G, s.t. $\mathsf{w}(G) \leq k$.

Since, for a wUDR $\mathcal{R} = (U, W)$, the restriction of U to any induced subgraph G' of G, together with W, is a wUDR of G', we state the following observation.

▶ Observation 1. For any $i \in \mathbb{N}$, $\mathcal{G}_i$ is (induced-)hereditary.

Intersection representations have been previously researched [6, 7, 8] and disk intersection representations have received special attention: Recognition of such graphs has for several graph classes been proven NP-hard [3, 4] or even $\exists\mathbb{R}$-hard [13]. Other related concepts are relative neighborhood graphs [10], proximity drawings [14] and witness Gabriel graphs [2, 9].

An interesting question is the recovery of a represented graph G based on its witness unit disk representation. In a similar direction, it has been recently shown that, given a unit witness *square* representation, this can be done in an output-sensitive manner in $\tilde{O}(n + m + m')$ [11].

Contribution. We present an algorithm to compute the graph G from a given wUDR $\mathcal{R}$ (Sec. 2) which improves over the runtime of the naïve computation approach and prove several upper bounds on the unit witness number for various graph classes (Sec. 3). Due to space restrictions, full proofs of results marked with ⋆ will be included in the complete version of this paper.

2 Computation of Witness Unit Disk Graphs

The computation of the represented graph G based on its UDR (given by the centers of the disks) is trivial, since G might contain $\mathcal{O}(n^2)$ edges and we can check all disks pairwise for overlap in $\mathcal{O}(n^2)$ time. However, if we do this for a given wUDR and naïvely check containment of every witness we obtain a runtime of $\mathcal{O}(m^2 w)$, which under a reasonable assumption of $w \in O(n^2)$ (i.e., there are no redundant witnesses and each of the $O(n^2)$ regions in the partition of the plane contains at most one witness) is equal to $O(n^4)$. We present a slightly improved approach.

▶ **Theorem 2.1.** *Given a wUDR $\mathcal{R} = (U, W)$, G' can be computed in $\mathcal{O}(\frac{n^3}{\log^{2-\vartheta} n})$ time (for any small sufficiently small constant ϑ) using $O(\frac{n^3}{\log^2 n})$ space.*

Proof. We compute the graph $G = (V, E)$ implied by U. For every pair of overlapping disks we will check if their overlap contains a witness. This is equivalent to querying the set of witness points with lens-shaped query regions. Using a well-known lifting technique, e.g., described by Agarwal [1], d-dimensional ball queries can be reduced to $(d + 1)$-dimensional half-space queries. In our case this means that disk queries are equivalent to 3-dimensional half-space queries and lens-queries are therefore equivalent to 3-dimensional wedge queries.

Matoušek [12] describes a data structure for 3-dimensional half-space range queries, which can be cascaded into a hierarchical data-structure for 3-dimensional wedge queries[1]. It requires $O(\frac{n^3}{\log^2 n})$ space, a preprocessing time of $O(\frac{n^3}{\log^{2-\vartheta} n})$ (for any arbitrary small constant ϑ) and admits queries in $O(\log^2 n)$ time. This means we can build the data structure, then query it $|E| \in O(n^2)$ times, once per edge of G. If the query yields any points contained in the query range, we remove the corresponding edge from E. The remaining edges give us

[1] Matoušek [12], in fact, describes this data-structure for a general dimension d.

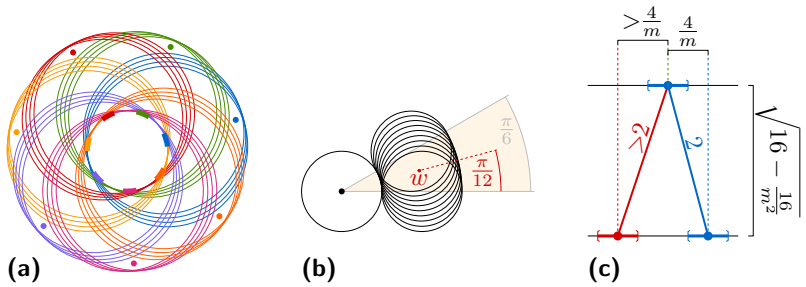


Figure 2 (a) Construction of a wUDR for a complete 7-partite graph. Sets are indicated by color, the centers of disks of a single set are all contained in a small circular arc, highlighted by a bold line. Witnesses are only included in circles of the same color, i.e., belonging to the same set. **(b)** wUDR of a star with degree eight, according to Thm. 3.2. A single witness w is placed. **(c)** Derivation of the vertical distance between two layers of a unit interval bigraph.

the desired graph $G' = (V, E')$. The runtime is dominated by the construction time of the data-structure and the theorem statement follows ◀

3 Bounds on the Unit Witness Number

In this section we provide results of lower and upper bounds on the unit witness number of different graph classes. All proofs of upper bounds are constructive, i.e., we describe how to obtain a wUDR with a bounded number of witnesses for these graph classes.

To begin with, note that there are two ways of representing a non-edge, i.e., two disks which do not overlap and two disks whose overlap includes a witness. In contrast any edge must be represented by two overlapping disks and we make the following easy observation.

▶ Observation 2. All disks of a wUDR (or a UDR) of a graph of diameter κ are necessarily contained in a disk of radius $\frac{\kappa+3}{2}$.

3.1 Graphs of diameter two

We start with the class $\mathcal{P}_k$ of graphs, vertices of which can be partitioned into k independent sets such that all pairs of vertices of different sets are connected (complete k-partite graphs).

▶ **Theorem 3.1.** *For all $k > 1$, $\mathsf{w}(\mathcal{P}_k) = k$.*

Proof. Let G' be a complete k-partite graph with sets $V_1, \ldots, V_k$, where v_i^j is the j-th vertex of set V_i. Then G' admits a wUDR $\mathcal{R} = (U, W)$, with

$$U(v_i^j) = \left(\frac{\cos(\xi)}{2}, \frac{\sin(\xi)}{2}\right) \quad \text{where} \quad \xi = \frac{i\pi}{k} + \frac{j\pi}{2k|V_i|}$$

All disks are placed such that their center is on a circle with radius 0.5. In particular, the vertices of a single set V_i are all placed on a circular arc segment, which does not contain any $v_j \in V_j$, $j \neq i$ (Fig. 2a). Therefore there is always a point, which is contained in all disks of vertices of V_i (and only those), which we use as a witness and we obtain $\mathsf{w}(\mathcal{P}_k) \leq k$.

By Obs. 2, all disks of a wUDR of G' are contained within a disk of radius 3. Now consider the case in which $|V_i| = 6, \forall i \in \{1, \ldots, k\}$. Since packing 5 unit disks into a disk of radius 3 requires at least two such disks to share a point, at least one witness has to be placed in the overlap of at least two disks $U(u)$ and $U(u')$, s.t. $u, u' \in V_j$. Moreover, every one of the

k sets $V_1, \ldots, V_k$ requires such a witness. Let $u \in V_j$ be the vertex whose disk contains a witness w. Since u is adjacent to every $v \in V_l$, $l \neq j$. This again holds for all sets $V_1, \ldots, V_k$. Therefore, we have to place at least one witness per set V_i, $i \in \{1, \ldots, k\}$, i.e., at least k in total. ◀

Note that in the construction of G', where $|V_i| = 6, \forall i \in \{1, \ldots, k\}$, we know that $k = \frac{n}{6}$. Specifically that means that in general for every $k > 1$ there are graphs with $k \in \Omega(n)$ and therefore we have $\mathsf{w}(\mathcal{P}_k) \in \Theta(n)$. This means the witness number of graphs with diameter two is linear in general, however for the very restricted subclass $\mathcal{S}$ of stars we obtain a different bound.

▶ **Theorem 3.2** ($\star$). *$\mathsf{w}(\mathcal{S}) = 1$, and for any star G, $\mathsf{w}(G) = 1$ if and only if G contains seven or more vertices.*

3.2 Unit interval bigraphs

A bipartite graph G is a *unit interval bigraph* if it admits a two-colored unit interval representation. In such a representation every vertex of the first (second) partition is represented by a red (blue) unit interval on the real line. A red and a blue vertex are connected with an edge, if and only if their intervals overlap. There are no edges between two vertices of the same color, even if their intervals overlap. Let $\mathcal{I}_2$ be the class of unit interval bigraphs. Unit interval bigraphs and their relation to many other graph classes have been characterized by Brown and Lundgren [5]. They are, for example, equivalent to permutation graphs. Further, caterpillars are a subclass of unit interval bigraphs.

By providing a construction of a wUDR for any graph $G \in \mathcal{I}_2$, we obtain an upper bound on the witness number of unit interval bigraphs. We further show that this bound is tight, by observing that a complete bipartite graph is in $\mathcal{I}_2$ and requires two witnesses for sufficiently large partitions by Thm. 3.1.

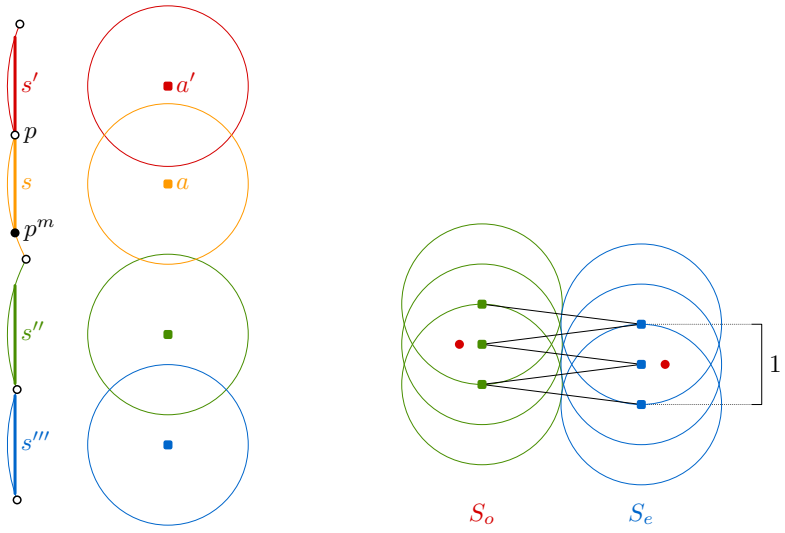
▶ **Theorem 3.3.** *$\mathsf{w}(\mathcal{I}_2) = 2$.*

Proof. Let G be a unit interval bigraph and let $I = X \cup Y$ be the two-colored unit interval representation of the graph G. Assume the leftmost interval starts at 0 and the right-most interval ends at μ. We scale I by $2/\mu$, such that the right-most interval ends at 2. Let c_i be the mid-point of an interval $i \in I$ after scaling. Note that two intervals i, j intersect if $|c_i - c_j| \leq 4/\mu$. For every interval i in X we place a red disk at $(c_i, 0)$. For every interval j in Y we place a blue disk at $\left(c_j, \sqrt{16 - \frac{16}{\mu^2}}\right)$. All red disks pairwise intersect, as do all blue disks. The centers of any red and any blue disk have a vertical distance of $\sqrt{16 - \frac{16}{\mu^2}}$. Therefore, a red and a blue disk intersect if they have a horizontal distance of $4/\mu$ or less, i.e., if their intervals intersect. This value is derived trigonometrically and illustrated in Fig. 2c.

The bound is tight, since a complete bipartite graph whose partitions are at least of size six is a unit interval bigraph requiring two witnesses by Thm. 3.1. ◀

3.3 Caterpillars, lobsters and beyond

Caterpillars are trees, which yield a path (which we will also call the *spine*) after removal of all leaves. They can also be defined as trees with a maximum distance t of any vertex to the spine. For any fixed t this yields *t-limbed* caterpillars (or just *t-caterpillars*) including paths ($t = 0$), caterpillars ($t = 1$) and *lobsters* ($t = 2$). In order to prove an upper bound on the unit witness number of t-caterpillars, we first state the following auxiliary lemma.



(a) Construction of s **(b)** Placement of spine disks

Figure 3 (a) Construction of the segment s (as described in the proof of Thm. 3.4) for a set A of disks based on the disks a and a' with the smallest vertical distance and the copies s', s'' and s''' of s placed to the left of a', a'' and a'''. Note that the construction is stretched vertically for better readability, while in the actual construction all disks of A pairwise overlap. **(b)** The placement of all disks for a spine of length 6 as described in the proof of Thm. 3.5.

▶ **Lemma 3.4.** *Let A be a set of unit disks on the same x-coordinate, i.e., all disks in A share a vertical tangent ℓ on their left. Then for every disk $a \in A$ we can place any set of disks A' to its left, s.t. $A \cap A' = a$ and all disks of A' have the same x-coordinate and therefore share a vertical tangent ℓ', which is parallel to ℓ.*

Proof. If $|A'| = 1$, the lemma follows trivially. Assume $|A'| > 1$ and let a and a' be disks in A such that the vertical distance between the centers of a and a' is minimal over all disks in A (with a' being above a). We calculate the position p of the center of a new disk, which is placed to the left of a, s.t. the distance between p and the center of a (and a') is 2. We mirror p vertically along the horizontal line through c_a to obtain a point p_m. All disks placed with their center on the vertical line segment s between p and p_m (but not on either p nor p_m) intersect only a and have the same x-coordinate, as illustrated in Fig. 3a. Further we can place copies of s to the left of any disk $a'' \in A$, at the same relative position as s to a and retain the same property that any disk placed with its center on such a copy only intersects a''. All newly placed disks share the same x-coordinate and form the set A' with the required properties. ◀

With this lemma established, we state our theorem on the upper bound of the unit witness number of the class of t-caterpillars $\mathcal{C}_t$.

▶ **Theorem 3.5** ($\star$). *For all t, $w(\mathcal{C}_t) \leq 2t + 2$.*

Proof. We will describe the construction of a wUDR for a given t-caterpillar G. Let S be the spine of G. We label the vertices in S from 1 to $|S|$. Let S_e and S_o be the vertices of even and odd label respectively. It is straightforward to place two stacks of disks for S_e and S_o, s.t. the disks i and $i + 1$ intersect, all disks for S_e (S_o) all have the same x-coordinate

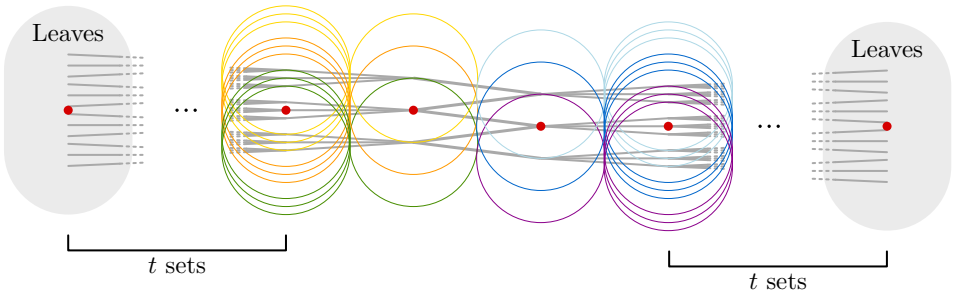


Figure 4 Construction of a witness unit disk representation for a t-caterpillar. The light gray edges indicate adjacencies, the witnesses are shown in red, disks of the first layer on either side have the same color as their parent in the spine for better readability.

and there is a point contained in exactly all disks of S_e (S_o) in which we can place a witness (see also Fig. 3b). A detailed placement is described in **??**. Note, there is a single vertical tangent shared between all disks of S_e (on their left) and a parallel vertical tangent shared between all disks of S_o on their right.

Now we can repeatedly apply Thm. 3.4 to place all disks of the subtrees rooted at the children of each even-indexed spine vertex to the left (in total we apply this lemma t times). This yields the construction is shown in Fig. 4, concluding the proof. ◀

1-caterpillars are simply caterpillars and Thm. 3.5 would yield a wUDR with 4 witnesses. However, since caterpillars are a subclass of $\mathcal{I}_2$ we already know from Thm. 3.3 that caterpillars can be realized with 2 witnesses, so it is unclear if Thm. 3.5 is tight for $t \geq 2$.

3.4 Realizing trees with witnesses proportional to the diameter

Finally, for the class of trees we present an upper bound of h for the witness number, where h is the smallest number of layers in a tree without the root over all possible roots (or equivalently the radius of the tree). Intuitively the upper bound is obtained by using just one side of the construction in Fig. 4 with the spine replaced with a single disk representing the root.

▶ **Theorem 3.6** ($\star$). *For any tree T it holds that $\mathsf{w}(T) \leq h$, where h is the radius of T.*

Sketch. Since a tree of radius h is an h-caterpillar, the proof is very similar to the proof of Thm. 3.5. We restrict the construction to just one side and observe that the "spine" of the tree consists only of the root, which does not require a witness. This yields the bound. ◀

A self-contained proof of Thm. 3.6 and a matching lower bound proof can be found in **??**.

4 Open Problems

Our results suggest the following questions. (i) Can witness disk graphs be computed in $\tilde{O}(m+n+t)$ time, where t is the number of edges in the graph? (ii) For a given k, what is the complexity of deciding if a graph has unit witness number smaller or equal to k?

References

1 Pankaj K. Agarwal. *Simplex Range Searching and Its Variants: A Review*, pages 1–30. Springer International Publishing, Cham, 2017. doi:10.1007/978-3-319-44479-6_1.

2 Boris Aronov, Muriel Dulieu, and Ferran Hurtado. Witness gabriel graphs. *Comput. Geom.*, 46(7):894–908, 2013. doi:https://doi.org/10.1016/j.comgeo.2011.06.004.

3 Sujoy Bhore, Maarten Löffler, Soeren Nickel, and Martin Nöllenburg. Unit disk representations of embedded trees, outerplanar and multi-legged graphs. In Helen C. Purchase and Ignaz Rutter, editors, *Graph Drawing and Network Visualization, GD 2021*, pages 304–317. Springer, 2021. doi:10.1007/978-3-030-92931-2_22.

4 Heinz Breu and David G. Kirkpatrick. Unit disk graph recognition is NP-hard. *Comput. Geom.*, 9(1–2):3–24, 1998. doi:10.1016/S0925-7721(97)00014-X.

5 David E Brown and J Richard Lundgren. Characterizations for unit interval bigraphs. *Congressus Numerantium*, 206:5–17, 2010.

6 Jérémie Chalopin, Daniel Gonçalves, and Pascal Ochem. Planar graphs have 1-string representations. *Discret. Comput. Geom.*, 43(3):626–647, 2010. doi:10.1007/s00454-009-9196-9.

7 Stefan Felsner. Rectangle and square representations of planar graphs. In János Pach, editor, *Thirty Essays on Geometric Graph Theory*, pages 213–248. Springer, 2013. doi:10.1007/978-1-4614-0110-0_12.

8 Daniel Gonçalves, Lucas Isenmann, and Claire Pennarun. Planar graphs as L-intersection or L-contact graphs. In Artur Czumaj, editor, *Symposium on Discrete Algorithms, SODA 2022*, pages 172–184. SIAM, 2018. doi:10.1137/1.9781611975031.12.

9 Carolina Haase, Philipp Kindermann, William J. Lenhart, and Giuseppe Liotta. Separability of witness gabriel drawings. In *33rd International Symposium on Graph Drawing and Network Visualization, GD 2025, Norrköping, Sweden, September 24-26, 2025*, volume 357 of *LIPIcs*, pages 13:1–13:18, 2025. doi:10.4230/LIPICS.GD.2025.13.

10 Jerzy W. Jaromczyk and Godfried T. Toussaint. Relative neighborhood graphs and their relatives. *Proc. IEEE*, 80(9):1502–1517, 1992. doi:10.1109/5.163414.

11 Maarten Löffler, Frank Staals, and Soeren Terziadis. Recovering graphs from their witness unit square representation (poster abstract). In Vida Dujmovic and Fabrizio Montecchiani, editors, *33rd International Symposium on Graph Drawing and Network Visualization, GD 2025, Norrköping, Sweden, September 24-26, 2025*, volume 357 of *LIPIcs*, pages 44:1–44:4. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2025. URL: https://doi.org/10.4230/LIPIcs.GD.2025.44, doi:10.4230/LIPICS.GD.2025.44.

12 Jiří Matoušek. Range searching with efficient hierarchical cuttings. SCG '92, page 276–285, New York, NY, USA, 1992. Association for Computing Machinery. doi:10.1145/142675.142732.

13 Colin McDiarmid and Tobias Müller. Integer realizations of disk and segment graphs. *J. Comb. Theory Ser. B*, 103(1):114–143, jan 2013. doi:10.1016/j.jctb.2012.09.004.

14 Roberto Tamassia, editor. *Handbook on Graph Drawing and Visualization*. Chapman and Hall/CRC, 2013. URL: https://www.crcpress.com/Handbook-of-Graph-Drawing-and-Visualization/Tamassia/9781584884125.

Guarding Offices with Maximum Dispersion*

Sándor P. Fekete[1], Kai Kobbe[1,2], Dominik Krupke[1], Joseph S. B. Mitchell[3], Christian Rieck[4], and Christian Scheffer[5]

1 **Department of Computer Science, TU Braunschweig**
s.fekete@tu-bs.de, krupke@ibr.cs.tu-bs.de
2 **Institute of Geometry, Graz University of Technology**
kai.kobbe@tugraz.at
3 **Department of Applied Mathematics and Statistics, Stony Brook University**
joseph.mitchell@stonybrook.edu
4 **Institute of Mathematics, University of Kassel**
christian.rieck@mathematik.uni-kassel.de
5 **Department of Electrical Engineering and Computer Science, Bochum University of Applied Sciences**
christian.scheffer@hs-bochum.de

Abstract

We investigate the Dispersive Art Gallery Problem with vertex guards for a class of orthogonal polygons that reflect the properties of real-world floor plans: These *office-like* polygons consist of rectangular rooms and corridors. In the dispersive variant of the Art Gallery Problem, the objective is not to minimize the number of guards but to maximize the minimum distance between any pair of guards, called the *dispersion distance*.

We prove that determining whether a guard set can achieve a dispersion distance of 4 in office-like polygons, where vertices are restricted to integer coordinates, is NP-complete. Complementarily, we present a simple worst-case optimal algorithm that guarantees a dispersion distance of 3. For the more restricted class of hole-free independent office-like polygons, we propose a dynamic programming approach that computes optimal solutions in polynomial time.

1 Introduction

The Art Gallery Problem is a widely studied problem in computational geometry: Given a polygonal region, the objective is to select a minimum number of points (*guards*) such that every point within the region is visible from at least one guard. Since its introduction by Victor Klee in 1973, numerous variants of the problem have been studied [9, 11, 12].

An important application involves deploying sensors across a designated area, where a key challenge is balancing sensor spacing with coverage requirements. While increasing the distance between sensors reduces mutual interference and helps preserve data integrity, the network must remain sufficiently dense to ensure complete coverage and continuous, reliable signal transmission throughout the region. This trade-off is captured by the Dispersive Art Gallery Problem, which is defined as follows: Given a polygon $\mathcal{P}$ and a rational number ℓ, determine whether there exists a guard set $\mathcal{G}$ for $\mathcal{P}$ such that the geodesic distance between any pair of guards in $\mathcal{G}$ is at least ℓ. Notably, this formulation focuses solely on pairwise distances between guards rather than minimizing their total number.

* An extended abstract of this paper appeared in the proceedings of the 50th International Symposium on Mathematical Foundations of Computer Science (MFCS 2025) DOI: 10.4230/LIPIcs.MFCS.2025.46. This work was partially supported by the German Research Foundation (DFG), grants 444569951 and 522790373. Moreover, it was supported by the National Science Foundation (CCF-2007275), and by the Austrian Science Fund (FWF) 10.55776/DOC183.

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

In this paper, we study the tractability of the problem in the setting of vertex guards for a class of orthogonal polygons that capture essential structural characteristics of real-world floor plans and rectangular galleries [4]. These polygons, referred to as *office-like polygons*[1], consist of orthogonal rooms and corridors; see Figure 1 for an illustrative example.

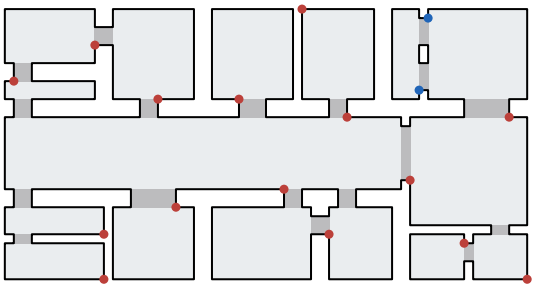


Figure 1 An example of an office-like polygon with three holes, together with a guard set of maximum dispersion. The blue guards have minimum distance among all pairs of guards.

1.1 Our Contributions

We summarize our main results for the Dispersive AGP with vertex guards in office-like polygons under r-visibility and L_1-geodesics:

- We give a worst-case optimal method showing that office-like polygons with integer vertices always admit a guard set with dispersion distance 3, which is sometimes necessary.
- We show that deciding the existence of a guard set with a dispersion distance of 4 is NP-complete, even for integer-coordinate polygons, as well as polyominoes.
- We present a dynamic programming algorithm for independent office-like polygons without holes to efficiently compute guard sets with maximum dispersion in polynomial time.

Due to space restrictions, proofs and technical details can be found in the full version [5].

1.2 Previous work

Prior work studied the Dispersive AGP for vertex guards in polyominoes under L_1-geodesic distance and r-visibility [10]. The problem of deciding whether a thin polyomino admits dispersion distance 5 is NP-complete, whereas simple polyominoes admit worst-case optimal guard sets with dispersion distance 3 computable in linear time. These results were later extended to polygons [6]: deciding dispersion distance 2 is NP-complete for polygons with holes, and even polygons without holes may require dispersion distance 2, which can be achieved using a constructive algorithm.

The classic AGP in office-like polygons was studied in [4]. For hole-free office-like polygons, a greedy algorithm computes guard sets of minimum cardinality, polynomial-time algorithms based on matching and flow techniques exist for r-independent instances, and the general problem under r-visibility is NP-hard. More broadly, the AGP is $\exists\mathbb{R}$-complete under classic line visibility [1], remains NP-hard for r-visibility [8], and becomes polynomial-time solvable for hole-free orthogonal polygons under r-visibility [13].

[1] This class of polygons was originally introduced by Cruz and Tomás [4] using the Portuguese expression SCOT for *salas e corredores ortogonais*, i.e., orthogonal rooms and corridors.

1.3 Preliminaries

Given any polygon $\mathcal{P}$, a subset $\mathcal{G} \subset \mathcal{P}$ of points is called a *guard set* for $\mathcal{P}$ if every point $p \in \mathcal{P}$ is *visible* to at least one point $g \in \mathcal{G}$. We restrict ourselves to vertex guards, i.e., guards are only placed on vertices of $\mathcal{P}$. Moreover, we adopt the *r-visibility* model, where two points p and q are said to *see each other* if the axis-aligned rectangle defined by p and q is entirely contained within $\mathcal{P}$. We measure the *distance* between two points in $\mathcal{P}$ as L_1-*geodesics*. With this, the *dispersion distance* ℓ of a guard set $\mathcal{G}$ is then defined as the minimum distance between any pair of guards.

An *office-like polygon* $\mathcal{P} = (\mathcal{R}, \mathcal{C})$ is an orthogonal polygon made up from a collection of rectangular rooms $\mathcal{R}$ that are linked by a collection of rectangular corridors $\mathcal{C}$ such that all rectangles in $\mathcal{R} \cup \mathcal{C}$ have pairwise disjoint interiors. A corridor connects two rooms if two opposite edges of the corridor are *strictly contained* in edges of the two rooms, one in each room; in particular, the corridor is *narrower* than the rooms, and no vertex of a room coincides with a vertex of a corridor. An example of an office-like polygon with several rooms and corridors, as well as a set of guards is depicted in Figure 1. Two corridors C_1 and C_2 are *independent* if no guard placed on a vertex of C_1 also guards C_2. If every pair of corridors is independent, then the polygon is *independent*.

2 Worst-case optimality

In this section, we present a worst-case optimal algorithm for the Dispersive AGP in office-like polygons with vertices at integer coordinates.

▶ **Theorem 2.1.** *For office-like polygons with vertices at integer coordinates, there always exists a guard set with dispersion distance at least* 3*, which is sometimes optimal.*

As an initial observation, one can easily construct instances in which every feasible guard set has a dispersion distance of at most 3; such an instance is illustrated in Figure 2(a). In this polygon, at least one guard is required in each of the three unit-square corridors, which necessarily forces at least one pair of guards to be at distance at most 3 from one another. Importantly, this polygon can be used as a building block, as it can be extended arbitrarily in the directions indicated by the arrows.

We now present the high-level idea of an algorithm that constructs guard sets with a dispersion distance of 3, which is therefore optimal in the worst case. The algorithm works in three phases. During each phase, a set of guards is placed while maintaining the invariant that every newly placed guard is at distance at least 3 from all guards placed earlier.

Phase (1): Place guards at corridor vertices for all corridors that connect two rooms vertically, ensuring full coverage of these corridors.

Phase (2): Apply the same procedure to corridors that connect two rooms horizontally.

Phase (3): Place guards in any rooms that remain uncovered after the previous phases.

During the initial two phases, guards are placed along the left boundaries of vertical corridors and along the bottom boundaries of horizontal corridors. As a result, any room that remains uncovered after these two phases must contain at least one vertex whose distance to every guard placed in an incident corridor is at least three. Figure 2(b) illustrates the guard placements computed by our algorithm. Guards placed during the first phase are shown in red, those placed in the second phase in blue, and those placed in the final phase in green.

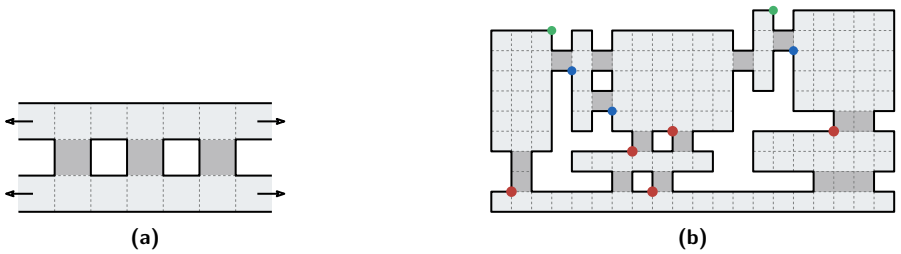


Figure 2 A dispersion distance of 3 is (a) sometimes best possible and (b) always realizable.

3 Computational complexity

We investigate the computational complexity of the respective decision problem, and show:

▶ **Theorem 3.1.** *Deciding whether there exists a guard set with a dispersion distance of* 4 *in independent office-like polygons with vertices at integer coordinates is* NP*-complete.*

Membership in NP is straightforward; thus, we focus on the NP-hardness reduction. We reduce from Planar 3Sat with Exactly Three Occurrences per Variable [2, 3, 7], where each variable appears twice negated and once unnegated. Given such a Boolean formula φ, we compute a rectilinear embedding of its clause-variable incidence graph and straightforwardly replace variables and clauses by the corresponding gadgets.

Variable gadget. The variable gadget, shown in Figure 3, admits exactly two feasible guard sets with a dispersion distance of 4, corresponding to the variable being assigned `true` (red vertices) or `false` (blue vertices). Crucially, no single guard set can simultaneously observe both the corridor labeled `true` and the corridor labeled `false`.

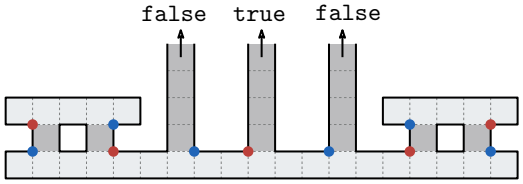


Figure 3 Variable gadget for office-like polygons with vertices at integer coordinates.

Clause gadget. Different clause gadgets accommodate both two- and three-literal clauses, as illustrated in Figure 4. In each gadget, guard sets with dispersion distance 4 cover all but one incoming corridor, and no such set can cover all corridors simultaneously; vertices with the same color indicate membership in the same guard set. Consequently, at least one corridor must be guarded from outside the gadget, ensuring that the corresponding clause gadget is satisfied under the associated truth assignment.

Bending gadget. The bending gadget, depicted in Figure 5, implements a 90° edge-bend. It effectively propagates an assignment across the turn: if the incoming corridor is guarded from outside the gadget, the outgoing corridor can be guarded from inside (blue vertices), and vice versa (red vertices). Importantly, no guard set with dispersion distance 4 can simultaneously cover both corridors from within it.

Figure 4 Clause gadgets for office-like polygons with vertices at integer coordinates.

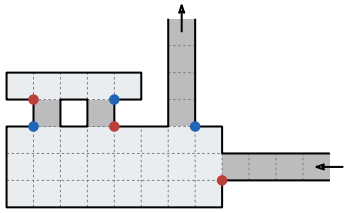


Figure 5 The bending gadget enables 90° turns within office-like polygons.

These gadgets enable us to prove that a Boolean formula is satisfiable if and only if the constructed polygonal region admits a guard set with dispersion distance 4, thereby establishing NP-hardness. Moreover, as illustrated in the figures, each constructed polygonal region is in fact a polyomino. Consequently, the reduction applies directly to polyominoes as well, and the result holds even under this restriction.

4 DP for hole-free independent office-like polygons

We now present a polynomial-time dynamic programming algorithm that computes the optimal solution for independent office-like polygons without holes. The main component of this approach is to solve geometric independent set problems.

▶ **Theorem 4.1.** *For independent office-like polygons without holes, there exists a polynomial-time algorithm that computes guard sets with maximum dispersion.*

The high-level idea of our approach is as follows: Consider a hole-free independent office-like polygon $\mathcal{P}$. We solve the decision problem of whether there exists a guard set with dispersion distance at least ℓ, and then perform a binary search over the $\mathcal{O}(n^2)$ candidate values implied by all pairwise vertex distances in $\mathcal{P}$ to solve the maximization problem.

For the decision problem, we associate with $\mathcal{P}$ a tree $G(\mathcal{P}) = (V, E)$ defined as follows: There is a node $v \in V$ for each room R_v of $\mathcal{P}$, and an edge $e = \{v, w\} \in E$ exactly when the corresponding rooms R_v and R_w are connected by a corridor C_e. We fix an arbitrary in-arborescence G' of $G(\mathcal{P})$ with all edges directed toward a root, see Figure 6(a).

For each node v, we define a corresponding subproblem, which is solved only after all predecessor nodes have been marked as processed, thereby enabling a bottom-up construction of the solution. To process the subproblem at node v, we remove the room R_v and, if present, the corridor C_a corresponding to the outgoing arc a, thereby partitioning $\mathcal{P}$ into several subpolygons, see Figure 6(b). We slightly abuse notation by letting C_a refer to the corridor associated with the underlying edge of a.

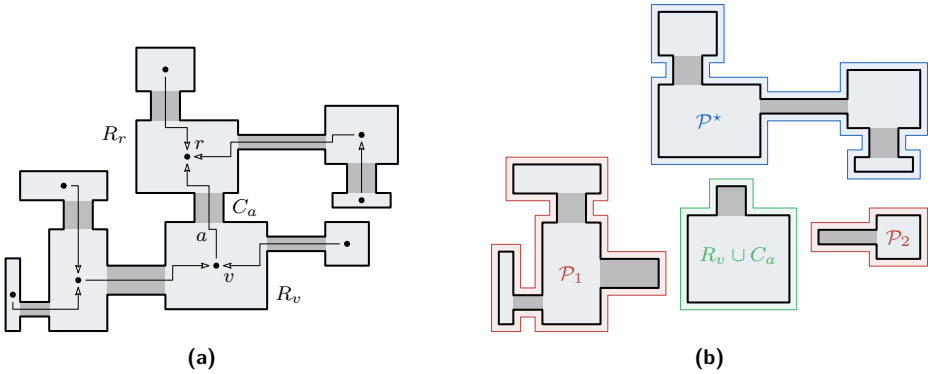


Figure 6 A hole-free independent office-like polygon $\mathcal{P}$ with (a) an in-arborescence of $G(\mathcal{P})$ and (b) the subpolygons considered for the subproblem at node v. Feasible guard sets for $\mathcal{P}_1$ and $\mathcal{P}_2$ are already available, guard placements in $R_v \cup C_a$ are enumerated, and $\mathcal{P}^\star$ is ignored.

Let $\mathcal{P}^\star \subseteq \mathcal{P}$ denote the subpolygon that contains the room R_r of the root r of G'. By convention, if $v = r$, we set $\mathcal{P}^\star = \emptyset$. Our objective is to compute a constant number of *candidate* guard sets for $\mathcal{P} \setminus \mathcal{P}^\star$ with the following property: if there exists a guard set for $\mathcal{P}$ with dispersion distance ℓ, then such a guard set can be obtained by placing guards at vertices of $\mathcal{P}^\star$ and choosing one of these candidate sets for $\mathcal{P} \setminus \mathcal{P}^\star$. The fact that a constant number of candidates suffices without knowing $\mathcal{P}^\star$ is argued using the independence of $\mathcal{P}$.

At this stage, for each subpolygon created by the removal of R_v and C_a, except for $\mathcal{P}^\star$, we have already recursively computed a constant number of candidate guard sets. Hence, it remains to select exactly one guard set from each of these subpolygons and to combine them with one of the constantly many possible guard placements in $R_v \cup C_a$, obtained by explicitly testing all feasible options, so as to construct a valid guard set for $\mathcal{P} \setminus \mathcal{P}^\star$.

This selection task can be formulated as a geometric independent set problem: from each subpolygon, select exactly one guard set such that the pairwise distance between guards from different subpolygons is at least ℓ. We address this through a sequence of greedy selection procedures. Embedding this decision routine as a subroutine within the above-mentioned binary search framework yields a guard set with maximum dispersion.

5 Conclusions

We conducted a comprehensive study of the Dispersive AGP in office-like polygons and obtained several results. Most notably, we proved that deciding whether a dispersion distance of 4 can be achieved in office-like polygons with holes and vertices at integer coordinates is NP-complete. This hardness result straightforwardly extends to polyominoes.

Complementing this, we established a tight bound by showing that a guard set with dispersion distance 3 always exists and can be computed by a simple algorithm. Moreover, there exist office-like polygons in which a dispersion distance of 3 is necessary, making the algorithm worst-case optimal.

For independent office-like polygons without holes, we developed a dynamic programming algorithm that computes an optimal solution in polynomial time. Whether this approach can be extended to non-independent polygons remains an open problem.

Finally, the variant involving point guards remains widely open.

References

1 Mikkel Abrahamsen, Anna Adamaszek, and Tillmann Miltzow. The art gallery problem is $\exists\mathbb{R}$-complete. *Journal of the ACM*, 69(1):4:1–4:70, 2022. doi:10.1145/3486220.

2 Piotr Berman, Marek Karpinski, and Alex D. Scott. Approximation hardness and satisfiability of bounded occurrence instances of SAT. Technical Report TR03-022, Electronic Colloquium on Computational Complexity (ECCC), 2003. URL: https://eccc.weizmann.ac.il/report/2003/022/.

3 Márcia R. Cerioli, Luérbio Faria, Talita O. Ferreira, Carlos A. J. Martinhon, Fábio Protti, and Bruce A. Reed. Partition into cliques for cubic graphs: Planar case, complexity and approximation. *Discrete Applied Mathematics*, 156(12):2270–2278, 2008. doi:10.1016/J.DAM.2007.10.015.

4 Vasco Cruz and Ana Paula Tomás. On r-guarding SCOTs - A new family of orthogonal polygons. In *Latin American Symposium on Theoretical Informatics (LATIN)*, pages 713–729, 2022. doi:10.1007/978-3-031-20624-5_43.

5 Sándor P. Fekete, Kai Kobbe, Dominik Krupke, Joseph S. B. Mitchell, Christian Rieck, and Christian Scheffer. Guarding offices with maximum dispersion. In *International Symposium on Mathematical Foundations of Computer Science (MFCS)*, pages 46:1–46:17, 2025. doi:10.4230/LIPICS.MFCS.2025.46.

6 Sándor P. Fekete, Joseph S. B. Mitchell, Christian Rieck, Christian Scheffer, and Christiane Schmidt. Dispersive vertex guarding for simple and non-simple polygons. In *Canadian Conference on Computational Geometry (CCCG)*, pages 33–40, 2024. URL: https://cosc.brocku.ca/~rnishat/CCCG_2024_proceedings.pdf#section.0.4.

7 M. R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, 1979.

8 Chuzo Iwamoto and Toshihiko Kume. Computational complexity of the r-visibility guard set problem for polyominoes. In *Discrete and Computational Geometry and Graphs*, pages 87–95, 2014. doi:10.1007/978-3-319-13287-7_8.

9 Joseph O'Rourke. *Art Gallery Theorems and Algorithms*. Oxford University Press, 1987.

10 Christian Rieck and Christian Scheffer. The dispersive art gallery problem. *Computational Geometry*, 117:102054, 2024. doi:10.1016/j.comgeo.2023.102054.

11 Thomas C. Shermer. Recent results in art galleries (geometry). *Proceedings of the IEEE*, 80(9):1384–1399, 1992. doi:10.1109/5.163407.

12 Jorge Urrutia. Art gallery and illumination problems. In *Handbook of Computational Geometry*, pages 973–1027. 2000. doi:10.1016/b978-044482537-7/50023-1.

13 Chris Worman and J. Mark Keil. Polygon decomposition and the orthogonal art gallery problem. *International Journal of Computational Geometry & Applications*, 17(02):105–138, 2007. doi:10.1142/S0218195907002264.

Sliding Cubes in Parallel*

Hugo A. Akitaya[1], Joseph Dorfer[2], Peter Kramer[3], Christian Rieck[4], Gabriel Shahrouzi[1], and Frederick Stock[1]

1 **Miner School of Computer and Information Sciences, University of Massachusetts Lowell**
`hugo_akitaya@uml.edu`, `gabriel_shahrouzi@student.uml.edu`, `frederick_stock@student.uml.edu`

2 **Institute of Algorithms and Theory, Graz University of Technology**
`joseph.dorfer@tugraz.at`

3 **Department of Computer Science, TU Braunschweig**
`kramer@ibr.cs.tu-bs.de`

4 **Institute of Mathematics, University of Kassel**
`christian.rieck@mathematik.uni-kassel.de`

Abstract

We study the classic ***sliding cube model*** for programmable matter under ***parallel*** reconfiguration in three dimensions. This problem asks for discrete reconfiguration sequences between two connected configurations of n indistinguishable cube ***modules*** under connectivity constraints; the ***makespan*** of such a sequence is the number of parallel operations performed.

We show that deciding the existence of such a sequence is NP-hard even for constant makespan and if the two input configurations have constant-size symmetric difference, solving an open question in [4, ESA '25]. We also show log-APX-hardness of the problem in sequential and parallel models, strengthening the APX-hardness claim in [2, SWAT '22]. Finally, we give an asymptotically worst-case optimal input-sensitive algorithm for reconfiguration, where the makespan, which is $\mathcal{O}(n)$ in the worst case, depends on the bounding box of the input.

Related Version with all proofs and details: `arXiv:2603.08537`

1 Introduction

Programmable matter systems, composed of large numbers of simple modules capable of self-reconfiguration, have attracted significant attention for applications in robotics, manufacturing, and distributed computing. A fundamental algorithmic challenge is to transform one connected configuration of modules into another while preserving connectivity and avoiding collisions; this problem has been studied extensively [1, 2, 4, 5, 8, 9, 11, 12, 13, 14]. We consider the Sliding Cubes model introduced by Fitch, Butler, and Rus [10], where identical modules move via discrete moves on a hyper-cube lattice, as shown in Figure 1.

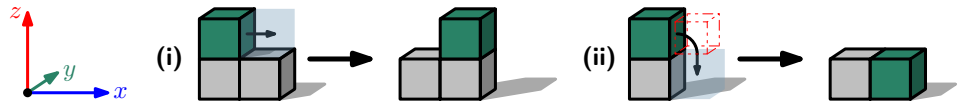


Figure 1 Two types of legal move, (i) slide and (ii) convex transition.

* Research was funded by the National Science Foundation (NSF), grant CCF-2348067, the Austrian Science Fund (FWF) grant 10.55776/DOC183, a fellowship of the German Academic Exchange Service (DAAD), and the Deutsche Forschungsgemeinschaft (DFG) grant 522790373.

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

As giving a detailed overview of related work here is impractical, we refer to recent papers instead and give a limited overview of the most closely related results. Early work focused on Sequential models in two or three dimensions, where only one module moves at a time. As a notable complexity result, the authors of [2] show that the decision version of this reconfiguration problem is NP-complete for pairs of configurations with symmetric difference linear in the number of variables in a given 3Sat instance. They claim, but do not prove, that their reduction can be adapted to show APX-hardness for three dimensions. A common algorithmic approach has been the formation of ***compact*** intermediate configurations, which are densely packed and monotone along all axes. The authors of [2] achieve input-sensitive compaction in two dimensions in $\mathcal{O}(nP)$ moves for n modules, where P is the bounding box perimeter. In [12], compaction is performed in moves proportional to the sum of coordinates of the cubes in the input, which is optimal. For three dimensions, universal reconfiguration was first shown in [1], the bounds on the number of necessary moves were later improved to $\mathcal{O}(n^2)$ with the aid of additional coauthors.

Recently, the authors of [4] introduced a two-dimensional Parallel Sliding Squares model, showing that this can dramatically accelerate reconfiguration by achieving worst-case optimal makespan $\mathcal{O}(P)$. They prove it is NP-complete to decide whether reconfiguration can be completed in one step, ruling out an FPT algorithm parameterized by the makespan and ask whether the problem remains hard if the input has constant symmetric difference.

Our contributions. We extend the parallel model from [4] to three dimensions, proving the following. We show that (1) it is NP-hard to distinguish between optimal makespan 1 and 2, even if configurations differ only by one module. (2) Finding the minimum makespan in both Parallel and Sequential Sliding Cubes is log-APX-hard, i.e., cannot be approximated by a factor better than $\Theta(\log n)$. (3) Schedules of makespan $\mathcal{O}(A + h)$, where A is the area of the projection of the configuration onto the base of its bounding box and h is the height of the bounding box, can be computed efficiently. This bound is $\mathcal{O}(n)$ in the worst-case.

2 Preliminaries

Our work considers the coordinated motion of unit cubes in the three-dimensional ***integer grid***, following and extending the notation of [4]. For $v \in \mathbb{Z}^3$, let $x(v), y(v), z(v) \in \mathbb{Z}$ refer to its coordinates along each of the three axes. The three axes point *east*, *north*, and *up* in that order. A ***cell*** in our setting is a cube $\mathcal{Q}_3$, anchored at its minimal coordinates. Two cells are then ***face-***, ***edge-***, or ***vertex-adjacent*** if they share a common face, edge, or vertex.

Configurations. A ***configuration*** C of n cube ***modules*** is defined as a set of n cells that are occupied by one module each, and valid exactly if their face-adjacency relation induces a connected graph. The ***projection*** $\lambda[C]$ of C along the λ-axis with ***area*** $|\lambda[C]|$ is

$$\lambda[C] = \{(\rho(\mathsf{m}), \phi(\mathsf{m})) \in \mathbb{Z}^2 \mid \mathsf{m} \in C, \{\rho, \phi\} = \{x, y, z\} \setminus \{\lambda\}\}.$$

The ***width***, ***depth***, and ***height*** correspond to measurements in the x, y, and z dimensions.

Moves. Modules can perform two types of ***move***, a ***slide*** or a ***convex transition***. A slide translates the module by one unit along the surface of two adjacent modules, while a convex transition displaces it along the surface of only one module along two axes. As in the two-dimensional model [4], both take unit time to complete and occur at a fixed, constant speed.

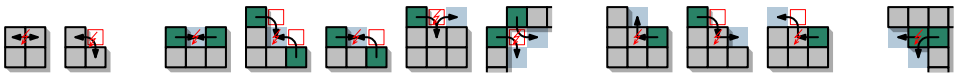


Figure 2 Examples of collisions. Figure from [4], used with the authors' permission.

Transformations. Parallel moves are performed in discrete ***transformations*** with unit-time duration. These must preserve a connected ***backbone*** which requires moving modules to be ***free***: moving modules do not provide connectivity. A set of modules M is free in a configuration C if the removal of any subset of M from C leaves a connected configuration.

In addition, transformations must not induce ***collisions***, in which the volume of two modules overlap. For instance, modules thus cannot trade places during a transformation or perform a convex transition through the same cell. We refer to [4] and Figure 2 for details.

Problem statement. The Parallel Sliding Cubes problem gives two configurations of n modules each and an integer $k \in \mathbb{N}^+$ and asks for a sequence of at most k transformations that obtains one from the other. Such a sequence is then called a ***schedule*** of ***makespan*** k.

3 Computational complexity

We provide several complexity results, starting with a negative answer to a question in [4], which asks whether there exists an algorithm of runtime $f(\delta) \cdot n^{\mathcal{O}(1)}$ for some function f of the symmetric difference size δ that computes optimal reconfiguration sequences.

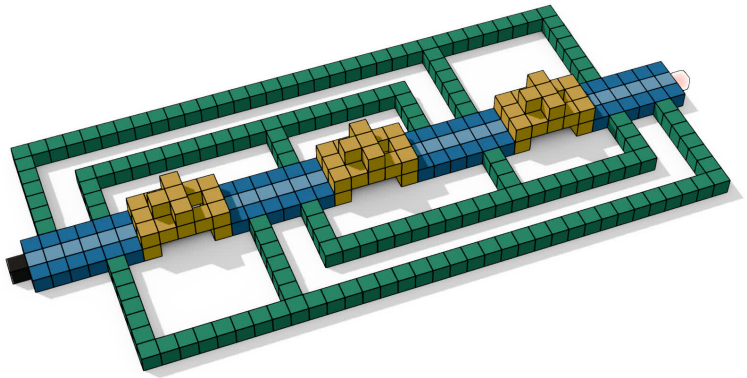
▶ **Theorem 3.1.** *Parallel Sliding Cubes is* para-NP*-complete parameterized by makespan and symmetric difference size. That is, it is* NP*-complete even if both values are constant.*

We reduce from a variant of 3Sat, which asks whether a given Boolean formula φ is satisfiable [6]. For our reduction, we define two configurations, illustrated in Figure 3, based on ***gadgets*** that represent variables (blue) and clauses (green) of φ that are held together by connectors (yellow) and differ by a single module (black/transparent) at either end. To solve the instance in one transformation, an uninterrupted path of modules must move between the two without cutting the configuration apart. It passes through each variable gadget on either the positive or negative side, breaking connectivity to incident clauses and encoding a Boolean assignment. As the instance can be solved in makespan two without deciding satisfiability, there is no polynomial-time $(2-\varepsilon)$-approximation with $\varepsilon > 0$, unless P = NP.

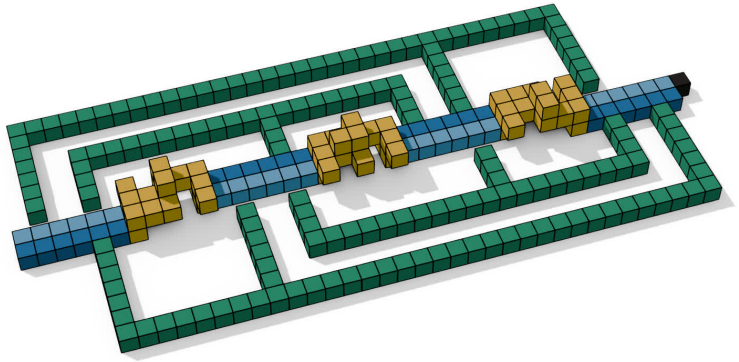
▶ **Theorem 3.2.** *Parallel and Sequential Sliding Cubes are* log-APX*-hard and thus cannot be approximated within a factor better than* $\Theta(\log(n))$*, unless* P = NP*.*

Proof sketch. Our proof of log-APX-hardness has two parts. First, we show log-APX-hardness of a variant of the problem that is easier to analyze, which we call Immobile. In this variant, there exist two types of modules: (1) ***immobile modules*** that cannot perform any moves, and (2) ***mobile modules*** that can move.

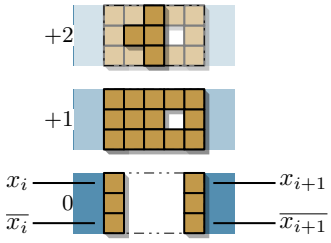
We then argue the assumptions made in the immobile variant remain valid if augment every immobile module with a ***spike gadget***. A spike gadget is a long, width-1 chain of modules extending in a fixed direction, that blocks the motion of all immobile modules in the Immobile variant. Any attempt to slide an immobile module disconnects the spike gadget from the remainder of the construction, thereby violating the backbone constraint.



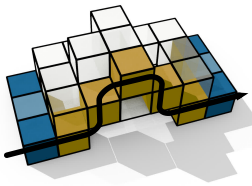
(a)



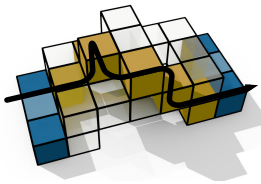
(b)



(c)



(d)



(e)

Figure 3 Our construction for $\varphi = (x_1 \vee x_3 \vee x_4) \wedge (x_1 \vee x_2 \vee x_3) \wedge (\overline{x_1} \vee \overline{x_2} \vee \overline{x_4}) \wedge (\overline{x_2} \vee \overline{x_3} \vee \overline{x_4})$ is shown in (a). In (b), a solution is illustrated (moving modules deleted). Variable gadgets are placed in ascending order left to right. In (c)–(e) we show the connector gadget and possible paths.

Our reduction is from the log-APX-hard problem Set Cover [7]. An instance $\Sigma_{n,k}$ consists of natural numbers $1, \ldots, n$ and a set $S = \{s_1, \ldots, s_k\}$ of subsets of $\{1, \ldots, n\}$ such that $\bigcup_{i=1}^{k} s_i = \{1, \ldots, n\}$. The task is then to find a set $S^\star$ of minimum cardinality such that $\bigcup_{s_i \in S^\star} s_i = \{1, \ldots, n\}$. A top-down view of the construction's blueprint is shown in Figure 4. Black lines describe thin paths of immobile modules, each of width and height 1, that connect all gadgets to one another.

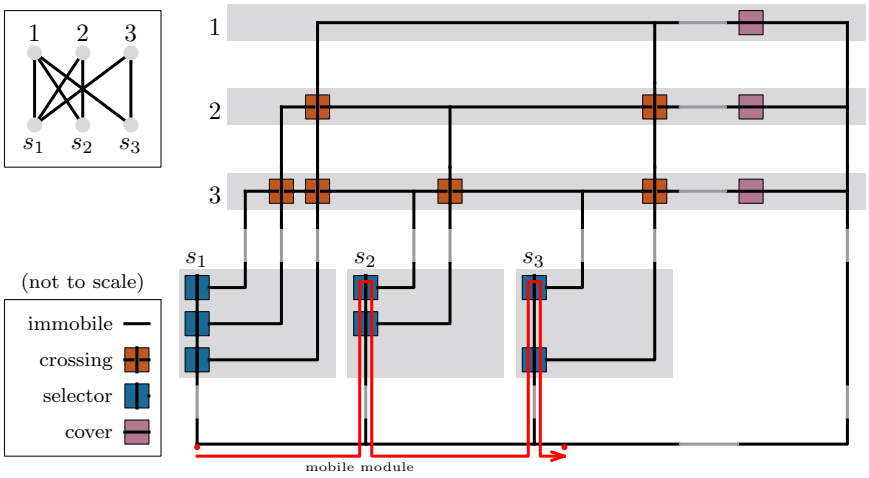


Figure 4 A diagram showing our reduction from the Set Cover instance with $n = 3$ over the sets $s_1 = \{1, 2, 3\}$, $s_2 = \{2, 3\}$, and $s_3 = \{1, 3\}$. The red path corresponds to the solution $\{s_2, s_3\}$.

The gadgets that directly relate to the set cover instance are the ***cover gadget*** and the ***selector gadget***. In particular, for every $j = 1, \ldots, n \in \mathbb{N}$, we place a cover gadget, depicted as a violet square in Figure 4, and in detail in Figure 5. In a cover gadget, the path of immobile modules is interrupted by a single mobile module m_j, highlighted in orange in Figure 5. Cover gadgets contain the majority of modules of the symmetric difference between the initial and the final configuration. In particular, the orange module is at level 0 the in C_I (the initial configuration), whereas in C_F (the final target configuration) it has been raised to level +1, directly above its original position. To enable motion of the orange modules inside the cover gadgets, we must ensure that the backbone remains connected. This is realized by the selector gadgets, depicted as a dark blue square in Figure 4, and in detail in Figure 6. The gadget is used to represent the containment of a number j in a set s_i. Within a selector gadget, one path of immobile modules is oriented orthogonally to a second path, and each path connects to the cover gadget associated with the number j, but from different directions. Moving a mobile module to connect the two paths establishes the backbone that in turn enables the orange module of the cover gadget to move.

More precisely: the dual graph of C_I forms a tree, no orange module in the symmetric difference may move without violating the backbone constraint. In C_I we place a single mobile module m at the bottom-left corner, and move it toward a selector s_i to filling the gap inside. This closes a cycle in the dual graph, and enables orange modules inside of the cover gadgets belonging to s_i to move to their respective final positions. Since the distance between the bottom row (which forms a comb with the selector gadgets, forcing us to route through it) and any selector is large, we aim to traverse these distances as rarely as possible. Equivalence between an optimal solution to the set cover instance and a schedule of minimum makespan follows from appropriate adjustments to inter-gadget distances. ◀

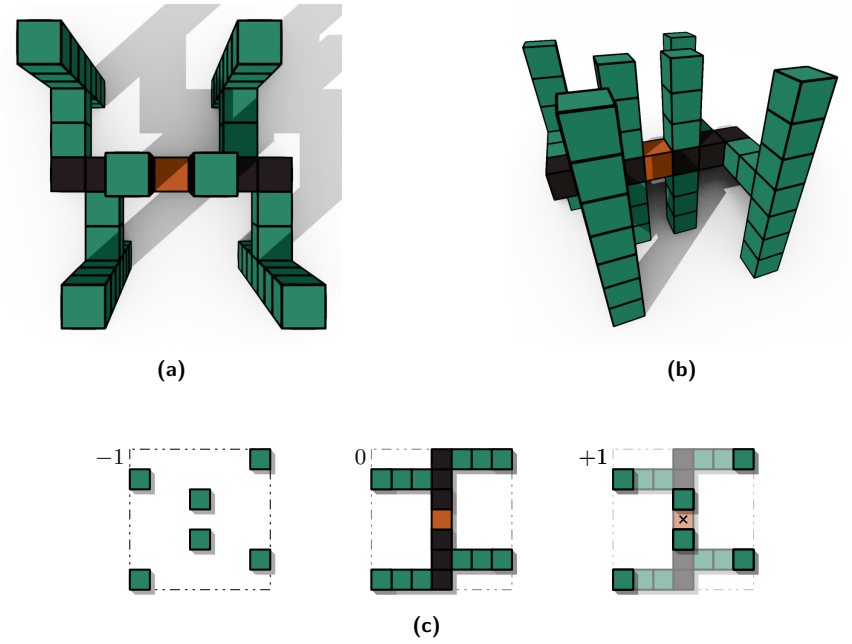


Figure 5 The cover gadget. The orange module is mobile with starting position at layer 0, and target position in layer +1. The black cubes are immobile. Green cubes correspond to spikes.

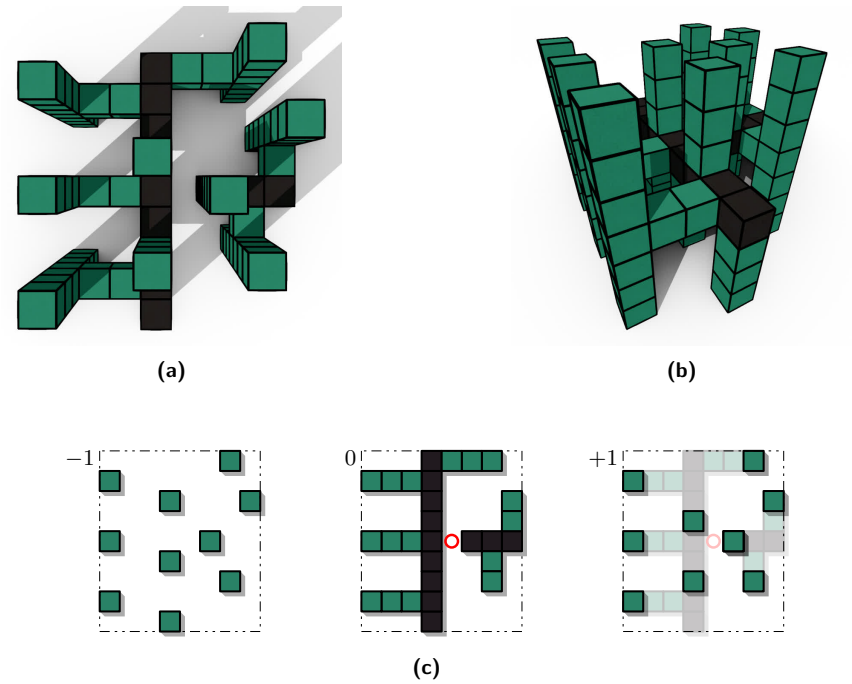


Figure 6 Selector gadgets enable modules within cover gadgets to perform their moves.

4 Teleport

A central component of our algorithmic approach generalizes LocateAndFree from [1], which locates a module in the outer face of the configuration and frees this module without changing the outer face. Our method operates under stricter constraints, introduced below.

A ***face*** of the configuration C is the set of module faces in the boundary of a component of the complement $\overline{C} = \mathbb{Z}^3 \setminus C$. The ***outer face*** of C is the face of the unbounded component of $\overline{C}$. Two adjacent module faces f_1 and f_2 in a face of C are ***slide-adjacent*** if either they bound the same cell $c \in \overline{C}$, or if a module in $c_1 \in \overline{C}$ containing f_1 can move to a cell $c_2 \in \overline{C}$ containing f_2 in one step from the configuration $C \cup \{c_1\}$. Each face of C can be seen as a 2-manifold obtained by joining slide-adjacent module faces at their common edge; this corresponds to a component of the "slide-adjacency graph" in [1]. An edge e of a configuration face is ***pinched*** if it is contained in more than two module faces in a face of C. In three dimensions, a pinched edge is contained in exactly four module faces and up to two configuration faces. We define an ***extended face*** of C as the manifold obtained by performing a topological surgery at each pinched edge, joining pairs of module faces f_1 and f_2 at the common pinched edge so that f_1 and f_2 are contained in the boundary of the same module in C. The ***outer extended face*** of C is the extended face that contains the outer face of C. We say that a module is *in a face* F of C if one of its faces is.

▶ **Lemma 4.1.** *Any connected configuration has a free module in the outer extended face.*

Given a subset S of a configuration C, a schedule ***teleports*** S if it transforms C into a new configuration C' with $(C \setminus S) \subset C'$. We show that if S, $S' = C' \setminus C$, and $C \cap C'$ are connected configurations, there is a schedule that teleports S to S'. Although the bound we prove is not tight, this result is of independent interest: It is stricter than LocateAndFree in that we teleport while maintaining the rest of the configuration, internal and external.

▶ **Lemma 4.2.** *Given a configuration C, a connected subset $S \subset C$ with $|S| \geq 2$ such that $C \setminus S$ is connected, an empty cell e adjacent to S, a free module $\mathsf{m} \in S$ so that e and m are in the same extended face of $S \setminus \{\mathsf{m}\}$, and so that both $(S \setminus \{\mathsf{m}\}) \cup \{e\}$ and $(C \setminus \{\mathsf{m}\}) \cup \{e\}$ are connected. There exists a schedule with makespan $2^{\mathcal{O}(|S|)}$ teleporting m to e.*

We require that $|S| \geq 2$ and $(S \setminus \{\mathsf{m}\}) \cup \{e\}$ is connected so we can bound the makespan as a function of $|S|$. If either of these conditions fail, m may be required to walk on modules not in S, giving a runtime dependency on $|C|$. Those conditions can be dropped if we are only concerned with the existence of schedules. Given two configurations $C \cup S$ and $C \cup S'$ we can apply successively Lemma 4.2 to grow a component of $S \cap S'$, obtaining this corollary:

▶ **Corollary 4.3.** *Given two configurations $C \cup S$ and $C \cup S'$ with connected C, S and S', there exist a schedule that teleports S into S'.*

5 A worst-case optimal algorithm

We leverage the results from Section 4 to efficiently reconfigure in the Parallel Sliding Cubes model. On a high level, our methods form a generalization of [4] to three dimensions. The strategy can be divided into three phases. The goal of (1) Gather is to obtain a ***snake*** which is informally a set of connected $5 \times 5 \times 5$ metamodules. Intuitively, the snake can move freely and quickly through the configuration without affecting the connectivity of the remaining configuration. To guide metamodules, we divide the grid into $5 \times 5 \times 5$ ***meta-cells***. In the (2) Compress phase, we use the snake to create a coarse projection of the

configuration onto one of the faces of the configuration's bounding box. We use this structure to sweep through and compress the rest of the configuration into a ***compact*** configuration of $5 \times 5 \times 5$ metamodules. A configuration C is compact if for every module $\mathsf{u} \in C$, the cuboid defined by the position of u and the origin is fully contained in the configuration:

$$\forall \mathsf{u} \in C: \quad \forall a \in [0, x(u)] : \forall b \in [0, y(u)] : \forall c \in [0, z(u)] : \quad (a, b, c) \in C.$$

These can be arbitrarily reconfigured due to our (3) Compact Reconfiguration phase.

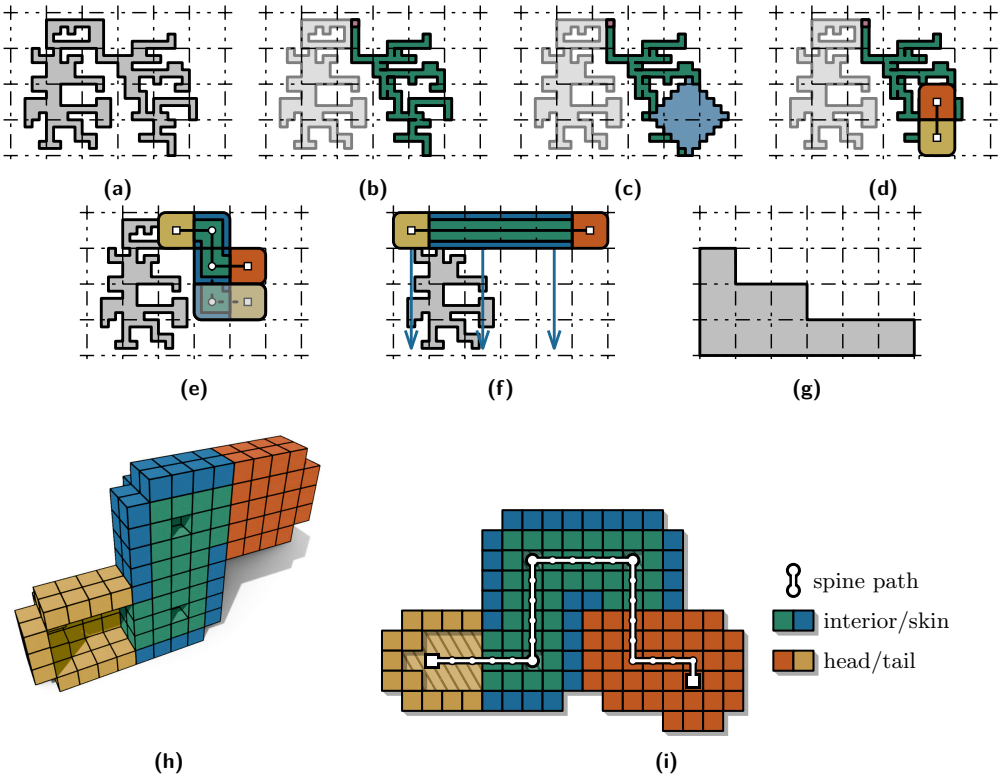


Figure 7 Overview of our algorithm (not to scale). In (a)–(d), we construct a snake, which we grow to $\Theta(A + h)$ modules in (e) and use it to compact the remaining configuration in (f)–(g). A high-level overview of a snake along its shape-defining spine path is shown in (h) and (i).

▶ **Theorem 5.1.** *For any instance of Parallel Sliding Cubes, a schedule of makespan $\mathcal{O}(A + h)$ can be computed in polynomial time, where $A = \max(|\lambda[C_1]|, |\lambda[C_2]|)$ and h is the maximal extent of either configuration along some axis $\lambda \in \{x, y, z\}$.*

Since the first two phases (illustrated in Figure 7) deal with a single input configuration, assume we are given a configuration C with projection area of C and height h along some axis. We start by computing a rooted spanning tree of C to guide the Gather phase. We then determine a subtree T_{s} of size $\Theta(A + h)$ and apply Lemma 4.2 to construct a constant-size ball, which induces a snake configuration. In the Compress phase, we move the snake along T_{s} in a depth-first fashion, growing it into a connected set of metamodules of sufficient size to cover the projection of C. Once this state is obtained, parallel translation of the snake along the chosen axis allows us to efficiently compact the remaining modules.

References

1 Zachary Abel, Hugo A. Akitaya, Scott Duke Kominers, Matias Korman, and Frederick Stock. A universal in-place reconfiguration algorithm for sliding cube-shaped robots in a quadratic number of moves. In *Symposium on Computational Geometry (SoCG)*, pages 1:1–1:14, 2024. doi:10.4230/LIPIcs.SoCG.2024.1.

2 Hugo A. Akitaya, Erik D. Demaine, Matias Korman, Irina Kostitsyna, Irene Parada, Willem Sonke, Bettina Speckmann, Ryuhei Uehara, and Jules Wulms. Compacting squares: Input-sensitive in-place reconfiguration of sliding squares. In *Scandinavian Symposium and Workshops on Algorithm Theory (SWAT)*, pages 4:1–4:19, 2022. doi:10.4230/LIPIcs.SWAT.2022.4.

3 Hugo A. Akitaya, Joseph Dorfer, Peter Kramer, Christian Rieck, Gabriel Shahrouzi, and Frederick Stock. Sliding cubes in parallel, 2026. arXiv:2603.08537.

4 Hugo A. Akitaya, Sándor P. Fekete, Peter Kramer, Saba Molaei, Christian Rieck, Frederick Stock, and Tobias Wallner. Sliding squares in parallel. In *European Symposium on Algorithms (ESA)*, pages 28:1–28:17, 2025. doi:10.4230/LIPIcs.ESA.2025.28.

5 Hugo A. Akitaya, Matias Korman, and Frederick Stock. Input-sensitive reconfiguration of sliding cubes. In *Canadian Conference on Computational Geometry (CCCG)*, pages 216–222, 2025. arXiv:2507.04170.

6 Mark de Berg and Amirali Khosravi. Optimal binary space partitions for segments in the plane. *International Journal on Computational Geometry and Applications*, 22(3):187–206, 2012. doi:10.1142/S0218195912500045.

7 Irit Dinur and David Steurer. Analytical approach to parallel repetition. In *Symposium on Theory of Computing (STOC)*, pages 624–633, 2014. doi:10.1145/2591796.2591884.

8 Adrian Dumitrescu and János Pach. Pushing squares around. *Graphs and Combinatorics*, 22(1):37–50, 2006. doi:10.1007/s00373-005-0640-1.

9 Adrian Dumitrescu, Ichiro Suzuki, and Masafumi Yamashita. Motion planning for metamorphic systems: feasibility, decidability, and distributed reconfiguration. *Transactions on Robotics*, 20(3):409–418, 2004. doi:10.1109/TRA.2004.824936.

10 Robert Fitch, Zack J. Butler, and Daniela L. Rus. Reconfiguration planning for heterogeneous self-reconfiguring robots. In *International Conference on Intelligent Robots and Systems (IROS)*, pages 2460–2467, 2003. doi:10.1109/IROS.2003.1249239.

11 Ferran Hurtado, Enrique Molina, Suneeta Ramaswami, and Vera Sacristán Adinolfi. Distributed reconfiguration of 2d lattice-based modular robotic systems. *Autonomous Robots*, 38(4):383–413, 2015. doi:10.1007/S10514-015-9421-8.

12 Irina Kostitsyna, Tim Ophelders, Irene Parada, Tom Peters, Willem Sonke, and Bettina Speckmann. Optimal in-place compaction of sliding cubes. In *Scandinavian Symposium and Workshops on Algorithm Theory (SWAT)*, pages 31:1–31:14, 2024. doi:10.4230/LIPICS.SWAT.2024.31.

13 Othon Michail, George Skretas, and Paul G. Spirakis. On the transformation capability of feasible mechanisms for programmable matter. *Journal of Computer and System Sciences*, 102:18–39, 2019. doi:10.1016/j.jcss.2018.12.001.

14 Joel Moreno and Vera Sacristán. Reconfiguring sliding squares in-place by flooding. In *European Workshop on Computational Geometry (EuroCG)*, pages 32:1–32:7, 2020. URL: https://computational-geometry.org/abstracts/eurocg/2020.pdf.

Undirected TSP as a Constrained GSTP Variant

Yılmaz Arslanoğlu

Independent Researcher
`yilmaz@hoketo.com`

Abstract

Classical formulations of the Traveling Salesman Problem (TSP) search for a 1-dimensional cycle of edges. We propose a novel formulation that constructs a 2-dimensional surface instead, motivated by the geometric intuition that a tour forms the boundary of a surface. This geometric shift maps symmetric TSP to a constrained Group Steiner Tree Problem variant (*cGSTP*), where maximizing the net weight of this tree is equivalent to minimizing the tour length. Based on this structure, we establish a Mixed-Integer Linear Program (MILP) that leverages the Euler characteristic to enforce topological consistency. The formulation is exact when applied to the complete input complex, despite being computationally intractable for all but the smallest instances. In practice, it provides an effective heuristic when restricted to a sparse candidate set such as Delaunay triangulation.

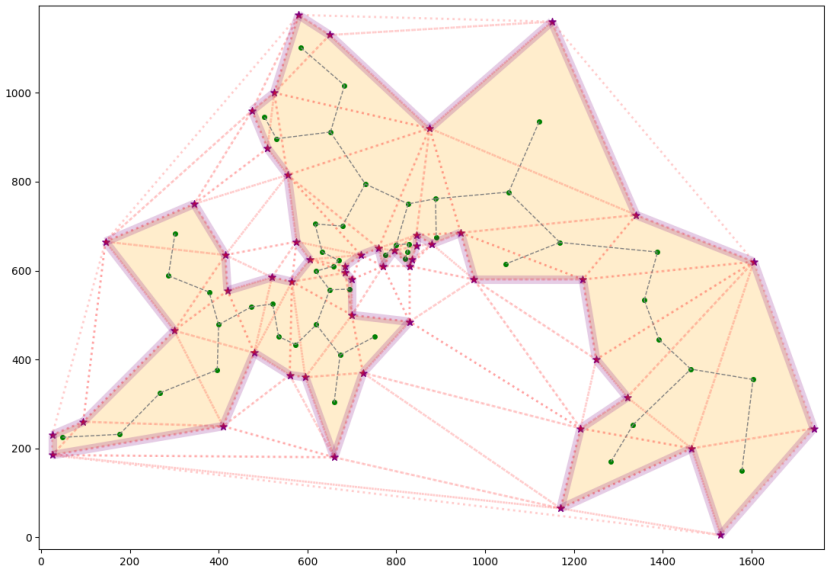


Figure 1 Visual intuition on `berlin52` ($N = 52$): select a connected set of triangles (surface); its boundary is the optimal tour.

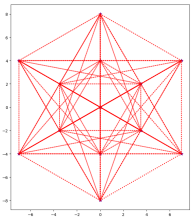
1 Introduction

TSP is conventionally modeled on a graph $G = (V, E)$: choose a minimum-weight Hamiltonian cycle by selecting edges $E' \subseteq E$ (Fig. 1). Edge-based models enforce local degree conditions but must also encode global connectivity, often via strong but exponential subtour elimination constraints [7], or via weaker alternatives such as MTZ [22] or flow-based constraints [15]. We propose a shift in perspective from *graph-theoretic* cycle selection to a *topological* construction. Instead of building the tour as a 1-dimensional sequence of edges, we construct a 2-dimensional surface whose boundary is the Hamiltonian cycle. Given a candidate triangle set $\mathcal{T} \subseteq \binom{V}{3}$,

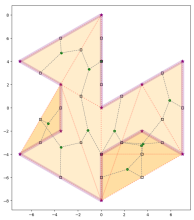
42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

we select a connected subset whose induced boundary length is minimized. When $\mathcal{T}$ is the universal set $\mathcal{K}_{All} := \binom{V}{3}$ (Fig. 2a), the formulation is exact but computationally feasible only for very small N. Restricting $\mathcal{T}$ to a sparse triangulation (e.g., Delaunay, Fig. 2c) yields a highly efficient candidate-set heuristic, though it may render the model infeasible if $\mathcal{T}$ contains no Hamiltonian cycle [10].

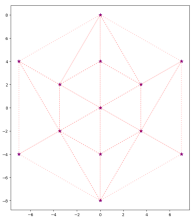
Related Work. Classical exact methods rely on branch-and-cut with subtour elimination [1, 7]. Tree-structured connectivity constraints are well-studied in flow-based routing formulations [15] and in Steiner variants [20]. Geometric approximation schemes for Euclidean TSP exploit recursive planar subdivisions [2, 23]. Our formulation aligns with boundary-minimization viewpoints from computational topology [12] and geometry [13, 14], planar duality principles [27], and optimization of cycles on surfaces [5], but differs in that we optimize the *surface itself* via a simplicial incidence structure. This yields a constrained GSTP [25], similar to geometric covering variants like TSP with Neighborhoods [11], whose objective resembles prize-collecting Steiner tree terms [16], except that "prizes" arise from simplicial incidences (i.e., edges of the Steiner tree).



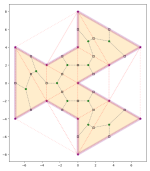
(a) Dense Input ($\mathcal{K}_{All}$)



(b) Dense Solution (non-planar; overlap is a projection artifact)



(c) Delaunay Input ($\mathcal{T}_{Delaunay}$)



(d) Delaunay Solution (Planar)

Figure 2 Nested hexagons ($N = 13$): with all triangles the optimum is an abstract non-planar surface; Delaunay input forces a planar one.

Contribution. Our contribution is fourfold:

(i) We establish (in Sec. 2) an equivalence between min-weight Hamiltonian cycles in the *complete* complex $\mathcal{K}_{All}$ (Fig. 2a) and max-weight dual trees in the bipartite incidence graph (Fig. 3);

(ii) We give a MILP that replaces subtour elimination by a global tree constraint plus a local Euler characteristic filter (Sec. 3);

(iii) We prove (in companion note [3]) that these constraints enforce a disk whose boundary is a single simple Hamiltonian cycle; and

(iv) We demonstrate that the formulation is fully *input-decoupled*: it treats the candidate

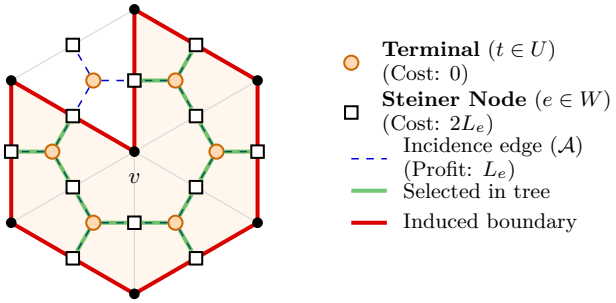


Figure 3 Bipartite incidence graph: triangles x_t (circles) and edges y_e (squares); dashed lines are adjacencies. A feasible solution selects a connected set of triangles (filled orange), inducing the highlighted boundary (red) around the center; translucent green edges indicate incidence edges chosen in the resulting tree.

triangle set $\mathcal{T}$ as a black-box constraint on the search space, working both on $\mathcal{K}_{All} = \binom{V}{3}$ as well as on sparse inputs such as Delaunay or Greedy triangulations [8, 10, 21].

2 Reduction to cGSTP

We first establish the structural transformation of the problem, demonstrating that the TSP on G corresponds to a cGSTP on a dual structure, augmented with topological counting (cardinality and degree) constraints. Given a TSP instance on $G = (V, E)$, we construct the *Bipartite Incidence Graph* $B = (U \cup W, \mathcal{A})$ (see Fig. 3), which represents the connectivity between triangles and edges:

- **Terminals (U):** A candidate triangle set $U := \mathcal{T} \subseteq \binom{V}{3}$.
- **Steiner Nodes (W):** The set of primal edges $W := E$.
- **Adjacency ($\mathcal{A}$):** An edge exists between $t \in U$ and $e \in W$ if triangle t contains edge e.

To establish equivalence, we assign costs and profits as detailed in Table 1, where L_e denotes the metric length of edge e. Each city v (e.g., the central node labeled v in Fig. 3) induces the group

$$U(v) := \{t \in U \mid v \in t\},$$

i.e., the set of candidate triangles incident to v. Group satisfaction is enforced through the group-coverage and topological admissibility constraints.

Table 1 Mapping to a cGSTP: costs, profits, and enforced topological degrees.

Primal Concept	cGSTP Component	Cost & Profit / Degree Constraint
Triangle ($t \in \mathcal{T}$)	Terminal Node ($t \in U$)	Cost 0; Deg $\in \{0, 3\}$ (t selected $\rightarrow$ all edges selected)
Edge ($e \in E$)	Steiner Node ($e \in W$)	Cost $2L_e$; Deg ≤ 2 (manifold condition)
Incidence ($e \subset t$)	Edge in $\mathcal{A}$	Profit L_e (connection reward)
City ($v \in V$)	Group $U(v)$	Target: Visit ≥ 1 terminal in group

Any feasible surface selection induces an active subgraph $B' \subseteq B$. Define its net weight as:

$$NW(B') := \sum_{(t,e)\in\mathcal{A}(B')} profit(t, e) \; - \sum_{e\in W(B')} cost(e). \tag{1}$$

Internal edges are incident to two selected triangles and contribute $2L_e - 2L_e = 0$ to $NW(B')$; boundary edges are incident to one triangle and contribute $L_e - 2L_e = -L_e$. Hence maximizing $NW(B')$ is equivalent to minimizing the total boundary length (the tour). Section 3 implements this equivalence directly.

3 Mathematical Formulation

We formulate the problem as finding a subgraph in the bipartite incidence graph $B = (U \cup W, \mathcal{A})$ that corresponds to a valid surface. Let $x_t, y_e, z_{te} \in \{0, 1\}$ be decision variables representing the selection of triangles, edges, and their incidences, respectively. The objective is to minimize the length of the boundary $\partial\mathcal{K}$ of the resulting surface complex $\mathcal{K}$:

$$\text{Min } Z = \sum_{e \in W} 2L_e y_e - \sum_{(t,e) \in \mathcal{A}} L_e z_{te} \tag{2}$$

Eq. (2) is exactly $Z = -NW(B')$ for the induced active subgraph B' (Eq. 1), so minimizing Z corresponds to maximizing the net weight.

We enforce topological validity via the following five structural constraints:

1. *Incidence linking (Terminal Node Degrees):* Every active triangle must use all three of its edges:

$$z_{te} \leq y_e$$

$$z_{te} \leq x_t$$

$$\sum_{e \subset t} z_{te} = 3x_t$$

2. *Manifold Condition (Steiner Node Degrees):* Every primal edge must be incident to at most two triangles (Fig. 4A depicts a violation). In the dual, this restricts the degree of Steiner nodes y_e within the incidence graph.

$$y_e \leq \sum_{t \supset e} z_{te} \leq 2y_e$$

3. *Node Cardinality Constraints:* To guarantee $\mathcal{K}$ is a triangulation of a topological disk with N vertices, we enforce $\sum x_t = N - 2$ (triangles) and $\sum y_e = 2N - 3$ (edges).
4. *Global Connectivity (Tree):* B' must form a single connected tree (see [4] for the detailed implementation). This ensures $\mathcal{K}$ is simply connected (topologically a disk).
5. *Local Connectivity (Steiner Groups):* To prevent vertex singularities (e.g., "bowties", Fig. 4B), the neighborhood of each city v must form a simple path (cf. the center vertex in Fig. 3). We enforce this using the local Euler characteristic of the incidence subgraph $H_v \subseteq B'$ [12]:

$$\chi(H_v) = |\mathrm{V}(H_v)| - |E(H_v)| = 1 \quad \forall v \in V. \tag{3}$$

Here H_v denotes the subgraph of B' induced by the triangles and edges incident to v. Under the global tree constraint, enforcing $\chi(H_v) = 1$ leads to a simple path, ruling out disconnected vertex links and local "bowtie" singularities (Fig. 4B). The presence of at least one terminal per group is thus also implicitly enforced [3].

Figure 4 Forbidden anomalies: (A) An edge incident to > 2 triangles; (B) A “bowtie” vertex violating the Euler characteristic filter.

4 Computational Validation and Results

We validated the correctness of the formulation on simple topological cases before measuring performance.

Visual Verification (Topology). To provide a visual verification, we apply the model to an instance consisting of two nested hexagons with a central node ($N = 13$). Fig. 2 compares the solver's behavior under two different candidate triangle sets: the universal set $\mathcal{K}_{All}$ and the Delaunay triangulation $\mathcal{T}_{Delaunay}$. In both cases, the formulation successfully extracts the optimal surface within the chosen input complex.

Validation on Non-Metric Spaces. To test the topological generality of our formulation, we generated random symmetric non-metric instances of small size where edge weights do not satisfy the triangle inequality. The input was $\mathcal{K}_{All}$. The solver successfully extracted valid dual trees and primal tours; Fig. 5 shows a representative instance with $N = 10$. We verified optimality against a standard TSP formulation (MTZ).

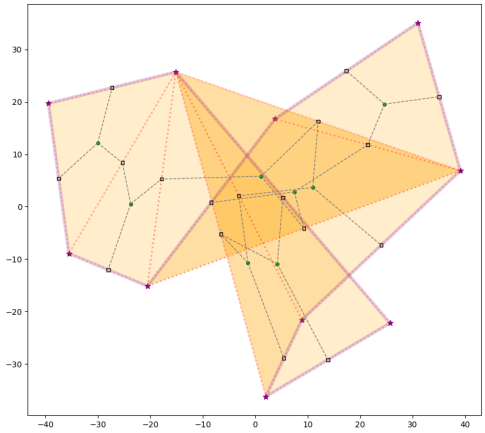


Figure 5 Random symmetric non-metric instance ($N = 10$, visualized by MDS). The topological formulation remains applicable even when the triangle inequality is violated.

Comparison. We restrict comparisons to compact static models, excluding dynamic cut-generation methods (e.g., Concorde [1]). We benchmarked three formulations:

1. **Conventional:** Standard Lifted-MTZ formulation [9] restricted to edges in the triangulation.
2. **SB-MTZ:** Surface-based formulation using MTZ [22]-based tree constraints.
3. **SB-Flow:** Surface-based formulation using Single-Commodity Flow [15]-based tree constraints.

We utilized a planar simplification of the model, which eliminates explicit Steiner node variables y_e, as the incidence graph collapses to the standard dual graph for planar triangulations [4]. For the sake of reproducibility, a MiniZinc [24] implementation is publicly available at

`https://github.com/hoketo/surface_based_tsp_with_minizinc`

We utilized HiGHS v1.12 [19] on an Apple M4 Max. Table 2 summarizes the performance across all instances, including several from TSPLIB [26]. All experiments use the Delaunay triangulation unless stated otherwise. The results indicate that the SB-Flow generally outperforms SB-MTZ as N grows. For instances ($N \leq 280$), it typically solves at the root. For larger instances, SB-Flow yields a substantially tighter relaxation, reflected in dramatically smaller Branch-and-Bound trees (e.g., 142 nodes vs. 23,205 for `pcb442`).

Table 2 Combined Computational Results. Post-presolve MILP size and performance; values are 10-run means (unless noted), with a timeout limit of 1-hour.

		MILP Size		HiGHS	
Instance	**Model**	**Vars**	**Cons**	**Time (s) [Solved]**	**Nodes**
Berlin52	Lifted-MTZ	341	523	0.08s [10/10]	1
($N = 52$)	SB-MTZ	553	1,107	**0.02s** [10/10]	1
	SB-Flow	684	1,054	0.03s [10/10]	1
St70	Lifted-MTZ	463	723	14.23s [10/10]	2,936
($N = 70$)	SB-MTZ	759	1,521	4.35s [10/10]	257
	SB-Flow	926	1,440	**0.20s** [10/10]	**1**
Ch130	Lifted-MTZ	879	1,362	108.6s [10/10]	16,058
($N = 130$)	SB-MTZ	1,458	2,922	**0.29s** [10/10]	**1**
	SB-Flow	1,793	2,775	1.62s [10/10]	**1**
Tnm199	SB-Flow (Delaunay)	2,764	4,283	**0.51s** [10/10]	**1**
($N = 199$)	SB-Flow (Greedy)	2,090	3,202	4.76s [10/10]	1.6
A280	Lifted-MTZ	1,907	2,992	DNF [0/5]	>53k
($N = 280$)	SB-MTZ	3,188	6,379	29.0s [10/10]	490
	SB-Flow	3,923	6,056	**5.68s** [10/10]	**1**
Pcb442	SB-MTZ	5,195	10,393	1656s* [7/10]	23,205
($N = 442$)	SB-Flow	6,412	9,879	**81.1s** [10/10]	**142**
d657	SB-Flow	9,634	14,837	TLE (0.60%)	>30k
($N = 657$)					

* For `pcb442` SB-MTZ, 3/10 runs exceeded the 1-hour time limit; time and nodes are averaged over the 7 solved runs.

Degenerate Inputs. We tested on `Tnm199` ($N = 199$) from the "Hard-to-Solve" dataset [18]. This instance is constructed to be practically intractable for standard exact solvers; even Concorde requires ≈ 4.7 days to find the global optimum [18]. Our formulation effectively exploits the Greedy triangulation to achieve a high-quality tour with 1.85% gap (vs. 6.17% for Delaunay), surpassing the 5.69% gap of the Christofides heuristic [6]. This demonstrates that the topological formulation scales in quality based on the preferred input complex.

To test the limitations of sparsification under degenerate geometry, we evaluated the instance `p100.100000` ($N = 100$), which arranges points on three parallel lines [17]. The results reveal a critical vulnerability in geometric sparsifiers. While Christofides remains

robust (2.52% gap), Delaunay and Greedy triangulations degrade badly (14%–119% gap), exposing a limitation of sparse candidate sets on near-degenerate geometry.

5 Conclusion

We have presented a surface-based formulation for TSP that shifts the decision space from edges to triangles. This changes the optimization object from a primal cycle to a dual surface: instead of directly enforcing Hamiltonicity on G, we enforce disk-admissibility of a 2-complex whose boundary is the tour. Our experiments indicate that this topological shift can yield improved B&B behavior relative to compact, static edge-based baselines on sparse geometric inputs, particularly when enforcing the global tree constraint via flow. While exact convergence becomes computationally prohibitive for larger instances in our current setup, the results suggest that the surface-based model provides a useful foundation for integrating additional geometric or topological constraints that could be cumbersome in primal edge formulations.

Acknowledgements

We thank Prof. Stefan Hougardy for his feedback and for providing the test instances. We also thank Anh Quyen Vuong for suggestions regarding the formulation and proof structure. Finally, we acknowledge the developers of HiGHS for providing a high-performance open-source solver, and the MiniZinc team for providing a flexible modeling framework that enables reproducible experiments.

References

1 David L. Applegate, Robert E. Bixby, Vašek Chvátal, and William J. Cook. *The Traveling Salesman Problem: A Computational Study*. Princeton University Press, 2006.

2 Sanjeev Arora. Polynomial time approximation schemes for euclidean traveling salesman and other geometric problems. *Journal of the ACM*, 45(5):753–782, 1998.

3 Yılmaz Arslanoğlu. A structural equivalence of symmetric TSP to a constrained group Steiner tree problem. arXiv preprint arXiv:2602.05773, 2026. URL: https://arxiv.org/abs/2602.05773.

4 Yılmaz Arslanoğlu. A surface-based formulation of the traveling salesman problem. arXiv preprint arXiv:2603.00075, 2026. URL: https://arxiv.org/abs/2603.00075.

5 Erin W. Chambers, Jeff Erickson, and Amir Nayyeri. Minimum cuts and shortest homologous cycles. In *Proceedings of the 25th Annual Symposium on Computational Geometry (SoCG 2009)*, pages 377–385. ACM, 2009.

6 Nicos Christofides. Worst-case analysis of a new heuristic for the travelling salesman problem. Technical report, Graduate School of Industrial Administration, Carnegie Mellon University, 1976.

7 George B. Dantzig, Ray E. Fulkerson, and Selmer M. Johnson. Solution of a large-scale traveling-salesman problem. *Operations Research*, 2(4):393–410, 1954.

8 Gautam Das and Deborah A. Joseph. Which triangulations approximate the complete graph? In *Optimal Algorithms*, volume 401 of *Lecture Notes in Computer Science*, pages 168–192. Springer, 1989.

9 Martin Desrochers and Gilbert Laporte. Improvements and extensions to the Miller-Tucker-Zemlin subtour elimination constraints. *Operations Research Letters*, 10(1):27–36, 1991.

10 Michael B. Dillencourt. Toughness and delaunay triangulations. *Discrete & Computational Geometry*, 5(6):575–601, 1990.

11 Adrian Dumitrescu and Joseph S. B. Mitchell. Approximation algorithms for TSP with neighborhoods in the plane. *Journal of Algorithms*, 48(1):135–159, 2003.

12 Herbert Edelsbrunner and John Harer. *Computational Topology: An Introduction*. American Mathematical Society, 2010.

13 Sándor P. Fekete, Andreas Haas, Michael Hemmer, et al. Computing nonsimple polygons of minimum perimeter. In *Experimental Algorithms (SEA 2016)*, volume 9685 of *Lecture Notes in Computer Science*, pages 134–149. Springer, 2016.

14 Sándor P. Fekete, Andreas Haas, Phillip Keldenich, Michael Perk, and Arne Schmidt. Computing area-optimal simple polygonizations. *ACM Journal of Experimental Algorithmics*, 27:2.6:1–2.6:23, 2022.

15 Bezalel Gavish and Stephen C. Graves. The traveling salesman problem and related problems. Technical Report GR-078-78, Operations Research Center, Massachusetts Institute of Technology, 1978.

16 Michel X. Goemans and David P. Williamson. A general approximation technique for constrained forest problems. *SIAM Journal on Computing*, 24(2):296–317, 1995.

17 Stefan Hougardy. On the integrality ratio of the subtour LP for euclidean TSP. *Operations Research Letters*, 42(8):495–499, 2014.

18 Stefan Hougardy and Xianghui Zhong. Hard to solve instances of the euclidean traveling salesman problem. Technical report, Research Institute for Discrete Mathematics, University of Bonn, 2018.

19 Qi Huangfu and Julian A. J. Hall. Parallelizing the dual revised simplex method. *Mathematical Programming Computation*, 10(1):119–142, 2018.

20 Adam N. Letchford, S. D. Nasiri, and Dirk O. Theis. Compact formulations of the steiner traveling salesman problem and related problems. *European Journal of Operational Research*, 228(1):83–92, 2013.

21 Adam N. Letchford and N. A. Pearson. Good triangulations yield good tours. *Computers & Operations Research*, 35(2):638–647, 2008.

22 C. E. Miller, A. W. Tucker, and R. A. Zemlin. Integer programming formulation of traveling salesman problems. *Journal of the ACM*, 7(4):326–329, 1960.

23 Joseph S. B. Mitchell. Guillotine subdivisions approximate polygonal subdivisions. *SIAM Journal on Computing*, 28(4):1298–1309, 1999.

24 Nicholas Nethercote, Peter J. Stuckey, Ralph Becket, Sebastian Brand, Gregory J. Duck, and Guido Tack. MiniZinc: Towards a standard CP modelling language. In *Principles and Practice of Constraint Programming (CP 2007)*, volume 4741 of *Lecture Notes in Computer Science*, pages 529–543. Springer, 2007.

25 Gabriele Reich and Peter Widmayer. Beyond steiner's problem: A VLSI oriented generalization. In *Graph-Theoretic Concepts in Computer Science (WG 1989)*, volume 411 of *Lecture Notes in Computer Science*, pages 196–210. Springer, 1990.

26 Gerhard Reinelt. TSPLIB — a traveling salesman problem library. *ORSA Journal on Computing*, 3(4):376–384, 1991.

27 Hassler Whitney. A theorem on graphs. *Annals of Mathematics*, 32(2):378–390, 1931.

Small Empty Cycles in Simple Drawings of K_n*

Anna Hofer[1], Joachim Orthaber[2], Birgit Vogtenhuber[1], and Alexandra Weinberger[3]

1 **Graz University of Technology**
anna.hofer@tugraz.at, birgit.vogtenhuber@tugraz.at
2 **Technische Universität Berlin**
orthaber@math.tu-berlin.de
3 **FernUniversität in Hagen**
alexandra.weinberger@fernuni-hagen.de

Abstract

Given a simple drawing of a graph, a sub-drawing of a plane cycle of length k induces two connected regions of the plane. If one of the two regions does not contain any of the remaining vertices of the drawing, we call the cycle an *empty plane k-cycle*. In this work, we examine simple drawings of the complete graph K_n with respect to the existence of empty plane k-cycles for small k. We show that every vertex in a simple drawing of K_n is incident to an empty plane k-cycle with $k = 5$ or $k = 6$.

1 Introduction

A *simple* drawing $\mathcal{D}$ of a graph is a drawing in the plane or on the sphere where vertices are distinct points, edges are represented by Jordan arcs connecting their end-points, and each pair of edges intersects at most once, either in a proper crossing or in a common end-point. A cycle in $\mathcal{D}$ is a *plane k-cycle* if it consists of $k \geq 3$ edges and no two of its edges cross in $\mathcal{D}$. For $k = 3$ we also call such a cycle a *triangle*. A plane k-cycle C partitions the plane or the sphere into two connected regions called the *sides* of C. If a side of C does not contain any of the remaining vertices of the drawing, the side is *empty*. A plane k-cycle is empty if at least one of its sides is empty. Note that in simple drawings of graphs on n vertices, every plane k-cycle with $3 \leq k < n$ has at most one empty side. If $k = n$, then C is a Hamiltonian cycle and both of its sides are empty. Rafla conjectured in 1988 [31] that every simple drawing of K_n contains such a plane Hamiltonian cycle. This conjecture has been verified for all simple drawings of K_n with $n \leq 10$ [11] as well as for several special families of simple drawings [2, 4, 5, 8]. In its full generality, it is still a (by now famous) conjecture. In this work, we focus on the following stronger and more recent conjecture:

▶ **Conjecture 1.1.** *[8, Conjecture 29] Every simple drawing of K_n with $n \geq 3$ vertices contains an empty plane k-cycle for each $3 \leq k \leq n$.*

The restriction of Conjecture 1.1 to straight-line drawings is an easy classroom exercise. For x-monotone drawings, it follows from the existence of a Hamiltonian cycle [5, 26]. More excitingly, Conjecture 1.1 has been proven for g-convex[1] drawings [8, Corollary 20]. For simple drawings in general, it has been shown for $k = 3$ [20] and $k = 4$ [10, Theorem 10]. Extending this line of research, we prove the following:

* J.O. was partially supported by DFG Grant FE 340/13-1.

[1] In the literature, e.g. [6, 8, 10], these drawings are often just called "convex" drawings. To avoid confusion with the convex straight-line drawings, we call them *g-convex* (short for generalized convex).

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

▶ **Theorem 1.2.** *Let $\mathcal{D}$ be a simple drawing of K_n for $n \geq 5$. Then there exists an empty plane k-cycle of length $k = 5$ or $k = 6$ at every vertex of $\mathcal{D}$.*

Further related work. In recent decades, plane sub-drawings of simple drawings of the complete graph K_n have received considerable attention. It is known that every simple drawing of K_n contains a plane sub-drawing with $2n - 3$ edges [36] and has at most $2n - 2$ uncrossed edges [32]. García, Pilz, and Tejel [17] showed that every maximal plane sub-drawing of a simple drawing of K_n has at least $\lceil \frac{3n}{2} \rceil$ edges and, furthermore, is 2-connected. There are numerous results on plane sub-drawings with additional structure, such as matchings [14, 30, 38, 15, 36, 2], paths and cycles [27, 2, 40], or trees [27, 40]. Some of the currently best bounds are: Every simple drawing of K_n contains a plane matching of size $\Omega(\sqrt{n})$, a plane cycle of length $\Omega(\frac{\log n}{\log \log n})$ [2], and a plane sub-drawing of any fixed tree on $(\log n)^{\frac{1}{4}-o(1)}$ vertices [40]. Regarding plane sub-structures of small cardinality, every simple drawing of K_n contains $\Omega(n)$ plane 4-cycles with pairwise disjoint interiors [41] as well as at least n empty triangles [3]. Moreover, Harborth and Mengersen [21] constructed *twisted drawings* of K_n [27] and Harborth [20] showed that they have only $2n - 4$ empty triangles.

For straight-line drawings of K_n (or, equivalently, n-point sets in general position), Erdős-Szekeres type questions on the existence and number of k-gons and k-holes in every sufficiently large point set P have been studied extensively. A k-gon in P is a simple convex polygon spanned by k points of P. A k-gon without any point of P in its interior is a k-hole. The famous Erdős-Szekeres Theorem [13] states that every set of $n \geq n_0$ points contains a k-gon, where the currently best upper bound on n_0 is $2^{k+\mathcal{O}\left(\sqrt{k \log(k)}\right)}$ [22]. For k-holes, the existence question was answered positively for $3 \leq k \leq 6$ [12, 19, 24, 18], while for $k \geq 7$ there exist arbitrarily large sets without k-holes [23]. Note that simple drawings of K_n might not contain k-gons (that is, sub-drawings isomorphic to a convex K_k) for $k \geq 5$ [21, 2]. However, it has been shown that every simple drawing of K_n contains a sub-drawing of K_k for $k = (\log n)^{\frac{1}{4}-o(1)}$ that is either convex or twisted [27, 40]. Concerning k-holes, it is known that simple drawings might not even contain 4-holes, but sufficiently large g-convex drawings of K_n always contain k-holes for every $3 \leq k \leq 6$ [10].

Outline. In Section 2, we introduce notations and show some lemmata on c-monotone drawings and on plane substructures, which we then use in Section 3 to prove Theorem 1.2. We conclude the paper in Section 4.

2 Definitions and auxiliary results

An edge e in a simple drawing $\mathcal{D}$ of a graph G is called *c-monotone* with respect to a point $\mathcal{O}$ if every ray emanating from $\mathcal{O}$ crosses e at most once. A simple drawing of G is c-monotone if there exists a point $\mathcal{O}$ so that every edge of the drawing is c-monotone with respect to $\mathcal{O}$. Every c-monotone drawing is strongly isomorphic[2] to a c-monotone drawing where all vertices lie on a common circle with center $\mathcal{O}$. We call two vertices v and w *neighboring* if they are adjacent in the circular order around $\mathcal{O}$. For an edge $e = \{v, w\}$ in a c-monotone drawing, we define the wedge Λ_e of edge e as the closed wedge with apex $\mathcal{O}$, bounded by the rays from $\mathcal{O}$ through the end-vertices v and w, that contains the edge e. We call an edge e

[2] Two drawings of the same graph are called *strongly isomorphic* if there is a homeomorphism of the sphere transforming one into the other. Note that plane sub-structures in simple drawings are invariant under strong isomorphism.

between neighboring vertices *long* if Λ_e contains all vertices of the drawing. Otherwise, by c-monotonicity, Λ_e contains only the end-vertices of e and we call e a *short* edge.

Our first result is that every c-monotone drawing of K_n with $n \geq 3$ contains a short edge:

▶ **Lemma 2.1.** *Let $\mathcal{D}$ be a c-monotone drawing of K_n with respect to a point $\mathcal{O}$ for $n \geq 3$. Then there exist two neighboring vertices v and w so that the edge $\{v, w\}$ is short.*

Lemma 2.1 can be shown by contradiction, using induction and the fact that for every vertex in a respective c-monotone drawing of K_n, its two neighbors are connected via an edge that is a long edge between neighbors in an induced sub-drawing on $n-1$ vertices.

For a vertex v in a simple drawing $\mathcal{D}$ of K_n, we denote the spanning sub-drawing consisting of all edges incident to v by $\text{star}(v)$. Note that, by simplicity of $\mathcal{D}$, $\text{star}(v)$ is plane.

▶ **Lemma 2.2.** *Let v be a vertex in a simple drawing $\mathcal{D}$ of K_n for $n \geq 5$. If there exists a vertex $w \neq v$ so that the double-star $\text{star}(v) \cup \text{star}(w)$ forms a plane sub-drawing of $\mathcal{D}$, then v is incident to two empty plane 5-cycles.*

Proof. Let $v \neq w$ be vertices as in the condition of Lemma 2.2; see Figure 1 for an illustration. Then $\text{star}(v) \cup \text{star}(w)$ without edge $\{v, w\}$ is a plane drawing of $K_{2,n-2}$ (bold edges in Figure 1). By [2, Theorem 1.6] the sub-drawing $\mathcal{D}'$ of $\mathcal{D}$ induced by $V \setminus \{v, w\}$ is strongly isomorphic to a c-monotone drawing of K_{n-2} where v assumes the role of $\mathcal{O}$. The circular order of the vertices in $\mathcal{D}'$ corresponds to the rotation of v (excluding vertex w). By Lemma 2.1, there exists at least one short edge $e = \{x, y\}$ in $\mathcal{D}'$ between two neighboring vertices x and y. In particular, this means that adding e to $\text{star}(v) \cup \text{star}(w)$ excluding the edge $\{v, w\}$ yields a plane sub-drawing. Let a and b be the vertices so that $(...a, x, y, b, ...)$ is the rotation of $V \setminus \{w\}$ around v in some fixed direction. Then (v, y, x, w, a) and (v, x, y, w, b) are two distinct empty plane 5-cycles incident to v (even for $n = 5$ where $a = b$). ◀

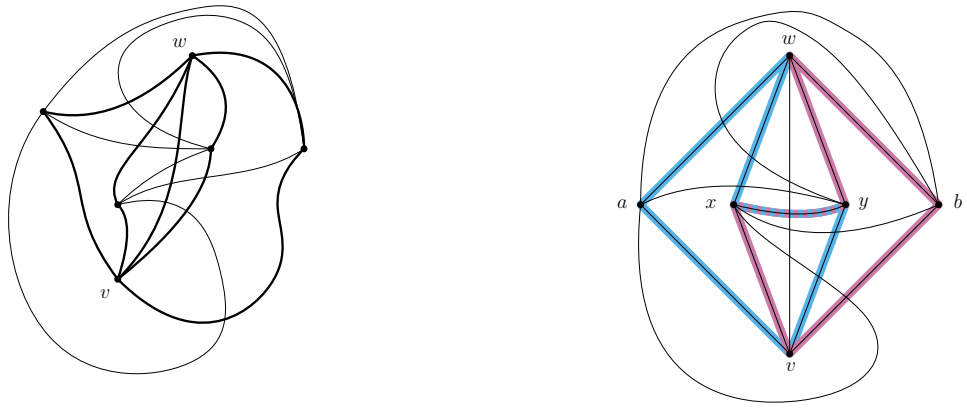


Figure 1 Two strongly isomorphic drawings of K_6 with plane double-star $\text{star}(v) \cup \text{star}(w)$.

We will use a result from García, Pilz, and Tejel [17] on the structure of plane sub-drawings.

▶ **Lemma 2.3** ([17, Corollary 3.4])**.** *Given a simple drawing $\mathcal{D}$ of K_n and some vertex v, there exists a spanning sub-tree T_v of K_{n-1} on $V \setminus \{v\}$, such that $\text{star}(v) \cup T_v$ forms a plane sub-drawing of $\mathcal{D}$.*

Furthermore, we will use so-called ice-cream cone drawings, which have been introduced in [25]. Every drawing $\mathcal{D}$ can, for any fixed vertex v, be transformed (via a homeomorphism

of the sphere) to a strongly isomorphic drawing in which $\text{star}(v)$ consists of straight-line edges going upward, v lies on the boundary of the unbounded cell, and all vertices in $V \setminus \{v\}$ lie on a horizontal line above v. Such a drawing is called an *ice-cream cone drawing at* v.

Let $\mathcal{D}'$ be the sub-drawing of such an ice-cream cone drawing at v, consisting of $\text{star}(v) \cup T_v$, where T_v is a spanning tree of $\mathcal{D}$ on $V \setminus \{v\}$ as given by Lemma 2.3. Note that, since $\mathcal{D}'$ is a plane drawing with n vertices and $2n-3$ edges and since T_v is a tree, all $n-1$ faces of $\mathcal{D}'$ are incident to v. Let the size of a face f be the number of its bounding edges. Every face of size k of $\mathcal{D}'$ constitutes an empty plane k-cycle in $\mathcal{D}$. We call two faces of $\mathcal{D}'$ *neighboring* if they share an edge that is incident to v. Faces of size $k = 3$ are called *triangular*. We call a bounded face f of size $k \geq 4$ an *umbrella* of size k. A bounded region P of $\mathcal{D}'$ is called a *pocket* of f if it does not contain f and if its boundary consists of a boundary edge e of f not incident to v and of the two edges from v to e. We denote e as the *top edge* of P, and the two boundary edges of P incident to v as the *pocket edges of* P. Further, for an umbrella f, we call all edges incident to v that are pocket edges for some pocket of f the *pocket edges of* f. See Figure 2 for an illustration of umbrellas and pockets. Note that an umbrella of size k has exactly $k-3$ pockets. An umbrella f is called *one-sided* if all its pockets either lie to the left or to the right of f (w.r.t. an ice-cream cone drawing). Otherwise, we call f *two-sided*. For a one-sided (resp. two-sided) umbrella f, exactly one (resp. two) of its boundary edges are also pocket edges of f.

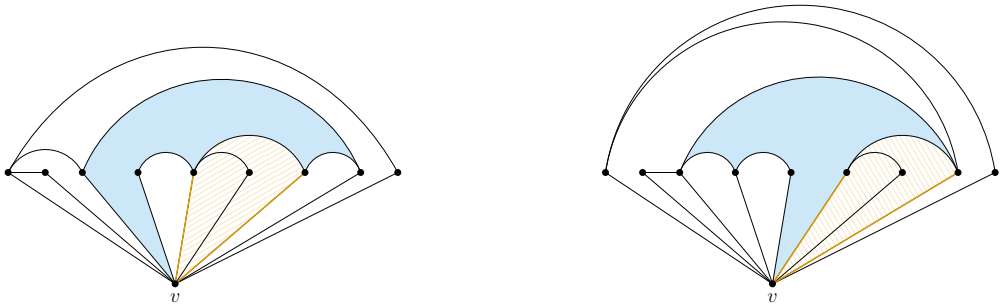


Figure 2 A one-sided (left) and a two-sided (right) umbrella highlighted in blue. The orange-shaded regions constitute pockets, the respective pocket edges are highlighted in orange.

The following property of umbrellas in ice-cream cone drawings can be shown recursively. The proof is similar to the one of Ruiz-Vargas for empty triangles [34, Theorem 1].

▶ **Lemma 2.4.** *Let f be an umbrella of size $k \geq 4$ in an ice-cream cone drawing of* $\text{star}(v) \cup T_v$. *Then each pocket of f contains at least one triangular face. Furthermore, there exists a triangular face that is not contained in any pocket of f.*

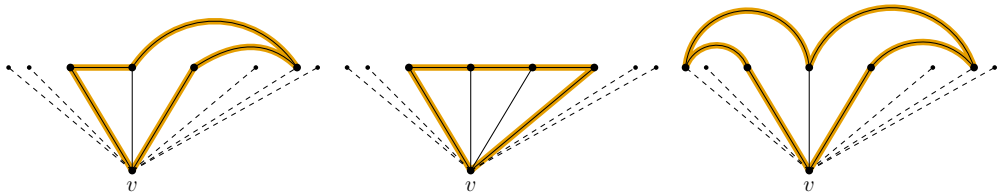


Figure 3 Combinations of neighboring faces of size 3 and 4 yielding empty plane 5- or 6-cycles.

▶ **Observation 2.5.** *Given a plane sub-drawing of* $\mathrm{star}(v) \cup T_v$*, the following cases immediately yield short empty cycles; see Figure 3. For a face of size four that shares exactly one edge with a neighboring triangular face or face of size four, the symmetric difference of the boundary edges forms an empty plane* 5*-cycle or* 6*-cycle, respectively. Similarly, a triangular face that has two neighboring triangular faces yields an empty plane* 5*-cycle.*

3 Proof of Theorem 1.2

Let $\mathcal{D}$ be a simple drawing of K_n for $n \geq 5$. We fix some vertex v and consider an ice-cream cone drawing of $\mathcal{D}$ at v. Let T_v be a spanning tree on $V \setminus \{v\}$ as in Lemma 2.3 so that $\mathcal{D}' := \mathrm{star}(v) \cup T_v$ is a plane sub-drawing of $\mathcal{D}$. If there exists some face of size 5 or 6 in $\mathcal{D}'$ then we are done. Therefore, assume that there are no faces of size 5 and 6. We make a case distinction on whether there exists a face of size $k \geq 7$.

First, assume that all faces are of size 3 and 4. By construction, $\mathcal{D}'$ has $2n-3$ edges. Let t be the number of triangular faces of $\mathcal{D}'$. Then $3t+4(n-1-t) = 2|E| = 2(2n-3)$. Solving for t yields that exactly two faces of $\mathcal{D}'$ are triangular. If the two triangular faces are edge-disjoint, then each of them has at least one neighboring face of size 4 with whom it only shares exactly one edge. By Observation 2.5, we obtain two empty plane 5-cycles incident to v. Otherwise, the triangular faces are neighboring and share exactly one edge. The remaining faces of size 4 all have to be nested as illustrated in Figure 4: Assume w.l.o.g. that the unbounded face is one of the two triangles. Then by Lemma 2.4, non-nestedness of the faces of size 4 would imply at least two triangles in disjoint respective pockets, a contradiction to having only two triangular faces in total. In particular, we obtain a plane double-star $\mathrm{star}(v) \cup \mathrm{star}(w)$ where w is the other end-vertex of the shared edge between the two triangular faces. By Lemma 2.2, there exist two empty plane 5-cycles incident to v.

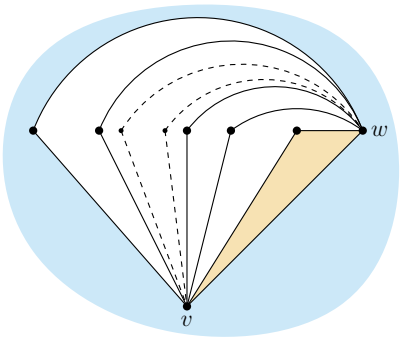


Figure 4 If $\mathcal{D}'$ has exactly two triangular faces (highlighted in blue and orange) that are neighboring and all the remaining faces are of size 4 then $\mathcal{D}'$ is a plane double-star.

Now assume there exists a face f of size at least 7 in $\mathcal{D}'$. We fix such a face f and consider a drawing $\mathcal{D}''$ that is strongly isomorphic to $\mathcal{D}'$ and in which f forms a one-sided umbrella. Then in $\mathcal{D}''$, the umbrella f has at least four consecutive pockets. We claim the following:

▶ Claim. Every three consecutive pockets of an umbrella contain an empty 5- or 6-cycle formed by neighboring faces, each of which has size 3 or 4.

To prove this claim, we iteratively find three consecutive pockets completely contained in the current ones, thus, containing fewer vertices, until we find a "good" combination of faces of size 3 or 4 along the lines of Observation 2.5, which concludes the proof.

4 Conclusion

We have shown that each vertex v in every simple drawing $\mathcal{D}$ of K_n is incident to at least one empty plane 5- or 6-cycle. The obvious next step is to see whether $\mathcal{D}$ always contains an empty plane 5-cycle *and* an empty plane 6-cycle, maybe even at every vertex. Under some conditions, namely, v is in a plane double-star $\text{star}(v) \cup \text{star}(w)$, we can actually guarantee two empty plane 5-cycles incident to v (Lemma 2.2). For $k = 3$ [16, Theorem 1.1] and $k = 4$ [26], it has been proven that each vertex in every simple drawing of K_n is incident to at least two empty plane k-cycles. Is this true for all k? A positive answer would immediately also imply Conjecture 1.1.

▶ **Open Problem 4.1.** *[26, Open Problem 3.52] Is each vertex in every simple drawing of K_n incident to at least two empty plane k-cycles for every $3 \leq k \leq (n-1)$ and one (on both sides empty) plane n-cycle?*

As auxiliary result for the proof of Theorem 1.2, we have also shown some properties of c-monotone drawings. We remark that Lemma 2.1 can be extended to drawings of general graphs: We can show (with a longer proof) that in every c-monotone drawing of an arbitrary graph G with $n \geq 3$ vertices, not all neighboring vertices can be joined by a long edge.

Acknowledgments. Research for this work was initiated during the 3^{rd} *Austrian Computational Geometry Reunion Meeting* in Strobl in August 2025. We thank the participants of the workshop for fruitful discussions and the productive environment. Further, we thank the anonymous reviewers for helpful comments.

References

1 Oswin Aichholzer, Alfredo García, Javier Tejel, Birgit Vogtenhuber, and Alexandra Weinberger. Twisted ways to find plane structures in simple drawings of complete graphs. In *Proceedings of the 38th International Symposium on Computational Geometry (SoCG 2022)*, pages 5:1–5:18, 2022. doi:10.4230/LIPIcs.SoCG.2022.5.

2 Oswin Aichholzer, Alfredo García, Javier Tejel, Birgit Vogtenhuber, and Alexandra Weinberger. Twisted ways to find plane structures in simple drawings of complete graphs. *Discrete & Computational Geometry*, 71(1):40–66, 2024. *Special Issue of few Selected Papers from SoCG 2022 [1].* doi:10.1007/s00454-023-00610-0.

3 Oswin Aichholzer, Thomas Hackl, Alexander Pilz, Pedro Ramos, Vera Sacristán, and Birgit Vogtenhuber. Empty triangles in good drawings of the complete graph. *Graphs and Combinatorics*, 31(2):335–345, 2015. doi:10.1007/s00373-015-1550-5.

4 Oswin Aichholzer, Joachim Orthaber, and Birgit Vogtenhuber. Separable drawings: Extendability and crossing-free Hamiltonian cycles. In *Proceedings of the 32nd International Symposium on Graph Drawing and Network Visualization (GD 2024)*, pages 34:1–34:17, 2024. doi:10.4230/LIPIcs.GD.2024.34.

5 Oswin Aichholzer, Joachim Orthaber, and Birgit Vogtenhuber. Towards crossing-free Hamiltonian cycles in simple drawings of complete graphs. *Computing in Geometry and Topology*, 3(2):5:1–5:30, 2024. doi:10.57717/cgt.v3i2.47.

6 Alan Arroyo, Dan McQuillan, R. Bruce Richter, and Gelasio Salazar. Convex drawings of the complete graph: Topology meets geometry. *Ars Mathematica Contemporanea*, 22(3):4:1–4:27, 2022. doi:10.26493/1855-3974.2134.ac9.

7 Helena Bergold, Stefan Felsner, Meghana M. Reddy, Joachim Orthaber, and Manfred Scheucher. Plane Hamiltonian cycles in convex drawings. In *Proceedings of the 40th*

International Symposium on Computational Geometry (SoCG 2024), pages 18:1–18:16, 2024. doi:10.4230/LIPIcs.SoCG.2024.18.

8 Helena Bergold, Stefan Felsner, Meghana M. Reddy, Joachim Orthaber, and Manfred Scheucher. Plane Hamiltonian cycles in convex drawings. *Discrete & Computational Geometry*, pages 1–28, 2025. *A conference version of this paper appeared at SoCG 2024 [7].* doi:10.1007/s00454-025-00752-3.

9 Helena Bergold, Joachim Orthaber, Manfred Scheucher, and Felix Schröder. Holes in convex and simple drawings. In *Proceedings of the 32nd International Symposium on Graph Drawing and Network Visualization (GD 2024)*, pages 5:1–5:9, 2024. doi:10.4230/LIPIcs.GD.2024.5.

10 Helena Bergold, Joachim Orthaber, Manfred Scheucher, and Felix Schröder. Holes in convex and simple drawings. *Journal of Graph Algorithms and Applications*, 29(3):23–38, 2025. *Special Issue of few Selected Papers from GD 2024 [9].* doi:10.7155/jgaa.v29i3.2999.

11 Helena Bergold and Manfred Scheucher. Investigating simple drawings of K_n using SAT, 2025. arXiv:2504.02650.

12 Paul Erdős. Some more problems on elementary geometry. *Australian Mathematical Society Gazette*, 5(2):52–54, 1978. URL: https://renyi.hu/~p_erdos/1978-44.pdf.

13 Paul Erdős and George Szekeres. A combinatorial problem in geometry. *Compositio Mathematica*, 2:463–470, 1935. URL: http://www.numdam.org/item/?id=CM_1935__2__463_0.

14 Jacob Fox and Benny Sudakov. Density theorems for bipartite graphs and related Ramsey-type results. *Combinatorica*, 29(2):153–196, 2009. doi:10.1007/s00493-009-2475-5.

15 Radoslav Fulek. Estimating the number of disjoint edges in simple topological graphs via cylindrical drawings. *SIAM Journal on Discrete Mathematics*, 28(1):116–121, 2014. *A conference version related to this paper appeared at SoCG 2013 [16].* doi:10.1137/130925554.

16 Radoslav Fulek and Andres J. Ruiz-Vargas. Topological graphs: Empty triangles and disjoint matchings. In *Proceedings of the 29th Annual Symposium on Computational Geometry (SoCG 2013)*, pages 259–266, 2013. doi:10.1145/2462356.2462394.

17 Alfredo García, Alexander Pilz, and Javier Tejel. On plane subgraphs of complete topological drawings. *Ars Mathematica Contemporanea*, 20(1):69–87, 2021. doi:10.26493/1855-3974.2226.e93.

18 Tobias Gerken. Empty convex hexagons in planar point sets. *Discrete & Computational Geometry*, 39(1):239–272, 2008. doi:10.1007/s00454-007-9018-x.

19 Heiko Harborth. Konvexe Fünfecke in ebenen Punktmengen. *Elemente der Mathematik*, 33:116–118, 1978. URL: https://eudml.org/doc/141217.

20 Heiko Harborth. Empty triangles in drawings of the complete graph. *Discrete Mathematics*, 191(1–3):109–111, 1998. doi:10.1016/S0012-365X(98)00098-3.

21 Heiko Harborth and Ingrid Mengersen. Drawings of the complete graph with maximum number of crossings. *Congressus Numerantium*, 88:225–228, 1992.

22 Andreas F. Holmsen, Hossein Nassajian Mojarrad, János Pach, and Gábor Tardos. Two extensions of the Erdős–Szekeres problem. *Journal of the European Mathematical Society*, 22(12):3981–3995, 2020. doi:10.4171/jems/1000.

23 Joseph D. Horton. Sets with no empty convex 7-gons. *Canadian Mathematical Bulletin*, 26(4):482–484, 1983. doi:10.4153/CMB-1983-077-8.

24 Carlos M. Nicolás. The empty hexagon theorem. *Discrete & Computational Geometry*, 38(2):389–397, 2007. doi:10.1007/s00454-007-1343-6.

25 Joachim Orthaber. *Crossing-Free Hamiltonian Cycles in Simple Drawings of the Complete Graph (And What We Found Along the Way).* Master's thesis, Graz University of Technology, Austria, 2022. doi:10.3217/012rz-4m722.

26 Joachim Orthaber. *Crossing-Free Structures: From Simple Drawings via Point Sets to Abstract Graphs*. PhD thesis, Graz University of Technology, 2025. doi:10.3217/b7r5e-3k596.

27 János Pach, József Solymosi, and Géza Tóth. Unavoidable configurations in complete topological graphs. *Discrete & Computational Geometry*, 30(2):311–320, 2003. *A conference version of this paper appeared at GD 2000 [28].* doi:10.1007/s00454-003-0012-9.

28 János Pach and Géza Tóth. Unavoidable configurations in complete topological graphs. In *Proceedings of the 8th International Symposium on Graph Drawing (GD 2000)*, pages 328–337, 2001. doi:10.1007/3-540-44541-2_31.

29 János Pach and Géza Tóth. Disjoint edges in topological graphs. In *Proceedings of the 7th Indonesia-Japan Joint Conference on Combinatorial Geometry and Graph Theory (IJCCGGT 2003)*, pages 133–140, 2005. doi:10.1007/978-3-540-30540-8_15.

30 János Pach and Géza Tóth. Disjoint edges in topological graphs. *Journal of Combinatorics*, 1(3):335–344, 2010. *A conference version of this paper appeared at IJCCGGT 2003 [29].* doi:10.4310/JOC.2010.v1.n3.a4.

31 Nabil H. Rafla. *The Good Drawings D_n of the Complete Graph K_n*. PhD thesis, McGill University, Montreal, 1988. URL: https://escholarship.mcgill.ca/concern/theses/x346d4920.

32 Gerhard Ringel. Extremal problems in the theory of graphs. In *Theory of Graphs and Its Applications: Proceedings of the Symposium Held in Smolenice in June 1963*, pages 85–90, 1964.

33 Andres J. Ruiz-Vargas. Empty triangles in complete topological graphs. In *Proceedings of the 7th Latin-American Algorithms, Graphs and Optimization Symposium (LAGOS 2013)*, pages 169–174, 2013. doi:10.1016/j.endm.2013.10.026.

34 Andres J. Ruiz-Vargas. Empty triangles in complete topological graphs. *Discrete & Computational Geometry*, 53(4):703–712, 2015. *Conference versions related to this paper appeared at SoCG 2013 [16] and LAGOS 2013 [33].* doi:10.1007/s00454-015-9671-4.

35 Andres J. Ruiz-Vargas. Many disjoint edges in topological graphs. In *Proceedings of the 8th Latin-American Algorithms, Graphs and Optimization Symposium (LAGOS 2015)*, pages 29–34, 2015. doi:10.1016/j.endm.2015.07.006.

36 Andres J. Ruiz-Vargas. Many disjoint edges in topological graphs. *Computational Geometry*, 62:1–13, 2017. *A conference version of this paper appeared at LAGOS 2015 [35].* doi:10.1016/j.comgeo.2016.11.003.

37 Andrew Suk. Disjoint edges in complete topological graphs. In *Proceedings of the 28th Annual Symposium on Computational Geometry (SCG 2012)*, pages 383–386, 2012. doi:10.1145/2261250.2261308.

38 Andrew Suk. Disjoint edges in complete topological graphs. *Discrete & Computational Geometry*, 49(2):280–286, 2013. *A conference version of this paper appeared at SCG 2012 [37].* doi:10.1007/s00454-012-9481-x.

39 Andrew Suk and Ji Zeng. Unavoidable patterns in complete simple topological graphs. In *Proceedings of the 30th International Symposium on Graph Drawing and Network Visualization (GD 2022)*, pages 3–15, 2023. doi:10.1007/978-3-031-22203-0_1.

40 Andrew Suk and Ji Zeng. Unavoidable patterns in complete simple topological graphs. *Discrete & Computational Geometry*, 73(1):79–91, 2025. *A conference version of this paper appeared at GD 2022 [39].* doi:10.1007/s00454-024-00658-6.

41 Ji Zeng. Note on disjoint faces in simple topological graphs. *Journal of Combinatorial Theory, Series B*, 171:28–35, 2025. doi:10.1016/j.jctb.2024.11.002.

Recognizing Subgraphs of Regular Tilings*

Eliel Ingervo[1] and Sándor Kisfaludi-Bak[2]

1 **Aalto University**
`eliel.ingervo@aalto.fi`
2 **Aalto University**
`sandor.kisfaludi-bak@aalto.fi`

Abstract

The $\{p, q\}$-tiling graph is the (finite or infinite) planar graph $T_{p,q}$ where all faces are cycles of length p and all vertices have degree q. We give algorithms for the problem of recognizing subgraphs and induced subgraphs of these tiling graphs, as follows.

- For $\frac{1}{p} + \frac{1}{q} > \frac{1}{2}$, these graphs correspond to the regular tilings of the sphere. These graphs are finite, thus recognizing their (induced) subgraphs can be done in constant time.
- For $\frac{1}{p} + \frac{1}{q} = \frac{1}{2}$, these graphs correspond to the regular tilings of the Euclidean plane. For the Euclidean square grid $T_{4,4}$ Bhatt and Cosmadakis (IPL 1987) showed that recognizing subgraphs is NP-hard, even if the input graph is a tree. We show that a simple divide-and conquer algorithm achieves a subexponential running time in all Euclidean tilings, and we observe that there is an almost matching lower bound in $T_{4,4}$ under the Exponential Time Hypothesis via known reductions.
- For $\frac{1}{p} + \frac{1}{q} < \frac{1}{2}$, these graphs correspond to the regular tilings of the hyperbolic plane. As our main contribution, we show that deciding if an n-vertex graph is isomorphic to a subgraph (or an induced subgraph) of the tiling $T_{p,q}$ can be done in quasi-polynomial ($n^{O(\log n)}$) time for any fixed q.

Our results for the hyperbolic case show that it has significantly lower complexity than the Euclidean variant, and it is unlikely to be NP-hard. The Euclidean results also show that the problem can be NP-hard even if the graph in question is a tree. Consequently, the known treewidth bounds for subgraphs of hyperbolic tilings do not lead to an efficient algorithm by themselves. Instead, we use convex hulls *within the tiling graph*, which have several desirable properties in hyperbolic tilings. Our key technical insight is that planar subgraph isomorphism can be computed via a dynamic program that builds a (canonical) sphere cut decomposition of a solution subgraph's convex hull.

Related Version The full version is available on ArXiv [15].

1 Introduction

Subgraph Isomorphism is the decision problem of whether a given *pattern graph* H is isomorphic to a subgraph of a given *host graph* G. In this paper, we are concerned with the variant of subgraph isomorphism where the host graph G is a fixed infinite planar graph that is highly regular, more precisely, comes from a regular tiling. For $p, q \geq 2$ the *$\{p, q\}$-tiling graph* is the (finite or infinite) planar graph $T_{p,q}$ where all faces are cycles of length p and all vertices have degree q. See [7, 6, 11, 1] for a more details on tilings. In particular, the graph $T_{6,3}$ corresponds to the Euclidean hexagonal tiling. In general, these graphs correspond to the so-called regular tilings of the three classical 2-dimensional geometries—spherical, Euclidean and hyperbolic—depending on whether $\frac{1}{p} + \frac{1}{q}$ is strictly greater than, equal to, or strictly less than $\frac{1}{2}$, respectively. Observe that $T_{p,q}$ is finite if and only if it is spherical; these tilings and their (induced) subgraphs can be recognized in constant time. Thus, we

* This work is supported by the Research Council of Finland, Grant 363444.

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

will focus on Euclidean and hyperbolic tilings. Note that we can assume H to be planar, as we can check planarity in a precomputaiton phase in linear time [13].

From this point onward, we will let n denote the number of vertices in the pattern graph H. The problem of recognizing subgraphs of Euclidean tilings can be seen as a natural graph drawing problem: given a planar graph, can it be drawn on a square, triangular, or hexagonal grid so that each of its edges correspond to a single grid edge? The problem was already studied in the eighties: Bhatt and Cosmadakis showed that subgraph isomorphism is NP-complete if $G = T_{4,4}$ is the infinite Euclidean grid. Furthermore, this is true even if H is a tree [3]. On the other hand, a randomized algorithm with running time $2^{O(\sqrt{|V(H)|}\log^2|V(H)|)}|V(G)|^{O(1)}$ has been proposed for the case when the host graph G is planar (or more generally, apex-minor free) and the pattern H is connected and has constant maximum degree [9].

The problem in hyperbolic tilings can also be motivated from the graph drawing/visualization perspective. Hyperbolic visualizations have been proposed for trees and other graphs with hierarchical structure [21, 22], and embeddings into low-dimensional hyperbolic spaces are generally recognized as practically useful both in the machine learning [23, 10, 26] and complex networks communities [20, 24, 12]. We set out to answer the following question.

▶ **Question 1.** *Is there a quasi-polynomial ($n^{\mathrm{poly}(\log n)}$) algorithm for $\{p,q\}$-Tiling (Induced) Subgraph Recognition when $\frac{1}{p}+\frac{1}{q}<\frac{1}{2}$?*

1.1 Our contribution.

Hyperbolic Tilings. Our main contribution concerns hyperbolic tilings. We answer Question 1 by giving a quasi-polynomial algorithm for any fixed q as follows.

▶ **Theorem 1.1.** *For any integers $p,q \geq 2$ with $\frac{1}{p}+\frac{1}{q}<\frac{1}{2}$ the $\{p,q\}$-Tiling Subgraph Recognition and $\{p,q\}$-Tiling Induced Subgraph Recognition can be solved in time $2^{O(q+q\log\frac{n}{p+q})}\cdot n^{O(1+\log\frac{n}{p+q})}$.*

In particular, the running time is $n^{O(\log n)}$ for any constant q or even $q = O(\log n)$. We get a polynomial algorithm if q is constant and $p = \Omega(n^{\varepsilon})$ for some constant $\varepsilon > 0$.

Due to the exponential expansion of the hyperbolic plane, a ball of radius r in any hyperbolic tiling contains $2^{\Omega(r)}$ vertices, thus when applying [9] in a ball of radius $\mathrm{diam}(H) \leq |V(H)|$ we get a host graph whose size is potentially exponential in n. Thus, we need a different approach. Using a general purpose subgraph isomorphism algorithms could seem appealing, however they cannot directly be used here as we cannot restrict our infinite host graph to polynomial size. A natural finite host graph would be to take a subgraph of $T_{p,q}$ with diameter n. Such a graph however has exponential size due to the exponential expansion of hyperbolic space. Our key idea is to use the (unknown) convex hull of of the embedded graph as a virtual host graph. We outline our approach as follows.

We aim for a dynamic programming algorithm that builds a solution mapping $\phi : V(H) \to T_{p,q}$. Note that having a small separator in H does not give an algorithm: indeed, after mapping the vertices of a separator S to $T_{p,q}$ with size $|S| = O(\log n)$, the resulting subproblems (embed each connected component of $H - S$ with the mapping ϕ already fixed on S) are not independent: two different subgraphs can overlap. Thus, the resulting subproblems must be separated geometrically. A natural choice is to use a so-called *sphere cut decomposition* [8] of H: this has been successful in the hyperbolic setting before [4]. A sphere cut separator (usually called a *noose*) of a plane graph H is a closed curve drawn in the plane that intersects the planar drawing of H in only $O(\log n)$ vertices. See Figure 2 (ii) for an example of three nooses in a sphere cut decomposition. Note, however, that as a

geometric curve, a sphere cut separator of a plane graph may require a description much more complex than the size of the corresponding vertex separator: indeed, in order to separate a tree embedded in the plane at a single vertex v, we must define a curve that avoids all branches and intersects the tree at only v (see Figure 1).

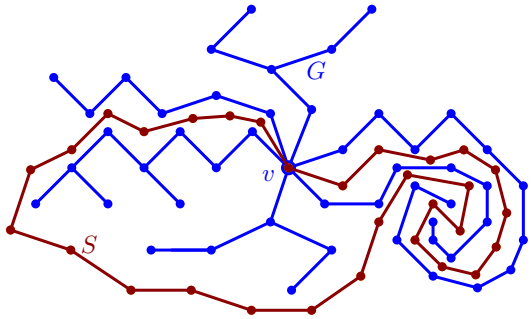


Figure 1 A plane graph G and a sphere cut separator S. The vertex separator corresponding to S has size 1, however the geometric curve S is much more complex.

Consequently, rather than finding a sphere cut decomposition of $\phi(H)$, we will build a sphere cut decomposition for the *convex hull* of $\phi(H)$. Crucially, the convex hull is defined with respect to the graph metric, as follows. The *interval* of a pair of vertices a, b in a graph G is the subgraph given by all shortest paths from a to b in G. A subgraph C of G is *convex* if for any $a, b \in V(C)$ we have that the interval of a, b in G is contained in C. The *convex hull* of a vertex set $X \subset V(G)$ is the intersection of all convex subgraphs that cover X.

It turns out that convex hulls in tiling graphs $T_{p,q}$ have several desirable properties. Recently, it was shown that the convex hull of the vertices of a connected n-vertex tiling graph has size $O(n)$ [18]. Consequently, any fixed solution embedding $\phi(H)$ of H has a sphere cut decomposition with curves that have complexity $O(\log n)$. (This requires normalizing the curves, similarly to [4].) Our dynamic programming algorithm embeds ever larger subgraphs of H (defined via a separator and its neighborhood on each side of the sphere cutting curve) into all possible sphere cuts of complexity $O(\log n)$. Unlike [4] where the goal was to get a solution for Independent Set, there are significant challenges remaining even after having an efficient way of enumerating all nooses: the embedding of H is being built and thus the convex hull of the embedded graph is not available to the algorithm.

We note here that most of our proofs use facts from hyperbolic geometry in a black-box fashion; our arguments are mostly relying on general properties of planar graphs. As a result, our algorithm is easily understood without any prior knowledge of hyperbolic geometry.

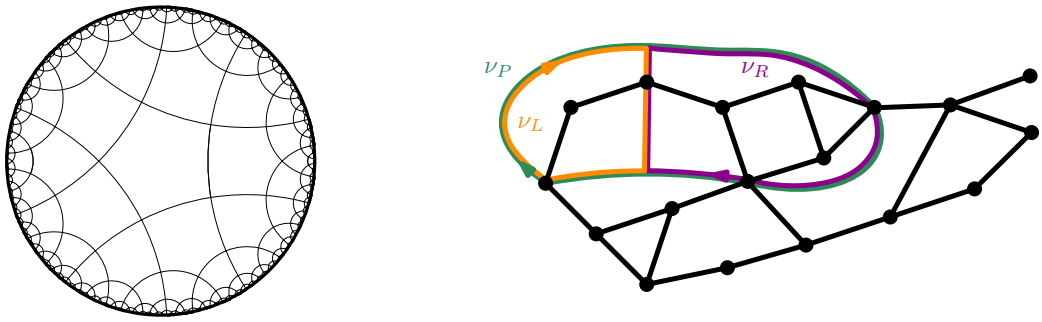
Euclidean Tilings. In the Euclidean setting, one can observe that rather than complicated nooses, it is sufficient to use a decomposition into rectangular regions. Compared to the randomized $2^{O(\sqrt{n}\log^2 n)}$ algorithm [9], a simple deterministic divide-and-conquer algorithm can shave a $\log n$ factor from the exponent in this special case.

▶ **Theorem 1.2.** *For any integers $p, q \geq 2$ with $\frac{1}{p} + \frac{1}{q} = \frac{1}{2}$ the $\{p, q\}$-Tiling (Induced) Subgraph Recognition can be solved in time $n^{O(\sqrt{n})} = 2^{O(\sqrt{n}\log n)}$.*

Moreover, we observe that a composition of known reductions (notably [3]) imply that the running time is nearly optimal for the square grid: under the Exponential Time Hypothesis (ETH) [14] there is no $2^{o(\sqrt{n})}$ algorithm for $\{4, 4\}$-Tiling Subgraph Recognition.

2 Preliminaries

$\{p,q\}$-Tiling (Induced) Subgraph Recognition. For two graphs $H = (V(H), E(H))$ and $G = (V(G), E(G))$, a *subgraph isomorphism* from H to G is an injective map $\phi : V(H) \to V(G)$ such that for each edge $uv \in E(H) \Rightarrow \phi(u)\phi(v) \in E(G)$. An *induced subgraph isomorphism* from H to G is a subgraph isomorphism $\phi' : V(H) \to V(G)$ such that for each edge $uv \notin E(H) \Rightarrow \phi'(u)\phi'(v) \notin E(G)$. The $\{p,q\}$-Tiling Subgraph Recognition problem asks, given a graph H, whether there exists a subgraph isomorphism from H to $T_{p,q}$. The $\{p,q\}$-Tiling Induced Subgraph Recognition variant asks the same with "subgraph" replaced by "induced subgraph."



(i) (ii)

Figure 2 (i) The hyperbolic $\{5,4\}$-tiling $\mathcal{T}_{5,4}$ in the Poincaré disk model. All angles are equal and all edges are (straight) line segments of the same hyperbolic length. (ii) Three nooses in a sphere cut decomposition. The nooses ν_L and ν_R are the children of ν_P.

Poincaré disk. We visualize the hyperbolic plane $\mathbb{H}^2$ using the *Poincaré disk model*, where all points of the hyperbolic plane are assigned to points in the Euclidean unit disk, and straight hyperbolic lines appear as circular arcs perpendicular to the Euclidean unit disk. See [2, 5, 16] for more information. Figure 2 (i) depicts $\mathcal{T}_{5,4}$ in this model.

3 Solving $\{p,q\}$-Tiling (Induced) Subgraph Recognition

In what follows, we assume $T_{p,q}$ to be a hyperbolic $\{p,q\}$-tiling graph, that is, $\frac{1}{p} + \frac{1}{q} < \frac{1}{2}$. To solve $\{p,q\}$-Tiling (Induced) Subgraph Recognition, we will divide the problem into smaller subproblems with closed oriented Jordan curves called *nooses* that intersect each face of a graph H at most once. We call $\mathrm{Reg}(\nu)$ the region that includes a noose ν and the component on its right. Note that when a noose is oriented in clockwise orientation, the interior and exterior of a noose correspond to the typical definition of the interior and exterior of a curve. We denote $T|_\nu$ the subgraph of $T_{p,q}$ given the vertices and edges in $\mathrm{Reg}(\nu)$. A *sphere cut decomposition* is a tree where each node corresponds to a noose of H and each non-leaf noose has two child nooses. See Figure 2 (ii).

▶ **Theorem 1.1.** *For any integers $p, q \geq 2$ with $\frac{1}{p} + \frac{1}{q} < \frac{1}{2}$ the $\{p,q\}$-Tiling Subgraph Recognition and $\{p,q\}$-Tiling Induced Subgraph Recognition can be solved in time $2^{O(q+q\log\frac{n}{p+q})} \cdot n^{O(1+\log\frac{n}{p+q})}$.*

Proof sketch. To solve this problem we use so called *normalized nooses*, which is a way to bound the number of nooses that intersect some set on vertices and faces. Normalized nooses alternate between vertices of $T_{p,q}$ and adjacent tile centers such that there is only $(pq)^k$ candidate nooses of length at most k. This is due to the fact that any vertex of $T_{p,q}$ has q adjacent tiles, and any tile center has p adjacent vertices of $T_{p,q}$. See Figure 3 (i). By alternating between vertices of $T_{p,q}$ and tile centers, a subcurve through the unbouded face of H could have a description of $O(n)$ vertices due to the exponential expansion of hyperbolic space. To bound the length of nooses, we have a characterization that uses rays and may use ideal points on the boundary of the Poinaré disk. That description takes only at most 4 vertices. See Figure 3 (ii)–(iv). For a subcurve through the unbounded face, we have a single way to draw the corresponding normalized subcurve, which uses rays and ideal edges on the ideal boundary. In hyperbolic tilings, Kisfaludi-Bak [17] and independently, Kopczynski [19] proved that n-vertex subgraphs of hyperbolic tilings have treewidth $O_{p,q}(\log n)$. Combining this with results from [4], [18] and [25], there is a (normalized) sphere cut decomposition of H where each noose has length $O(1 + \log \frac{n}{p+q})$.

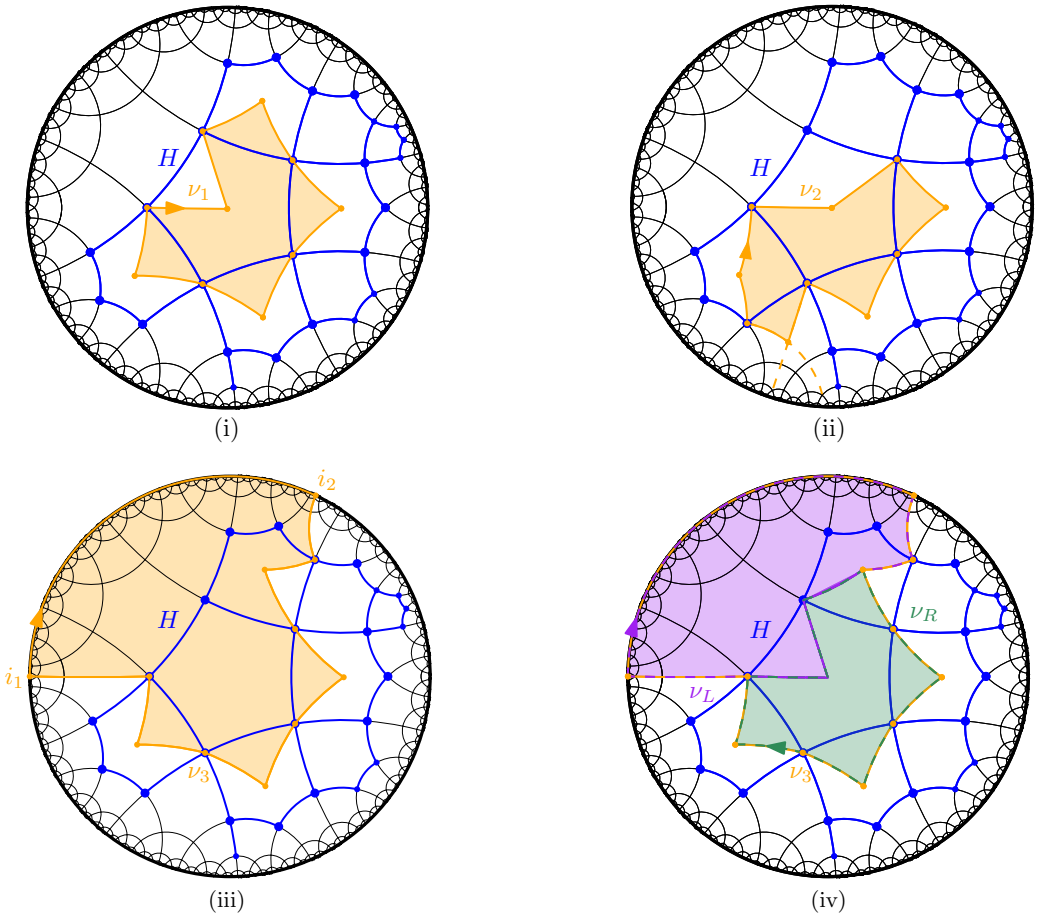


Figure 3 Each figure depicts a convex plane graph H embedded in $T_{5,4}$ (in black) as well as: (i) a normalized noose ν_1 of H, such that $\nu_1 = \nu_1^-$, (ii) a normalized noose ν_2 of H with pre-rays from two intersecting rays (the continuations of the rays are dashed), (iii) a normalized noose ν_3 with two ideal vertices i_1 and i_2, (iv) two child nooses ν_L, ν_R of ν_3.

Boundary assignments. We now introduce the key components of our algorithm. One of the difficulties in designing this algorithm is to assign some subgraph of H to a given noose: a graph H has $2^{|V(H)|}$ subgraphs. Notice that a vertex set B separating H is insufficient to define a subgraph, as we need to know what connected components of $H - B$ should fall in a given noose region. A *boundary assignment* for a graph H and candidate noose ν is a bijection ϕ from a subset of $V(H)$ to vertices of ν. A *boundary neighborhood constraint* is a vertex set $\widetilde{B} \subset N_H(B) \setminus B$, and it expresses the neighbors of B that will be mapped to the interior of the noose. Later, this set together with B will be used for identifying the subgraph of H we want to embed into $\mathrm{Reg}(\nu)$. We say that a triplet of a noose, a boundary assignment, and a boundary neighborhood constraint $(\nu, \phi, \widetilde{B})$ is *valid* if:

1. $\phi : B \to V(\nu)$ is a subgraph isomorphism from $H[B]$ to $T|_\nu$, and
2. there is a partition of $V(H - B)$ into A, A' such that there is no edge in $H - B$ from A to A', and $\widetilde{B} = N(B) \cap A$.

For a graph H, we want to enumerate valid triplets such that some sphere cut decomposition of H is in that collection. We compute all candidate nooses ν of length $O(1 + \log \frac{n}{p+q})$ and all boundary assignments of vertices of H to ν. For each of the at most q neighbors of a vertex assigned to ν, we assign that neighbor either to the interior of ν, that is in $\widetilde{B}$, or to the exterior of ν. That makes 2^q possible boundary neighborhood constraints per vertex. We prove that for an n-vertex graph H and the tiling $T_{p,q}$, there are $2^{O(q+q\log\frac{n}{p+q})} \cdot n^{O(1+\log\frac{n}{p+q})}$ valid triplets that can be enumerated in $2^{O(q+q\log\frac{n}{p+q})} \cdot n^{O(1+\log\frac{n}{p+q})}$ time.

Dynamic programming table. We solve $\{p, q\}$-Tiling (Induced) Subgraph Recognition with a dynamic programming algorithm where we process each valid triplet once. Recall that there are $2^{O(q+q\log\frac{n}{p+q})} \cdot n^{O(1+\log\frac{n}{p+q})}$ of them. The dynamic programming table entry $A(\nu, \phi, \widetilde{B})$ for a valid triplet $(\nu, \phi, \widetilde{B})$ is TRUE if and only if there exists a subgraph isomorphism $\overline{\phi}$ from $H(\nu, \phi, \widetilde{B})$ to $T|_\nu \subseteq T_{p,q}$ such that $\overline{\phi}|_B = \phi$ and $(\overline{\phi})^{-1}(V(\nu)) = B$, that is, the isomorphism $\overline{\phi}$ assigns only vertices of B to the noose, and this assignment is the same as ϕ. The entry is FALSE otherwise. We prove that for a triplet $(\nu, \phi, \widetilde{B})$, the problem is equivalent to deciding whether there are two valid triplets $(\nu_L, \phi_L, \widetilde{B}_L)$ and $(\nu_R, \phi_R, \widetilde{B}_R)$, that we have processed earlier that are *potential children* of $(\nu, \phi, \widetilde{B})$. On a high level this means that ν_L and ν_R could be child nooses of ν and ϕ, ϕ_L and ϕ_R, and $\widetilde{B}, \widetilde{B}_L$ and $\widetilde{B}_R$ are well-defined. When solving the induced variant of Tiling Subgraph Recognition, we cannot embed two non-neighboring vertices of H to neighboring vertices of $T_{p,q}$ when computing leaf triplets and whether two triplets are potential children of a third triplet.

Running time. We enumerate all $(pq)^{O(1+\log\frac{n}{p+q})}$ valid triplets in time $2^{O(q+q\log\frac{n}{p+q})} \cdot n^{O(1+\log\frac{n}{p+q})}$ and then we process them in order. Triplets are processed by matching pairs of nooses from a collection of at most $(pq)^{O(1+\log\frac{n}{p+q})}$ nooses, where each pair takes $n^{O(1)}$ time. This takes a total time of

$$2^{O(q+q\log\frac{n}{p+q})} \cdot n^{O(1+\log\frac{n}{p+q})} + (pq)^{O(1+\log\frac{n}{p+q})} \cdot n^{O(1)} = 2^{O(q+q\log\frac{n}{p+q})} \cdot n^{O(1+\log\frac{n}{p+q})}. \quad \blacktriangleleft$$

References

1 Colin Adams. *The tiling book: An introduction to the mathematical theory of tilings.* American Mathematical Society, 2023.

2 Riccardo Benedetti and Carlo Petronio. *Lectures on Hyperbolic Geometry.* Springer Science & Business Media, 1992. doi:10.1007/978-3-642-58158-8.

3 Sandeep N. Bhatt and Stavros S. Cosmadakis. The complexity of minimizing wire lengths in vlsi layouts. *Information Processing Letters*, 25(4):263–267, 1987. doi:10.1016/0020-0190(87)90173-6.

4 Thomas Bläsius, Jean-Pierre von der Heydt, Sándor Kisfaludi-Bak, Marcus Wilhelm, and Geert van Wordragen. Structure and independence in hyperbolic uniform disk graphs. In *41st International Symposium on Computational Geometry*, volume 332, pages 21:1–21:16, 2025. Full version: https://arxiv.org/abs/2407.09362. doi:10.4230/LIPICS.SOCG.2025.21.

5 James W. Cannon, William J. Floyd, Richard Kenyon, and Walter R. Parry. Hyperbolic geometry. In James W. Cannon, editor, *Non-Euclidean Geometry and Curvature*, chapter 2. American Mathematical Society, Providence, RI, 2017. Two-dimensional spaces. Vol. 3. doi:10.1090/mbk/110.

6 John H. Conway, Heidi Burgiel, and Chaim Goodman-Strauss. *The Symmetries of Things*. A K Peters / CRC Press, Boca Raton, FL, 2008. doi:10.1201/b21368.

7 H. S. M. Coxeter and W. O. J. Moser. *Generators and Relations for Discrete Groups*. Ergebnisse der Mathematik und ihrer Grenzgebiete. Springer, Berlin, 4th edition, 1980. doi:10.1007/978-3-662-21943-0.

8 Frederic Dorn, Eelko Penninkx, Hans L. Bodlaender, and Fedor V. Fomin. Efficient exact algorithms on planar graphs: Exploiting sphere cut decompositions. *Algorithmica*, 58(3):790–810, 2010. doi:10.1007/S00453-009-9296-1.

9 Fedor V. Fomin, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. Subexponential parameterized algorithms for planar and apex-minor-free graphs via low treewidth pattern covering. *SIAM J. Comput.*, 51(6):1866–1930, 2022. doi:10.1137/19M1262504.

10 Octavian Ganea, Gary Becigneul, and Thomas Hofmann. Hyperbolic entailment cones for learning hierarchical embeddings. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 1646–1655. PMLR, 10–15 Jul 2018.

11 Branko Grünbaum and Geoffrey Colin Shephard. *Tilings and patterns*. Courier Dover Publications, 1987.

12 Luca Gugelmann, Konstantinos Panagiotou, and Ueli Peter. Random hyperbolic graphs: Degree sequence and clustering. In *Automata, Languages, and Programming (ICALP 2012), Part II*, volume 7392 of *Lecture Notes in Computer Science*, pages 573–585. Springer, 2012. doi:10.1007/978-3-642-31585-5_51.

13 John Hopcroft and Robert Tarjan. Efficient planarity testing. *Journal of the ACM (JACM)*, 21(4):549–568, 1974.

14 Russell Impagliazzo and Ramamohan Paturi. On the complexity of k-sat. *J. Comput. Syst. Sci.*, 62(2):367–375, 2001. doi:10.1006/JCSS.2000.1727.

15 Eliel Ingervo and Sándor Kisfaludi-Bak. Recognizing subgraphs of regular tilings, 2026. URL: https://arxiv.org/abs/2603.06367.

16 Birger Iversen. *Hyperbolic Geometry*, volume 25 of *London Mathematical Society Student Texts*. Cambridge University Press, 1992. doi:10.1017/CBO9780511569333.

17 Sándor Kisfaludi-Bak. Hyperbolic intersection graphs and (quasi)-polynomial time. In Shuchi Chawla, editor, *Proceedings of the 2020 ACM-SIAM Symposium on Discrete Algorithms, SODA 2020*, pages 1621–1638. SIAM, 2020. doi:10.1137/1.9781611975994.100.

18 Sándor Kisfaludi-Bak, Tze-Yang Poon, and Geert van Wordragen. Shortest paths, convexity, and treewidth in regular hyperbolic tilings, 2025. arXiv:2510.26110.

19 Eryk Kopczynski. Hyperbolic minesweeper is in P. In Martin Farach-Colton, Giuseppe Prencipe, and Ryuhei Uehara, editors, *10th International Conference on Fun with Algorithms,*

FUN 2021, volume 157 of *LIPIcs*, pages 18:1–18:7. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPICS.FUN.2021.18.

20 Dmitri Krioukov, Fragkiskos Papadopoulos, Maksim Kitsak, Amin Vahdat, and Marián Boguñá. Hyperbolic geometry of complex networks. *Physical Review E*, 82(3):036106, 2010. doi:10.1103/PhysRevE.82.036106.

21 John Lamping and Ramana Rao. The hyperbolic browser: A focus+context technique for visualizing large hierarchies. *Journal of Visual Languages and Computing*, 7(1):33–55, 1996.

22 Tamara Munzner and Paul Burchard. Visualizing the structure of the world wide web in 3d hyperbolic space. In *Procedings of the 1995 Symposium on Virtual Reality Modeling Language, VRML 1995*, pages 33–38, 1995. doi:10.1145/217306.217311.

23 Maximilian Nickel and Douwe Kiela. Learning continuous hierarchies in the lorentz model of hyperbolic geometry. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018*, volume 80, pages 3776–3785, 2018.

24 Fragkiskos Papadopoulos, Maksim Kitsak, M. Ángeles Serrano, Marián Boguñá, and Dmitri Krioukov. Popularity versus similarity in growing networks. *Nature*, 489(7417):537–540, 2012. doi:10.1038/nature11459.

25 Neil Robertson and Paul D. Seymour. Graph minors. x. obstructions to tree-decomposition. *J. Comb. Theory B*, 52(2):153–190, 1991. doi:10.1016/0095-8956(91)90061-N.

26 Frederic Sala, Chris De Sa, Albert Gu, and Christopher Re. Representation tradeoffs for hyperbolic embeddings. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 4460–4469. PMLR, 10–15 Jul 2018.

Towards Computing Average Merge Tree Based on the Interleaving Distance*

Elena Farahbakhsh Touli[1], Ingrid Hotz[1], and Talha Bin Masood[1]

1 **Scientific Visualization Group, Department of Science and Technology (ITN), Linköping University, Sweden**
elena.farahbakhsh.touli@liu.se
ingrid.hotz@liu.se
talha.bin.masood@liu.se

Abstract

The interleaving distance is a key tool for comparing merge trees, which serve as topological summaries of scalar functions. In this work, we introduce a notion of an average merge tree for a pair of merge trees based on the interleaving distance. Since such an average is generally not unique, we propose a method to construct a representative average merge tree. We prove that the resulting merge tree satisfies a natural notion of averaging for the two given merge trees. Finally, we present illustrative examples to highlight the structure of the proposed average merge tree.

Related Version arXiv:2603.00783

1 Introduction

Tree-like structures are fundamental in many areas, including topological data analysis, shape comparison, hierarchical clustering, and phylogenetics. A merge tree is one such structure that encodes how the connected components of a space evolve as we sweep through the values of some scalar function defined on that space. Merge trees have applications in visualization, chemistry, physics, and beyond [17, 24, 10, 22, 18, 2, 16].

Several distances have been proposed to compare trees. One category of methods relate to the edit distance and the alignment distance, both of which have been shown to be SNP-hard (Strictly NP-hard) to compute [12, 25]. Another method that has been given much attention recently is the *interleaving distance*. This distance has been the subject of extensive research in a number of subsequent works [21, 4, 3, 6, 7, 1]. Even though it is NP-hard to compute the interleaving distance between merge trees, in [21] the authors proposed two fixed-parameter tractable (FPT) algorithms to compute the interleaving distance between merge trees. Interleaving distance was first introduced by the authors in [5] in the context of persistence modules and later extended to the *labeled interleaving distance* to quantify the similarity of labeled merge trees [11, 15, 19, 24, 23].

In this paper, we aim to statistically analyze merge trees. The central question we address is: given two merge trees, what is the average merge tree between them? Building on the conjecture of [13], an average merge tree with respect to the interleaving distance is known to exist but is generally not unique. Consequently, we propose a method to construct a representative merge tree that aggregates the structure of the two given merge trees.

* We thank Prof. Aasa Feragen (Denmark) for providing the data used in this paper's evaluation. This work was partially supported by the Swedish Research Council (VR) grant 2023-04806; the Swedish e-Science Research Center (SeRC); and the Wallenberg Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation.

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

The problem of defining or computing a representative tree from a collection of input trees has been investigated in [8] on the space of tree like shapes [9]. Later, the authors in [11, 24] proposed a method for computing a 1-center tree from a collection of labeled merge trees. In labeled merge trees, corresponding labels in the two trees must be matched to each other, and computing the interleaving distance between two labeled merge trees with identical labels can be done in polynomial time. However, in this study, we focus on computing the average tree for unlabeled merge trees using the interleaving distance, a problem that has not been previously studied.

In Section 2, we present the necessary definitions and background. Section 3 introduces our method for computing the average merge tree of two given merge trees. Section 4 provides illustrative examples. Finally, Section 5 concludes the paper with a summary and discussion.

2 Theoretical Foundations

This section is dedicated to reviewing the main background material and introducing the fundamental definitions that will be used throughout this paper.

2.1 Merge trees

Consider a connected topological space $\mathbb{M}$ and a continuous scalar function $h : \mathbb{M} \to \mathbb{R}$, which can be viewed as a height function. For each real number $a \in \mathbb{R}$, the sublevel set of h is defined as $\mathbb{M}_a = \{ x \in \mathbb{M} \mid h(x) \leq a \}$. As a increases, the corresponding sublevel sets expand. At $a = -\infty$, the sublevel set is empty. As the value of a grows, two types of events can occur. New connected components appear (at local minima), and existing components merge (at saddle points). Eventually, when $a = +\infty$, the entire space $\mathbb{M}$ is included. The merge tree T^h captures this process, describing the topological evolution of sublevel sets under a continuous sweep of the function value from $-\infty$ to $+\infty$. The result of this paper can be applied to the more general setting of merge trees, using the following definition.

▶ **Definition 2.1** ([21]). A merge tree is a rooted tree T^h with a continuous function h assigned to $|T|$, where $|T|$ is the underlying space of T. The function is strictly decreasing from the root to the leaves.

2.2 Interleaving Distance

The interleaving distance for merge trees is defined as follows:

▶ **Definition 2.2** ([14]). For two given merge trees T_1^f and T_2^g we say that a pair of compatible maps $\alpha : |T_1^f| \to |T_2^g|$ and $\beta : |T_2^g| \to |T_1^f|$ are ε-interleaved if the following four conditions for each $u \in |T_1^f|$ and $w \in |T_2^g|$ are satisfied:

$$\begin{aligned} &\text{(C1)} \quad g(\alpha(u)) = f(u) + \varepsilon && \text{(C3)} \quad f(\beta(w)) = g(w) + \varepsilon \\ &\text{(C2)} \quad \beta(\alpha(u)) = u^{2\varepsilon} && \text{(C4)} \quad \alpha(\beta(w)) = w^{2\varepsilon} \end{aligned}$$

where $u^{2\varepsilon}$ ($w^{2\varepsilon}$) is the unique ancestor of u (w) with the function value $f(u) + 2\varepsilon$ ($g(w) + 2\varepsilon$).

Conditions (C2) and (C4) imply that if two points $u_1, u_2 \in |T_1^f|$ are mapped to a single point $w \in |T_2^g|$, the function value of their nearest common ancestor (NCA) should not exceed $f(u_1) + 2\varepsilon$. In other words, this means that their NCA should not be too far from either of the points. For more information, see Figure 1.

The interleaving distance between two merge trees T_1^f and T_2^g is the infimum of all $\varepsilon \geq 0$ such that there are two ε-interleaved maps $\alpha : |T_1^f| \to |T_2^g|$ and $\beta : |T_2^g| \to |T_1^f|$.

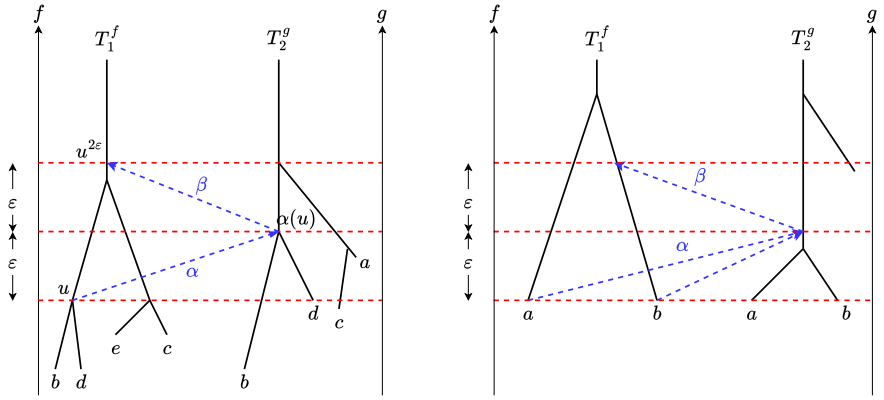


Figure 1 Left — A comparison of two mappings α and β, where vertex u is mapped to $\alpha(u)$, located ε above u, and $\beta(\alpha(u))$ is an ancestor of u, situated 2ε higher; Right — An example where the maps α and β are not ε-interleaved, since $\beta(\alpha(a)) \neq a^{2\varepsilon}$.

Later Touli and Wang [21] showed that the interleaving distance can be calculated using a single map (ε-*good map*) instead of two compatible maps.

▶ **Definition 2.3** ([21]). We write $u_1 \preceq u_2$ to mean that u_2 is either equal to u_1 or an ancestor of u_1. Given a continuous map $\alpha' : |T_1^f| \to |T_2^g|$ and a vertex $w \in |T_2^g|$, for a point w which is not in the $\mathrm{Img}(\alpha')$ we denote by w^A the nearest ancestor of w that lies in $\mathrm{Img}(\alpha')$. A continuous map $\alpha' : |T_1^f| \to |T_2^g|$ is called an ε-*good map* if it satisfies the following three conditions:

(S1) $g(\alpha'(u)) = f(u) + \varepsilon$,

(S2) If $\alpha'(u_1) \preceq \alpha'(u_2)$, then $u_1^{2\varepsilon} \preceq u_2^{2\varepsilon}$,

(S3) If $w \in |T_2^g| \setminus \mathrm{Img}(\alpha')$, then $g(w^A) - g(w) \leq 2\varepsilon$.

▶ **Theorem 2.4** ([21]). *For given pair of merge trees T_1^f and T_2^g, there exists an ε-good map $\alpha' : |T_1^f| \to |T_2^g|$, if and only if there exists a pair of ε-interleaved maps $\alpha : |T_1^f| \to |T_2^g|$ and $\beta : |T_2^g| \to |T_1^f|$.*

In this paper, we show that the definition of interleaving distance can be simplified further by modifying the second condition in Definition 2.3 to the following condition.

(S2′) If $\alpha'(u_1) = \alpha'(u_2)$, then $u_1^{2\varepsilon} = u_2^{2\varepsilon}$

▶ **Theorem 2.5.** *The second condition of an ε-good map can be replaced with condition (S2').*

Proof. It is straightforward to verify that (S2) implies (S2'). We now prove this by contradiction. Suppose there exists a continuous map $\alpha' : |T_1^f| \to |T_2^g|$ satisfying the conditions (S1), (S2'), and (S3) but not (S2). Assume for contradiction that there exist $u_1, u_2 \in T_1^f$ such that $\alpha'(u_1) \prec \alpha'(u_2)$, and $u_1^{2\varepsilon} \not\preceq u_2^{2\varepsilon}$, that is, the condition S2 is not satisfied. Then by the first condition (S1), $f(u_1) < f(u_2)$. This situation can only occur if $u_1^{2\varepsilon} |_< u_2^{2\varepsilon}$ [1].

[1] The notation $x |_< y$ (resp. $x |_= y$) indicates that x and y lie on different branches and that $f(x) < f(y)$ (resp. $f(x) = f(y)$).

Since the map α' is continuous, there must exist a point (u'_1) between u_1 and $NCA(u_1, u_2)$ such that $u'_1|_{=}u_2$ and $u'^{2\varepsilon}_1 \neq u^{2\varepsilon}_2$. Furthermore, u'_1 and u_2 are mapped to the same point in T^g_2, which contradicts (S2') (see Figure 2). ◀

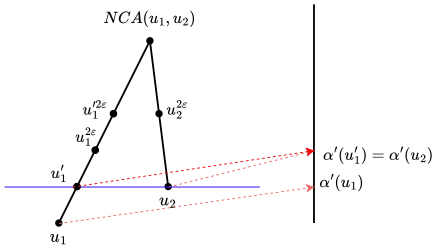


Figure 2 Illustrations of interleaving distance differences, u'_1 and u_2 in $|T^f_1|$ map to the same point in $|T^g_2|$, but their 2ε-extensions differ: $u'^{2\varepsilon}_1 \neq u^{2\varepsilon}_2$.

To compute the ε-good map $\alpha' : |T^f_1| \to |T^g_2|$, Touli and Wang [21] first construct the *augmented trees* $\hat{T}^f_1$ and $\hat{T}^g_2$ and then run a dynamic programming method. Two such methods are presented in [21].

3 Computing the Average Tree between Two Merge Trees

In this section, given a pair of merge trees T^f_1 and T^g_2 with $d_{ID}(T^f_1, T^g_2) \leq \varepsilon$, we aim to find a third merge tree T^k_3 such that

$$d_{ID}(T^f_1, T^k_3) \leq \frac{\varepsilon}{2} \quad \text{and} \quad d_{ID}(T^g_2, T^k_3) \leq \frac{\varepsilon}{2}.$$

The intuition behind the algorithm for finding the average tree using the interleaving distance is as follows (see Figure 3 for illustration):
Step I. We construct the augmented trees $\hat{T}^f_1$ and $\hat{T}^g_2$ as illustrated in [21], and find the ε-good map α'.
Step II. We shift the tree $\hat{T}^f_1$ upward by $\frac{\varepsilon}{2}$, constructing a new tree T^k_3 where

$$k(p) = f(p) + \tfrac{\varepsilon}{2}.$$

This guarantees that the distance between $\hat{T}^f_1$ and T^k_3 is equal to $\frac{\varepsilon}{2}$. Next, we modify and add some edges to T^k_3 so that its distance to $\hat{T}^g_2$ is also at most $\frac{\varepsilon}{2}$, while preserving the distance between $\hat{T}^f_1$ and T^k_3 as exactly $\frac{\varepsilon}{2}$.
Step III. We add edges to T^k_3 for the points that are outside the image of the ε-good map $\alpha' : |\hat{T}^f_1| \to |\hat{T}^g_2|$. This step is necessary because their distance to the nearest ancestor (which is in the image of α') can exceed ε, creating difficulties in constructing the map $\beta' : |T^k_3| \to |\hat{T}^g_2|$. In particular, for all points outside the image of β', their distance to the nearest ancestor w_A (which is in the image) must be at most ε.

To handle this, we first trim subtrees outside the image of α', ensuring that their distance to their root (the nearest ancestor in the image) is less than ε. Then, we attach the root of each trimmed subtree to the first vertex in ${\alpha'}^{-1}(w_A)$, chosen according to lexicographical order.
Step IV. We focus on vertices that are mapped to the same vertex by the map α'. According to the ε-good map (in its simpler form), for any pair of points u_1 and u_2 in T^k_3 that are

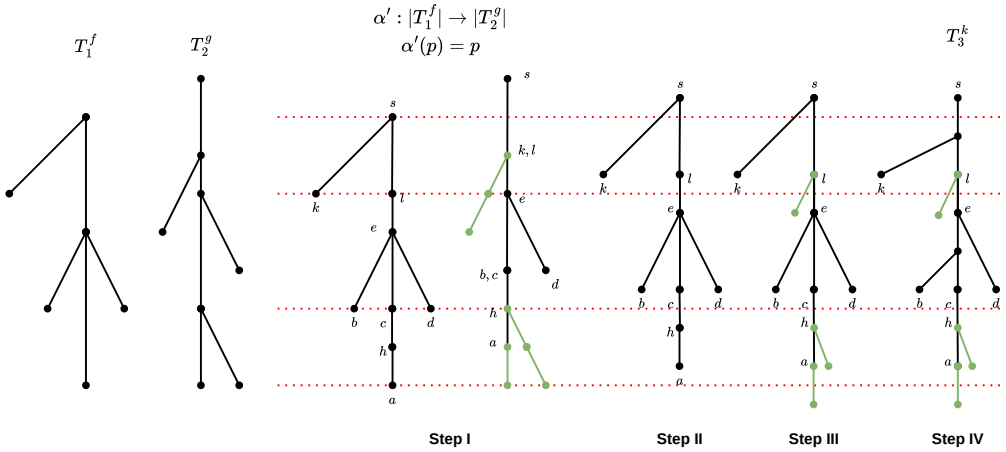


Figure 3 Two merge trees T_1^f and T_2^g are displayed on the left. The process consists of the following steps: **(Step I)** construct the augmented trees $\hat{T}_1^f$ and $\hat{T}_2^g$ and identify the ε-good map α'. **(Step II)** shift $\hat{T}_1^f$ upward by $\frac{\varepsilon}{2}$. **(Step III)** incorporate edges from $\hat{T}_2^g \setminus \mathrm{Img}(\alpha')$; and **(Step IV)** for nodes that are mapped to single point in $\hat{T}_2^g$ shift their nearest common ancestor downward by ε.

mapped to a single vertex in $\hat{T}_2^g$, their distance to the NCA is at most 2ε. However, we require this distance to be at most ε. To achieve this, we adjust the NCA, moving it to a position that is ε higher than the nodes u_1 and u_2. (For further illustration, see Figure 3.)

Refer to Algorithm 1 for detailed steps involved in building the average merge tree T_3^k. In the process, we construct a map

$$\gamma : |T_1^f| \cup |T'^g_{A,2}| \to |T_3^k|$$

where the map is surjective and $|T'^g_{A,2}| \subset |T_2^g|$ denotes the union of subtrees in $|T_2^g| \setminus \mathrm{Img}(\alpha')$ rooted at the points in the set A and trimmed such that every point in the subtree lies within a distance of at most ε from its root in A.

▶ Observation 1. The graph produced by Algorithm 1, together with the continuous function k defined therein, forms a merge tree. The running time of the algorithm is dominated by its first step. Using the slower algorithm in [21], this step runs in

$$O\big(n^3 2^{\mathsf{b}} \mathsf{b}^{\mathsf{b}+1}\big)\,,$$

where b is defined in Figure 4. In particular, if b is bounded, then the algorithm runs in polynomial time.

▶ **Theorem 3.1.** *If $d_{ID}(T_1^f, T_2^g) \le \varepsilon$, and T_3^k is constructed as described in Algorithm 1, then $d_{ID}(T_1^f, T_3^k) \le \frac{\varepsilon}{2}$ and $d_{ID}(T_2^g, T_3^k) \le \frac{\varepsilon}{2}$.*

Proof. Proof is in [20]. ◀

4 Illustrative Example

In this section, we consider two leaf datasets presented in [8] (see Figure 5). The data are provided as PNG images, which we annotate in order to extract the corresponding

Algorithm 1 Finding Mean Tree by Using Interleaving Distance

Step I: Construct the augmented trees $\hat{T}_1^f$ and $\hat{T}_2^g$, and determine the continuous ε-good map $\alpha' : |\hat{T}_1^f| \to |\hat{T}_2^g|$. Store this map. To store the map, it is sufficient to record the pairs (S, w) along the downward paths (from the roots to the first-level vertices), using *Slower FPT-Algorithm* or *the Faster FPT-Algorithm*, where $F(S, w) = 1$ from [21].
Step II: Construct a tree T_3^k such that $T_3 = \hat{T}_1$ (i.e. $V(T_3) = V(\hat{T}_1)$ and $E(T_3) = E(\hat{T}_1)$). For each $p \in |T_3^k|$, define $k(p) = f(p) + \frac{\varepsilon}{2}$.

(Constructing γ) For any $p \in |\hat{T}_1^f|$, set $\gamma(p) = p$.
Step III: If there exists a rooted subtree $|\hat{T}_{2,w^A}^g|$ of $|\hat{T}_2^g|$, rooted at w^A, such that w^A is the only vertex in this subtree contained in $\mathrm{Img}(\alpha')$:

1. Remove all points q in this subtree where $g(q) < g(w^A) - \varepsilon$.
2. Add the remaining part, $|\hat{T'}_{2,w^A}^g|$, to the first point in the lexicographic order of ${\alpha'}^{-1}(w^A)$ (denote it v_1).
3. Replace v_1 with w^A in $E(T_3)$ and $V(T_3^k)$.
4. Insert vertices and edges of $|\hat{T'}_{2,w^A}^g|$ into $E(T_3)$ and $V(T_3)$.

(Constructing γ) For each $p \in {T'}_{2,w^A}^g$, set $k(p) = g(p) - \frac{\varepsilon}{2}$. For every vertex $w \in |{T'}_{2,w^A}^g|$, set $\gamma(w) = w$. Also, set $\gamma(v_1) = w^A$.
Step IV: Sort mapped vertices by function values, starting from the smallest. For any $S = \{v_1, v_2, \ldots, v_n\} \subseteq |\hat{T}_1^f|$ such that $\alpha'(v_i) = w$ for all $v_i \in S$:

1. Identify their nearest common ancestor v_S.
2. If $k(v_S) > \varepsilon + k(v_i)$ for some v_i, then along the path from v_i to v_S, find all p such that $k(p) = \varepsilon + k(v_i)$. Label them ${v'}_1, {v'}_2, \ldots, {v'}_m$.
3. Preserve the path $\pi(v_S, v_1')$, merge ${v'}_1, {v'}_2, \ldots, {v'}_m$ into V_1' with the same neighbors as v_1'.
4. Add V_1' to $V(T_3)$ if missing; update $E(T_3)$ accordingly.
5. If extra vertices appear in $\pi(v_S, {v'}_i)$ for $i > 1$, insert them proportionally along $\pi(v_S, {v'}_1)$.

(Constructing γ) For any $p \in \pi(v_S, {v'}_i)$, set $\gamma(p) = p'$ where $p' \in \pi(v_S, {v'}_1)$ and $k(p) = k(p')$. Remove any p with ${v'}_i \prec p \prec v_S$ for $i > 1$, and redefine $\gamma(p) = q$ with the same function value.

merge trees. We use average geodesic distance as the scalar field for which the merge tree is computed. For this scalar field we get maxima at all the end points and minimum at approximately in the middle of the leaves. After computing the merge trees for each dataset, we then compute the average merge tree between them. The result is plotted in Figure 5.

5 Conclusions and Discussions

In this paper, we studied the problem of computing an average merge tree for a given pair of merge trees under the interleaving distance. This is the first time that an average merge tree, computed from a set of merge trees using the interleaving distance, has been constructed. We also demonstrated how our method computes the average merge tree of two leaf datasets. The time complexity of the algorithm is determined by the time required to compute the ε-good map, or equivalently, the interleaving distance between the two trees, and in some cases, FPT algorithms can be employed. In the future, we plan to extend this approach to find an average merge tree for a set of merge trees, while ensuring the method is efficient.

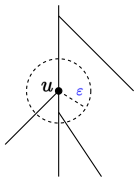


Figure 4 ε-ball around vertex u. The ε-degree around u is 10 which is the summation of all the nodes in the ε-ball.

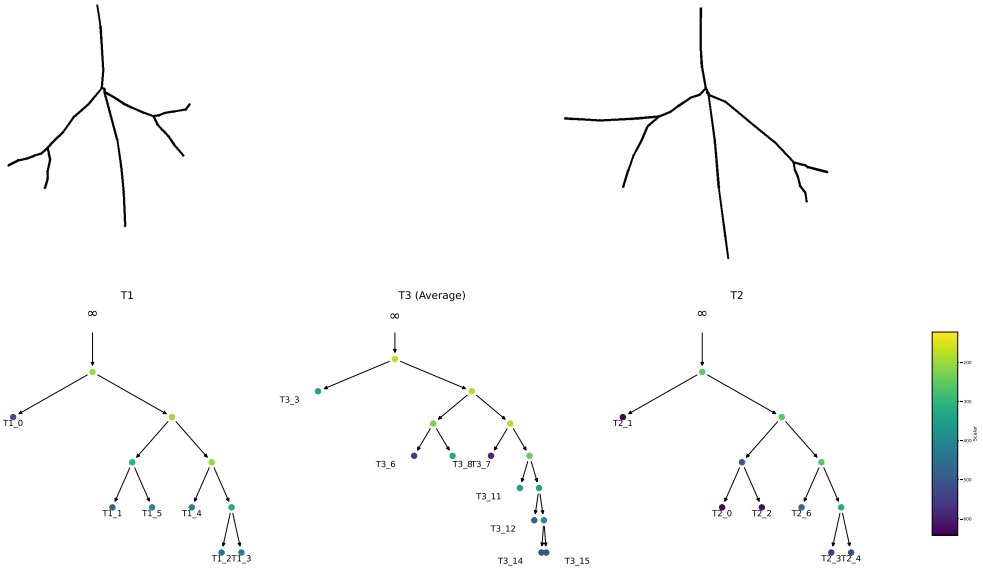


Figure 5 Top: the left and right trees show the first two leaf datasets from [8]. Bottom: the corresponding input merge trees T_1 and T_2, with the associated colorbar and their average merge tree. We display only the labels of the leaf nodes.

References

1 Pankaj K. Agarwal, Kyle Fox, Abhinandan Nath, Anastasios Sidiropoulos, and Yusu Wang. Computing the Gromov-Hausdorff distance for metric trees. *ACM Trans. Algorithms*, 14(2):24:1–24:20, 2018.

2 Silvia Biasotti, Leila De Floriani, Bianca Falcidieno, Patrizio Frosini, Daniela Giorgi, Claudia Landi, Laura Papaleo, and Michela Spagnuolo. Describing shapes by geometrical-topological properties of real functions. *ACM Computing Surveys (CSUR)*, 40(4):1–87, 2008.

3 Håvard Bakke Bjerkevik, Magnus Bakke Botnan, and Michael Kerber. Computing the interleaving distance is np-hard. *Foundations of Computational Mathematics*, 20(6):1237–1271, 2020. doi:10.1007/s10208-019-09442-y.

4 Håvard Bakke Bjerkevik and Magnus Bakke Botnan. Computational Complexity of the Interleaving Distance. In Bettina Speckmann and Csaba D. Tóth, editors, *34th International Symposium on Computational Geometry (SoCG 2018)*, volume 99 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 13:1–13:15, Dagstuhl, Germany, 2018. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. URL: https://drops.dagstuhl.de/entities/document/10.4230/LIPIcs.SoCG.2018.13, doi:10.4230/LIPIcs.SoCG.2018.13.

5 Frédéric Chazal, David Cohen-Steiner, Marc Glisse, Leonidas J. Guibas, and Steve Y. Oudot. Proximity of persistence modules and their diagrams. In *Proceedings of the Twenty-Fifth Annual Symposium on Computational Geometry (SoCG)*, pages 237–246, 2009. doi:10.1145/1542362.1542407.

6 Justin Curry, Haibin Hang, Washington Mio, Tom Needham, and Osman Berat Okutan. Decorated merge trees for persistent topology. *Journal of Applied and Computational Topology*, 6(3):371–428, 2022. doi:10.1007/s41468-022-00089-3.

7 Vin de Silva, Elizabeth Munch, and Amit Patel. Theory of interleavings on categories with a flow. *Theory and Applications of Categories*, 33(21):583–607, 2018. URL: http://www.tac.mta.ca/tac/volumes/33/21/33-21a.pdf.

8 Aasa Feragen, Søren Hauberg, Mads Nielsen, and François Lauze. Means in spaces of tree-like shapes. In *2011 International Conference on Computer Vision*, pages 736–746, 2011. doi:10.1109/ICCV.2011.6126311.

9 Aasa Feragen, François Lauze, Pechin Lo, Marleen de Bruijne, and Mads Nielsen. Geometries on spaces of treelike shapes. In Ron Kimmel, Reinhard Klette, and Akihiro Sugimoto, editors, *Computer Vision – ACCV 2010: 10th Asian Conference on Computer Vision, Revised Selected Papers, Part II*, volume 6493 of *Lecture Notes in Computer Science*, pages 160–173. Springer, 2010. doi:10.1007/978-3-642-19309-5_13.

10 Christoph Flamm, Ivo L. Hofacker, Peter F. Stadler, and Michael T. Wolfinger. Barrier trees of degenerate landscapes. *Zeitschrift für Physikalische Chemie*, 216(2):155–173, 2002. doi:10.1524/zpch.2002.216.2.155.

11 Ellen Gasparovic, Elizabeth Munch, Steve Oudot, Katharine Turner, Bei Wang, and Yusu Wang. Intrinsic interleaving distance for merge trees. *La Matematica*, 4:40–65, 2025. doi:10.1007/s44007-024-00143-9.

12 Tao Jiang, Lusheng Wang, and Kaizhong Zhang. Alignment of trees: An alternative to tree edit. *Theoretical Computer Science*, 143(1):137–148, 1995.

13 Fangfei Lan, Salman Parsa, and Bei Wang. Labeled interleaving distance for reeb graphs. *Journal of Applied and Computational Topology*, 8(8):2367–2399, 2024. doi:10.1007/s41468-024-00193-6.

14 Dmitriy Morozov, Kenes Beketayev, and Gunther H. Weber. Interleaving distance between merge trees. In *Workshop on Topological Methods in Data Analysis and Visualization: Theory, Algorithms and Applications*, 2013.

15 Elizabeth Munch and Anastasios Stefanou. The ℓ^∞-cophenetic metric for phylogenetic trees as an interleaving distance, 2018. URL: https://arxiv.org/abs/1803.07609, arXiv:1803.07609.

16 Dilip Mathew Thomas and Vijay Natarajan. Symmetry in scalar field topology. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2035–2044, 2011. doi:10.1109/TVCG.2011.236.

17 Dilip Mathew Thomas and Vijay Natarajan. Multiscale symmetry detection in scalar fields by clustering contours. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):2427–2436, 2014. doi:10.1109/TVCG.2014.2346332.

18 Signe Sidwall Thygesen, Alexei I. Abrikosov, Peter Steneteg, Talha Bin Masood, and Ingrid Hotz. Level of detail visual analysis of structures in solid-state materials. In *EuroVis Short Papers*, 2023.

19 Elena F. Touli and Talha Bin Masood. Efficient Heuristic Algorithms for Interleaving Distance between Merge Trees . In *2025 IEEE Workshop on Topological Data Analysis and Visualization (TopoInVis)*, pages 74–88, Los Alamitos, CA, USA, November 2025. IEEE Computer Society. URL: https://doi.ieeecomputersociety.org/10.1109/TopoInVis68599.2025.00012, doi:10.1109/TopoInVis68599.2025.00012.

20 Elena F. Touli, Ingrid Hotz, and Talha Bin Masood. Towards computing average merge tree based on the interleaving distance, 2026. URL: https://arxiv.org/abs/2603.00783, arXiv:2603.00783.

21 Elena F. Touli and Yusu Wang. FPT-algorithms for computing Gromov-Hausdorff and interleaving distances between trees. *Journal of Computational Geometry*, 13(1):89–124, 2022.

22 Florian Wetzels, Talha Bin Masood, Nanna Holmgaard List, Ingrid Hotz, and Christoph Garth. Exploring electron density evolution using merge tree mappings. In Christian Tominski, Manuela Waldner, and Bei Wang, editors, *EuroVis 2024 - Short Papers*. The Eurographics Association, 2024. doi:10.2312/evs.20241069.

23 Lin Yan, Talha Bin Masood, Farhan Rasheed, Ingrid Hotz, and Bei Wang. Geometry-aware merge tree comparisons for time-varying data with interleaving distances. *IEEE Transactions on Visualization and Computer Graphics*, 29(8):3489–3506, 2022.

24 Lin Yan, Yusu Wang, Elizabeth Munch, Ellen Gasparovic, and Bei Wang. A structural average of labeled merge trees for uncertainty visualization. *IEEE Transactions on Visualization and Computer Graphics*, 26(1):832–842, Jan 2020. doi:10.1109/TVCG.2019.2934242.

25 Kaizhong Zhang and Tao Jiang. Some max snp-hard results concerning unordered labeled trees. *Information Processing Letters*, 49(5):249–254, 1994. URL: https://www.sciencedirect.com/science/article/pii/0020019094900620, doi:10.1016/0020-0190(94)90062-0.

On quadrilaterals in higher order Voronoi diagrams

Andrea de las Heras-Parrilla[1,2], Clemens Huemer[2], and Javier Tejel[3]

1 **Universidad Francisco de Vitoria**
andrea.delasheras@ufv.es
2 **Universitat Politècnica de Catalunya**
clemens.huemer@upc.edu
3 **Universidad de Zaragoza**
jtejel@unizar.es

Abstract

Let S be a set of N points in the plane in general position. The Voronoi diagram of order k, denoted by $V_k(S)$, is a partition of the plane into faces, where all the points inside a face have the same set of k nearest points from S. In this paper, we study upper bounds on the maximum number of quadrilateral faces, denoted by $q_k(S)$, that $V_k(S)$ can contain. In particular, we show that for $2 \leq k \leq N-2$,

$$q_k(S) \leq \min\left\{ \frac{k(2N-k-1) - \sum_{j=0}^{k-1} e_j}{2}, \frac{(k-1)(2N-k) - \sum_{j=0}^{k-2} e_j}{2} \right\}$$

where e_j denotes the number of j-edges of S, and a j-edge is a half-plane defined by the oriented line through a pair of points of S that contains j points of S in its interior. Furthermore, we also show some other upper bounds and the tightness of the upper bounds in some cases.

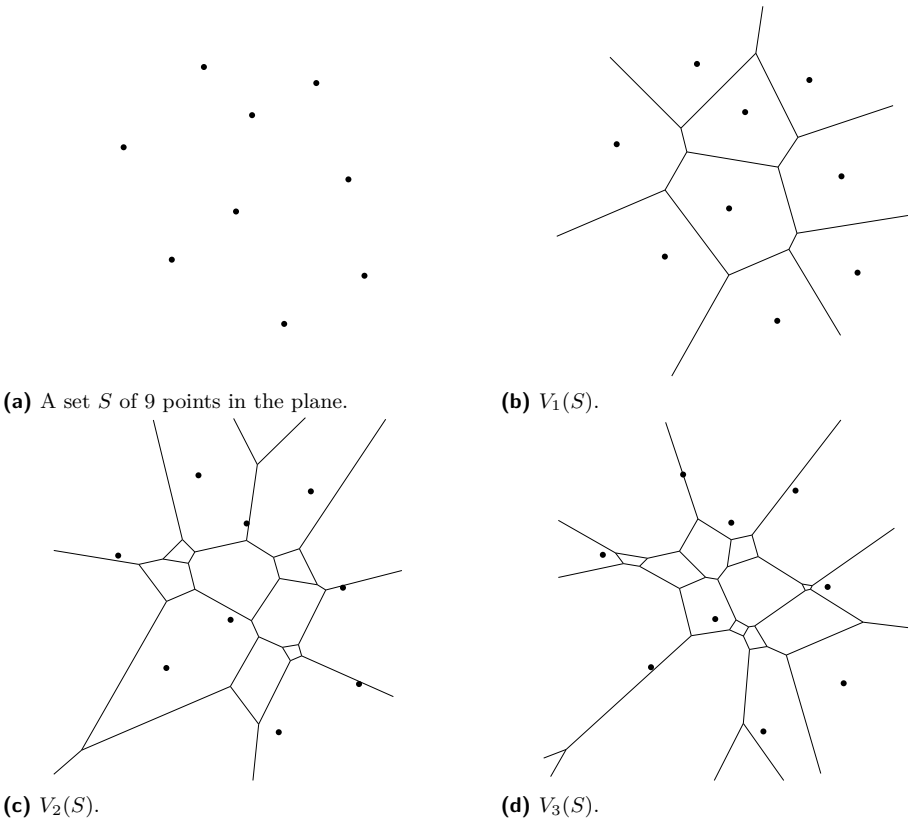
1 Introduction

Let S be a set of $N \geq 3$ points in general position in the plane, meaning that no three points of S are collinear, and no four points of S are cocircular. For $1 \leq k \leq N-1$, the *Voronoi diagram of order k* of S, denoted by $V_k(S)$, is a partition of the plane into regions (faces), where all the points inside a face have the same set of k nearest points from S. Figure 1a shows a set S of 9 points and Figures 1b, 1c, and 1d its corresponding Voronoi diagrams of order 1, 2, and 3, respectively.

Higher order Voronoi diagrams were introduced by Shamos and Hoey [15] and have a lot of applications in different fields, such as facility location, multivariate density estimation, classification, information retrieval systems, or geometric optimization. We refer the reader to [3, 4, 6, 7, 10, 16–19, 21, 23, 25–27] and the references therein for a broad overview on the combinatorial and algorithmic aspects of the Voronoi diagrams of order k, as well as their applications.

A formula for the number of faces in $V_k(S)$ is well known, see Section 2. It is also well known that for $k \geq 1$, every face (bounded or unbounded) in $V_k(S)$ is convex and that for $k \geq 2$, $V_k(S)$ cannot contain any triangular face (for instance, see [10, 25, 26]). Thus, for $k \geq 2$, every bounded face in $V_k(S)$ is a convex polygon of size at least 4. Then, a natural question concerns how many faces in $V_k(S)$ are quadrilaterals, pentagons, hexagons, etc., for arbitrary point sets S and arbitrary values of k. This problem of counting quadrilaterals, pentagons, hexagons, etc. in $V_k(S)$ has been extensively studied for $k = 1$ and for random point sets, especially with respect to a homogeneous Poisson point process [5, 8, 14, 24]. Most of these results are experimental and are summarized in [26]. For instance, in this setting,

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.



(a) A set S of 9 points in the plane.

(b) $V_1(S)$.

(c) $V_2(S)$.

(d) $V_3(S)$.

Figure 1 The Voronoi diagrams of orders 1, 2 and 3, for the point set S in Figure 1a.

the expected number of sides of a face of $V_k(S)$ is 6, for every $1 \leq k \leq N-2$ [24], and the estimated probability that a face is a quadrilateral is 0.1068, for $k = 2$ [8].

In a recent paper [9], Claverol et al. studied two special configurations of N points from this perspective. In particular, they showed for the first configuration that all bounded faces of $V_k(S)$ are hexagons for $k \geq \frac{N+3}{2}$, and that $V_k(S)$ does not contain a quadrilateral for $3 \leq k \leq \frac{N+1}{4}$. For the second configuration, they showed that for every $2 \leq k \leq N-2$, $V_k(S)$ contains exactly one quadrilateral. A similar study was carried out in [13] for $k = 1$, where the points are placed on the Archimedean spiral.

In this paper, for any point set S and any value of k, we focus on computing $q_k(S)$, the *number of quadrilaterals* that $V_k(S)$ contains, and on giving upper bounds for this number. For $k = 1$, it is not difficult to show that $0 \leq q_1(S) \leq N-4$ and that these bounds are tight. See Figure 2 for some examples. For $k \geq 2$, bounding $q_k(S)$, as well as proving the tightness of the bounds, is much more involved, as we show in Sections 3 and 4. In Section 2, we briefly describe several basic properties of higher order Voronoi diagrams.

2 Preliminaries

Given a set S of N points in the plane in general position, we denote the *number of vertices* in $V_k(S)$ by $nv_k(S)$. It is well-known [19, 26] that $nv_k(S) = (4k-2)N - 2k^2 - \left(\sum_{j=0}^{k-1} e_j + \sum_{j=0}^{k-2} e_j\right)$, where e_j denotes *the number of j-edges* of S, and a j-edge is a half-

Figure 2 Left: A set S of N points such that $q_1(S) = 0$. Right: A set S of N points in general position (after slightly perturbing the horizontal aligned points) such that $q_1(S) = N - 4$.

plane defined by the oriented line through a pair of points of S that contains j points of S in its interior [12]. Moreover, the number of unbounded faces (edges) in $V_k(S)$ is e_{k-1}, and an unbounded edge in $V_k(S)$ corresponds to an unbounded segment of the bisector of the two points of S that define a $(k-1)$-edge.

Given a face f in $V_k(S)$, let P_k be the subset of k points in S that are closest to every point in f. Each vertex of f is either the circumcenter of one point from P_k and two points from $S \setminus P_k$, whose circle encloses $k-1$ points from P_k and no other point of S, or the circumcenter of two points from P_k and one point from $S \setminus P_k$, whose circle encloses $k-2$ points from P_k and no other point of S. In the first case, we say that the vertex is of *type 1*, and in the second case, of *type 2*. Any vertex v of $V_k(S)$ is incident to exactly three edges, and the type of v is the same for the three faces that are incident to v.

Let $c_k(S)$ be the *number of circles through three points of S that enclose k other points of S*. It is known [19, 26] that $c_k(S) = (k+1)(2N-k-2) - \sum_{j=0}^{k} e_j$ and that *the number of vertices of type 1* in $V_k(S)$, denoted by $nv1_k(S)$, is c_{k-1}, and *the number of vertices of type 2* in $V_k(S)$, denoted by $nv2_k(S)$, is c_{k-2}.

For $2 \leq k \leq N-2$, every bounded face has at least two vertices of types 1 and 2, arranged so that not all vertices of the same type are consecutive [19]. Therefore, every quadrilateral has two vertices of each type, appearing in alternating order [10]. Furthermore, two quadrilaterals cannot be adjacent in $V_k(S)$ [9].

▶ **Lemma 2.1.** *Let S be a set of N points in the plane in general position. For $2 \leq k \leq N-2$, two quadrilaterals of $V_k(S)$ cannot be adjacent.*

In several of our constructions, we use cocircular points that are slightly perturbed later. The use of cocircular points has the following property [26].

▶ **Lemma 2.2.** *Given a set S of points in the plane, not in general position, suppose that a circle C with center p passes through 4 points of S and encloses exactly $k-2$ points of S. Then, after slightly perturbing the 4 points on C to obtain a new set S' of points, p gives rise to a quadrilateral in $V_k(S')$. We say that p is a degenerate quadrilateral in $V_k(S)$.*

The edges of $V_k(S)$ can be labeled as follows [10]. An edge that delimits a face of $V_k(S)$ is a (possibly unbounded) segment of the perpendicular bisector of two points i and j in S. Thus, an edge e of $V_k(S)$ has labels i and j, where the label i is on the side (half-plane) of the edge that contains i and the label j is on the other side. This method of labeling the edges induces the following property on the vertices [10]. For any vertex v, the number of different labels on the edges incident to v is only three, say labels i, j, l, and v is the center of the circle passing through the points i, j, and l. Moreover, the cyclic order of the edge labels

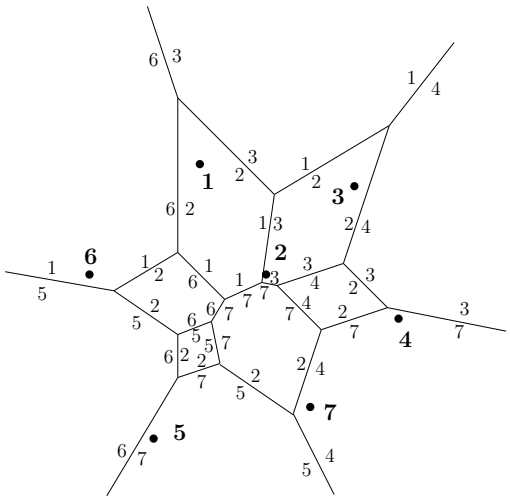


Figure 3 The edge labeling of $V_2(S)$, for this set of seven points.

around v is i, i, j, j, l, l, when v is of type 1, and i, j, l, i, j, l, when v is of type 2. Figure 3 shows the labeling of the edges of $V_2(S)$, for the set of seven points in the figure.

3 Upper bounds

In this section, we provide upper bounds for $q_k(S)$.

▶ **Theorem 3.1.** *Let S be a set of N points in general position in the plane. Then, for $2 \leq k \leq N-2$,*

$$q_k(S) \leq \min\left\{\frac{k(2N-k-1)-\sum_{j=0}^{k-1} e_j}{2}, \frac{(k-1)(2N-k)-\sum_{j=0}^{k-2} e_j}{2}\right\} \tag{1}$$

In addition, if $k \neq \frac{N}{2}$, then

$$q_k(S) \leq \frac{(4k-2)N-2k^2-\left(\sum_{j=0}^{k-1} e_j+\sum_{j=0}^{k-2} e_j\right)-\left\lceil\frac{e_{k-1}}{3}\right\rceil}{4} \tag{2}$$

Proof. For any k, two quadrilaterals of $V_k(S)$ cannot be adjacent by Lemma 2.1. Thus, each quadrilateral requires two vertices of each type, so

$$q_k(S) \leq \min\left\{\frac{nv1_k(S)}{2}, \frac{nv2_k(S)}{2}\right\} \tag{3}$$

Since $nv1_k(S) = c_{k-1}$ and $nv2_k(S) = c_{k-2}$, where $c_k(S) = (k+1)(2N-k-2)-\sum_{j=0}^{k} e_j$, Equation (1) readily follows. To prove Equation (2), let $nv'_k(S)$ be the number of vertices of $V_k(S)$ that do not belong to any quadrilateral of $V_k(S)$. Since two quadrilaterals cannot be adjacent in $V_k(S)$, then $q_k(S) = \frac{nv_k(S)-nv'_k(S)}{4}$.

We now show that $nv'_k(S) \geq \left\lceil\frac{e_{k-1}}{3}\right\rceil$. Let h be a $(k-1)$-edge of S, and let x and y be the two points of S defining h. Then, the unbounded edge of $V_k(S)$ corresponding to h is on the bisector b_{xy} defined by x and y. Note that the only option for two unbounded edges to belong to the same bisector b_{xy} is that x and y define two $(k-1)$-edges. This is only possible

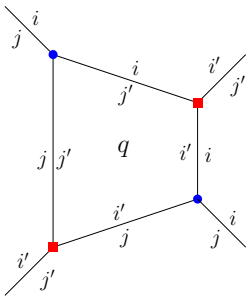


Figure 4 Edges not in q that are adjacent to opposite vertices of q belong to the same bisector. Vertices of type 1 are blue discs and vertices of type 2 are red squares.

when $k-1=\frac{N}{2}-1$ ($\frac{N}{2}-1$ points of S are to the left of the oriented line passing through x and y, and other $\frac{N}{2}-1$ points of S are to the right of this oriented line). As a consequence, if $k\neq\frac{N}{2}$, then a bisector cannot define more than one unbounded edge of $V_k(S)$.

Assume that $k\neq\frac{N}{2}$. We claim that if b_{xy} is the bisector defining one of the unbounded edges in $V_k(S)$, then there is a vertex v of $V_k(S)$ on b_{xy} such that v does not belong to any quadrilateral in $V_k(S)$. To prove this claim, we use the following fact that easily follows from the labeling of the edges of $V_k(S)$ around the vertices of a quadrilateral (see Figure 4 for an illustration): If q is a quadrilateral in $V_k(S)$, then the edges of $V_k(S)$ not belonging to q that are incident to two opposite vertices of q must belong to a same bisector b defined by two points of S. Thus, starting at the unbounded edge belonging to b_{xy}, we explore in order the segments of b_{xy} that are edges of $V_k(S)$. By the preceding observation about the edges of a quadrilateral, each time we reach a quadrilateral using a segment of b_{xy}, we leave the quadrilateral using another segment on b_{xy}. Thus, as a bisector cannot contain two unbounded edges, when exploring in order the segments of b_{xy} that belong to $V_k(S)$, we necessarily have to reach a vertex v not belonging to any quadrilateral, as claimed.

Since the degree of each vertex in $V_k(S)$ is three and a bisector cannot define more than one unbounded edge, then $nv'_k(S)\geq\lceil\frac{e_{k-1}}{3}\rceil$. Therefore, we have

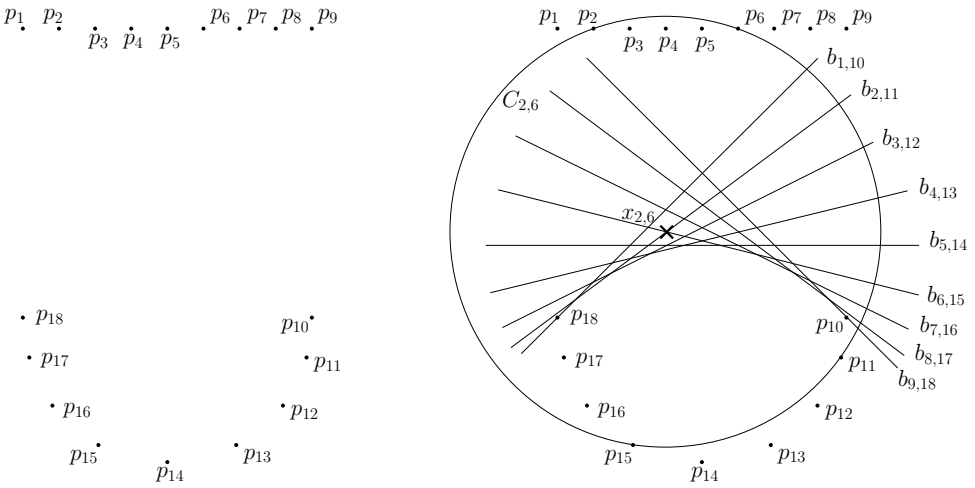
$$q_k(S)\leq\frac{nv_k(S)-\lceil\frac{e_{k-1}}{3}\rceil}{4}=\frac{(4k-2)N-2k^2-\left(\sum_{j=0}^{k-1}e_j+\sum_{j=0}^{k-2}e_j\right)-\lceil\frac{e_{k-1}}{3}\rceil}{4}$$

and Equation (2) holds. ◀

We note that bounds on $q_k(S)$ only in terms of N and k can be obtained by bounding the number of j-edges. It is well known that for a set S of points, the number of $(k-1)$-edges is the number of k-sets, where a k-set is a subset T of S of cardinality k such that T can be separated from $S\setminus T$ by a line. It is also well known [2, 22] that the number of $(\leq k)$-sets is at least $3\binom{k+1}{2}$, for $0\leq k\leq N/2-1$, hence $\sum_{j=0}^{k}e_j\geq 3\binom{k+2}{2}$. Further bounds on $\sum_{j=0}^{k}e_j$ are summarized in [1]. As a side remark, we mention that the number of quadrilaterals with vertices in S also depends on the number of k-edges [2, 22], and the prominent open problem of giving a tight asymptotic upper bound on the number of k-sets [11].

4 Tightness of the upper bounds

We first show that if $k=\frac{N}{2}$, the upper bound for $q_k(S)$ given in Theorem 3.1 is tight.



(a) The point set S when $m = 4$.

(b) The bisectors $b_{i,n+i}$ and the circle $C_{2,6}$ that passes through the four points p_2, p_6, p_{9+2} and p_{9+6}.

Figure 5 Building a point set S with $2n$ points in general position such that $q_n(S) = \frac{(n-1)n}{2}$.

▶ **Theorem 4.1.** *Let S be a set of $N = 2n$ points in general position in the plane. Then, $q_n(S) \leq \frac{n(n-1)}{2}$ and this bound is tight.*

Proof. We use the well-known relation $c_k + c_{N-k-3} = 2(k+1)(N-k-2)$, which holds for every set of N points and every $0 \leq k \leq N-3$, see e.g. [20]. Then, for $N = 2n$ and $k = n$, we have $nv1_k(S) + nv2_k(S) = c_{n-1} + c_{n-2} = c_{k-1} + c_{N-k-2} = 2k(N-k-1) = 2n(n-1)$. Thus,

$$\min\left\{\frac{nv1_k(S)}{2}, \frac{nv2_k(S)}{2}\right\} \leq \frac{n(n-1)}{2},$$

and the upper bound on $q_n(S)$ follows from Inequality (3).

It remains to provide a set S of $N = 2n$ points with $q_n(S) = \frac{n(n-1)}{2}$. Assume that n is odd, so $n = 2m+1$ for some $m \geq 1$. Allowing aligned and cocircular points, we first build a set $S = \{p_1, \ldots, p_{2n}\}$ of $2n$ points in convex position as follows. The first n points are on the horizontal line $y = 0$, with coordinates $p_i = (-m-1+i, 0)$, for $i = 1, \ldots, n$. The remaining n points are on a circle with center $(0, -2m)$ and radius m. The coordinates of these n points are $p_{n+m+j+1} = \left(-\frac{2m^2 j}{m^2+j^2}, -\frac{m(3m^2+j^2)}{m^2+j^2}\right)$, for $j = -m, \ldots, m$; see Figure 5a.

For $i = 1, \ldots, n$, let $b_{i,n+i}$ be the bisector defined by points p_i and p_{n+i}, and consider the arrangement of the n bisectors; see Figure 5b. By the construction of S, note that the number of crossing points in the arrangement is $\frac{n(n-1)}{2}$. For $i < j$, we denote by $x_{i,j}$ the crossing point between $b_{i,n+i}$ and $b_{j,n+j}$.

By the definition of S, for any $i < j$, the four points p_i, p_j, p_{n+i} and p_{n+j} are cocircular, lying on a circle $C_{i,j}$ with center $x_{i,j}$. This can be checked by elementary, but a bit lengthy, calculation, which we omit here. In addition, the points $p_{i+1}, \ldots, p_{j-1}$ are necessarily inside $C_{i,j}$, as well as the points $p_{n+1}, \ldots, p_{n+i-1}$ and the points $p_{n+j+1}, \ldots, p_{2n}$. That is, for any pair $i < j$, $C_{i,j}$ contains $n-2$ points of S; see Figure 5b for an illustration.

Thus, the set of $\frac{n(n-1)}{2}$ crossing points $x_{i,j}$ is a set of degenerate quadrilaterals when computing $V_n(S)$. Therefore, by Lemma 2.2, if we slightly perturb the points of S to obtain a new set of points S' in general position, without aligned or cocircular points, then $q_n(S') \geq \frac{n(n-1)}{2}$. As a consequence, there are sets S of $N = 2n$ points in general position

such that $q_n(S) = \frac{n(n-1)}{2}$. Finally, if $n = 2m$ is even, we only need to remove the points p_n and p_{2n} from S (the initial point set with $4m + 2$ points) and argue similarly. ◀

When $k = 2$, we have from Theorem 3.1, that $q_2(S) \leq N - 1 - \frac{e_0}{2}$ for any set S of N points, where $e_0 = |CH(S)|$ is the number of points of S on the boundary of $CH(S)$, the convex hull of S. We can prove the tightness of this upper bound in some cases.

▶ **Theorem 4.2.** *Given positive integers $e_0 \geq 3$ and $m \geq 1$, there exist sets S of $N = m \cdot e_0$ points in general position in the plane such that $|CH(S)| = e_0$ and $q_2(S) = N - 1 - \lceil \frac{e_0}{2} \rceil$.*

Figure 6 illustrates the cases $e_0 = 3$ and $e_0 = 4$. Since there are cocircular points in these examples, some quadrilaterals that appear in $V_2(S)$ are degenerate. By Lemma 2.2, these degenerate quadrilaterals give rise to quadrilaterals, after slightly perturbing the points.

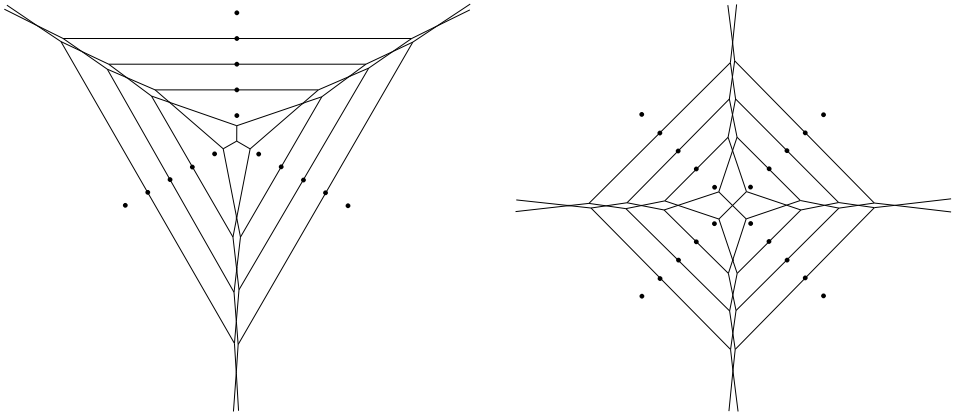


Figure 6 Building point sets S with N points in general position such that $q_2(S) = N - 1 - \lceil \frac{e_0}{2} \rceil$.

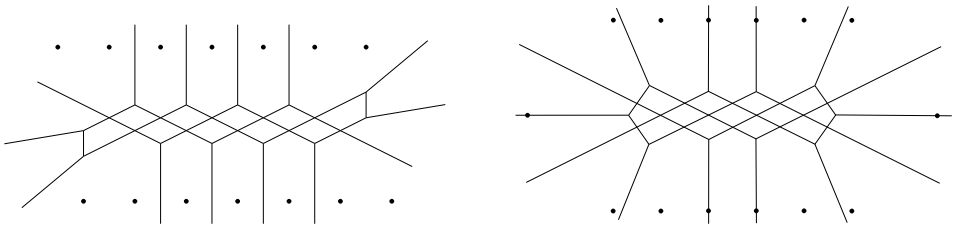


Figure 7 Left: A point set S of N points in convex position for which $q_3(S) = N - 4$. Right: A point set S of N points in convex position for which $q_4(S) = \frac{3}{2}N - 7$.

For a set S of N points in convex position, the upper bound on $q_k(S)$ given in Theorem 3.1 simplifies, for small values of k, to $q_k(S) \leq \frac{(k-1)(N-k)}{2}$ because $e_j = N$ for any j. This upper bound can be slightly improved, as the following theorem shows.

▶ **Theorem 4.3.** *Given a positive integer k, let $N = kn + r$, for some positive integers $n \geq 3$ and $1 \leq r < k$. Let S be a set of N points in convex position in the plane. If k is odd, or if k and N are even, then $q_k(S) \leq \frac{(k-1)(N-k)}{2} - 1$.*

For $k = 3$ and $k = 4$, previous theorem yields the upper bounds $N - 4$ and $\frac{3}{2}N - 7$, respectively. These bounds are tight in several cases, as illustrated in Figure 7. Note that some quadrilaterals that appear in $V_3(S)$ (left) and $V_4(S)$ (right) are degenerate.

▶ **Theorem 4.4.** *For $N > 9$ with N not a multiple of 3, there exist sets S of N points in convex position for which $q_3(S) = N - 4$. For $N > 12$ with N even and not a multiple of 4, there exist sets S of N points in convex position for which $q_4(S) = \frac{3}{2}N - 7$.*

Acknowledgments. The authors acknowledge the support of the project PID2023-150725NB-I00, funded by MICIU/AEI/10.13039/501100011033. Clemens Huemer and Andrea de las Heras-Parrilla also acknowledge the support of Gen. Cat. DGR 2021-SGR-00266. Javier Tejel also acknowledges the support of the project E41-23R, funded by Gobierno de Aragón.

References

1 Bernardo M. Ábrego, Silvia Fernández-Merchant, and Gelasio Salazar. The rectilinear crossing number of kn: Closing in (or are we?). In János Pach, editor, *Thirty Essays on Geometric Graph Theory*, pages 5–18. Springer New York, New York, NY, 2013. doi:10.1007/978-1-4614-0110-0_2.

2 Bernardo M. Ábrego and Silvia Fernández-Merchant. A lower bound for the rectilinear crossing number. *Graphs and Combinatorics*, 21:293–300, 2005. doi:10.1007/s00373-005-0612-5.

3 Franz Aurenhammer. Voronoi diagrams-a survey of a fundamental geometric data structure. *ACM Comput. Surv.*, 23(3):345–405, 1991. doi:10.1145/116873.116880.

4 Franz Aurenhammer and Otfried Schwarzkopf. A simple on-line randomized incremental algorithm for computing higher order Voronoi diagrams. *International Journal of Computational Geometry and Applications*, 2(2):363–381, 1992. doi:10.1142/S0218195992000214.

5 Marshall W. Bern, David Eppstein, and Frances Yao. The expected extremes in a Delaunay triangulation. *International Journal of Computational Geometry & Applications*, 01(01):79–91, 1991. doi:10.1142/S0218195991000074.

6 Cecilia Bohler, Panagiotis Cheilaris, Rolf Klein, Chih-Hung Liu, Evanthia Papadopoulou, and Maksym Zavershynskyi. On the complexity of higher order abstract Voronoi diagrams. *Computational Geometry*, 48(8):539–551, 2015. doi:10.1016/j.comgeo.2015.04.008.

7 Cecilia Bohler, Rolf Klein, and Chih-Hung Liu. An efficient randomized algorithm for higher-order abstract Voronoi diagrams. *Algorithmica*, 81(6):2317–2345, 2019. doi:10.1007/s00453-018-00536-7.

8 Georges Le Caer and John S. Ho. The Voronoi tessellation generated from eigenvalues of complex random matrices. *Journal of Physics A: Mathematical and General*, 23(14):#3279, 1990. doi:10.1088/0305-4470/23/14/025.

9 Mercè Claverol, Andrea de las Heras-Parrilla, and Clemens Huemer. Special constructions to understand the structure of higher order Voronoi diagrams. In *Book of Abstracts of the 20th Spanish Meeting on Computational Geometry*, pages 45–48, 2023. URL: https://egc23.web.uah.es/wp-content/uploads/2024/02/EGC2023-Booklet.pdf.

10 Mercè Claverol, Andrea de las Heras-Parrilla, Clemens Huemer, and Alejandra Martínez-Moraian. The edge labeling of higher order Voronoi diagrams. *Journal of Global Optimization*, 90(3):515–549, 2024. doi:10.1007/s10898-024-01386-0.

11 T. K. Dey. Improved bounds for planar k -sets and related problems. *Discrete and Computational Geometry*, 19:373–382, 1998. doi:10.1007/PL00009354.

12 Paul Erdös, László M. Lovász, Andrew Simmons, and Ernst G. Straus. Chapter 13 - Dissection Graphs of Planar Point Sets. In Jagdish N. Srivastava, editor, *A Survey of Combinatorial Theory*, pages 139–149. North-Holland, Netherlands, 1973. doi:10.1016/B978-0-7204-2262-7.50018-1.

13 Mark Frenkel, Irina Legchenkova, Nir Shvalb, Shraga Shoval, and Edward Bormashenko. Voronoi diagrams generated by the Archimedes spiral: Fibonacci numbers, chirality and aesthetic appeal. *Symmetry*, 15(3):#746, 2023. doi:10.3390/sym15030746.

14 Anthony L. Hinde and Robert Edwards Miles. Monte carlo estimates of the distributions of the random polygons of the Voronoi tessellation with respect to a Poisson process. *Journal of Statistical Computation and Simulation*, 10(3-4):205–223, 1980. doi:10.1080/00949658008810370.

15 Dan Hoey and Michael Ian Shamos. Closest-point problems. In *Proceedings of the 16th Annual Symposium on Foundations of Computer Science*, SFCS '75, pages 151–162, USA, 1975. IEEE Computer Society. doi:10.1109/SFCS.1975.8.

16 Bomin Jiang, Zhiyong Sun, Brian D.O. Anderson, and Christian Lageman. Higher order mobile coverage control with applications to clustering of discrete sets. *Automatica*, 102:27–33, 2019. doi:10.1016/j.automatica.2018.12.028.

17 Josef Kallrath, Joonghyun Ryu, Chanyoung Song, Mokwon Lee, and Deok-Soo Kim. Near optimal minimal convex hulls of disks. *Journal of Global Optimization*, 80(3):551–594, 2021. doi:10.1007/s10898-021-01002-5.

18 John W. Krussel and Barry F. Schaudt. A note on higher order Voronoi diagrams. *Nordic Journal of Computing*, 1(2):268–272, 1994. URL: https://www.cs.helsinki.fi/njc/njc1_papers/number2/note2.pdf.

19 Der-Tsai Lee. On k-nearest neighbor Voronoi diagrams in the plane. *IEEE Trans. Comput.*, 31(6):478–487, 1982. doi:10.1109/TC.1982.1676031.

20 Roderik Lindenbergh. A Voronoi poset. *Journal for Geometry and Graphics*, 7:41–52, 2003.

21 Chih-Hung Liu, Evanthia Papadopoulu, and Der-Tsai Lee. The k-nearest-neighbor Voronoi diagram revisited. *Algorithmica*, 71(2):429–449, 2015. doi:10.1007/s00453-013-9809-9.

22 L. Lovász, K. Vesztergombi, U. Wagner, and E. Welzl. Convex quadrilaterals and k-sets. In J. Pach, editor, *Towards a Theory of Geometric Graphs*, volume 342, pages 139–148. Contemporary Mathematics, 2004. doi:10.1090/conm/342.

23 Juan Enrique Martínez-Legaz, Vera Roshchina, and Maxim Todorov. On the structure of higher order Voronoi cells. *Journal of Optimization Theory and Applications*, 183(1):24–49, 2019. doi:10.1007/s10957-019-01555-2.

24 Robert Edwards Miles. On the homogeneous planar Poisson point process. *Mathematical Biosciences*, 6:85–127, 1970. doi:10.1016/0025-5564(70)90061-1.

25 Robert Edwards Miles and Robert James Maillardet. The basic structures of Voronoi and generalized Voronoi polygons. *Journal of Applied Probability*, 19(A):97–111, 1982. doi:10.2307/3213553.

26 Atsuyuki Okabe, Barry Boots, Kokichi Sugihara, and Sung Nok Chiu. *Spatial Tessellations: Concepts and Applications of Voronoi Diagrams*. John Wiley & Sons, Chichester, UK, 2 edition, 2000.

27 Patrick O'Neil and Thomas Wanner. Analyzing the squared distance-to-measure gradient flow system with k-order Voronoi diagrams. *Discrete & Computational Geometry*, 61(1):91–119, 2019. doi:10.1007/s00454-018-0037-6.

Disproving two conjectures on the Hamiltonicity of Venn diagrams*

Sofia Brenner[1], Linda Kleist[2], Torsten Mütze[1], Christian Rieck[1], and Francesco Verciani[1]

1 **Institute of Mathematics, University of Kassel**
{sbrenner, tmuetze, christian.rieck, verciani}@mathematik.uni-kassel.de

2 **Department of Computer Science, University of Hamburg**
linda.kleist@uni-hamburg.de

Abstract

In 1984, Winkler conjectured that every simple Venn diagram with n curves can be extended to a simple Venn diagram with $n+1$ curves. His conjecture is equivalent to the statement that the dual graph of any simple Venn diagram has a Hamilton cycle. In this work, we construct counterexamples to Winkler's conjecture for all $n \geq 6$. As part of this proof, we computed all 3.430.404 simple Venn diagrams with $n = 6$ curves (even their number was not previously known), among which we found 72 counterexamples. We also construct monotone Venn diagrams, i.e., diagrams that can be drawn with n convex curves, and are not extendable, for all $n \geq 7$. Furthermore, we also disprove another conjecture about the Hamiltonicity of the (primal) graph of a Venn diagram. Specifically, while working on Winkler's conjecture, Pruesse and Ruskey proved that this graph has a Hamilton cycle for every simple Venn diagram with n curves, and conjectured that this also holds for non-simple diagrams. We construct counterexamples to this conjecture for all $n \geq 4$.

Related Version arXiv:2503.18554

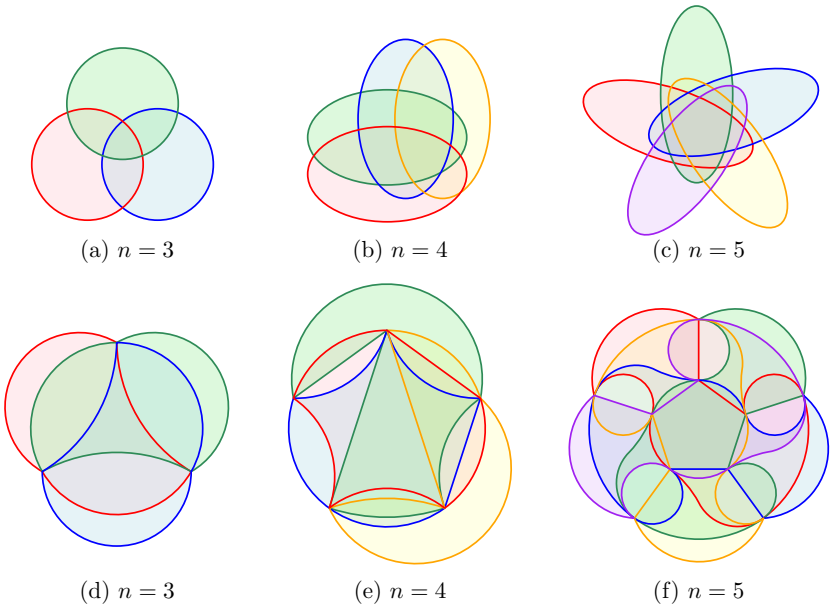
1 Introduction

Venn diagrams are a popular tool to illustrate the relations between sets and operations on them, such as union and intersection, or the well-known inclusion-exclusion principle. Historically, these diagrams were introduced by John Venn [18] (1834–1923) in the context of formal logic. Most illustrations depict the Venn diagram with three sets represented by unit circles, as in Figure 1(a). It may even come as a surprise that Venn diagrams exist for any number of sets, not just three. Formally, an *n-Venn diagram* is a collection of n simple closed curves in the plane that intersect in only finitely many points and create exactly 2^n regions, one for every possible combination of being inside or outside of each of the n curves. Figures 1(b) and (c) display diagrams with $n = 4$ and $n = 5$ curves, respectively.

A Venn diagram is *simple* if at most two of the n curves intersect in any point: The three diagrams in the top row of Figure 1 are simple, whereas the three diagrams in the bottom row are non-simple. An easy application of Euler's formula shows that a simple Venn diagram has exactly $2^n - 2$ crossings, whereas non-simple diagrams may have much fewer. For humans, simple Venn diagrams tend to be easier to read than non-simple diagrams.

* Sofia Brenner, Torsten Mütze, Christian Rieck, and Francesco Verciani were supported by German Science Foundation grant 522790373.

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.



(a) $n = 3$ (b) $n = 4$ (c) $n = 5$

(d) $n = 3$ (e) $n = 4$ (f) $n = 5$

Figure 1 Venn diagrams with $n = 3, 4, 5$ curves: (a), (b), and (c) are simple, whereas (d), (e), and (f) are non-simple; (a), (b), and (d) are reducible, whereas (c), (e), and (f) are irreducible; (a), (b), (c), and (d) are monotone, whereas (e) and (f) are non-monotone; (a), (b), (c), (d), and (f) are exposed, whereas (e) is not.

An interesting subclass of Venn diagrams are those that can be drawn with n convex curves. This property can be captured purely combinatorially, as follows: A *k-region* in an n-Venn diagram is a region that lies inside of exactly k of the n curves (and outside of the remaining $n - k$ curves). A *monotone* Venn diagram is one in which for every $0 < k < n$, every k-region is adjacent to both a $(k-1)$-region and a $(k+1)$-region. Bultena, Grünbaum and Ruskey [3] proved that monotone diagrams are precisely the ones that can be drawn such that every curve has a convex shape. Formally, every diagram made of convex curves is monotone, and every monotone diagram can be convexified. Monotone diagrams can be realized as a *wire diagram*, where the n curves are drawn as horizontal lines at heights $1, 2, \ldots, n$, and each crossing between neighboring curves occurs within a narrow vertical strip. Furthermore, the diagram implicitly wraps around cyclically at the left and right boundary; see Figure 2. Note that a convex realization can be obtained from the wire diagram by placing it in a very thin annulus with large enough radius.

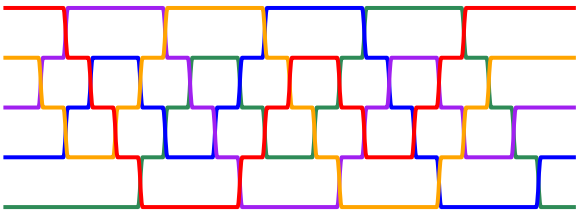


Figure 2 Representation of the 5-Venn diagram from Figure 1(c) as a wire diagram.

2 Winkler's conjecture

In 1984, Peter Winkler [20] raised the following intriguing conjecture about extending a simple Venn diagram by adding one curve to it.

▶ **Conjecture 2.1.** *Every simple n-Venn diagram can be extended to a simple $(n+1)$-Venn diagram by the addition of a suitable curve.*

He also mentioned the conjecture as an unsolved open problem in his popular book [21] on mathematical puzzles, and reiterated it in his 'Puzzled' column [22] in the Communications of the ACM. The conjecture is also listed as the first open problem in Ruskey and Weston's survey [15] on Venn diagrams. Concerning small cases, the conjecture is easily verifiable for all simple Venn diagrams on $n \in \{3, 4, 5\}$ curves, simply because there are so few of them. Another class of Venn diagrams for which the conjecture is easily seen to be true are reducible diagrams. An n-Venn diagram is *reducible* if one of its curves can be removed so that the remaining $n-1$ curves form an $(n-1)$-Venn diagram. Otherwise the diagram is called *irreducible*. The diagram in Figure 1(c) is irreducible, because any 4 of the 5 curves create 20 regions instead of 16. If a diagram is reducible, then the curve C that can be removed splits every region of the resulting $(n-1)$-Venn diagram into two, and therefore this curve touches every region in the n-Venn diagram. We can thus add an $(n+1)$st curve by following C and subdividing every region into two, changing sides along C exactly once along every segment between two consecutive crossings. Thus, the interesting instances of Conjecture 2.1 are irreducible diagrams.

Grünbaum [9] modified Winkler's conjecture, dropping the requirement for the diagrams to be simple. This makes the problem harder in the sense that more diagrams are allowed, but easier in the sense that there is substantially more freedom when adding the new curve. This variant of the conjecture was proved subsequently by Chilakamarri, Hamburger and Pippert [4], by using a straightforward reduction to a classical result of Whitney [19].

2.1 Translation to a Hamilton cycle problem

Problems on Venn diagrams such as Winkler's conjecture are best discussed by considering the *dual graph* of the Venn diagram, i.e., the plane graph that has a vertex inside every region, and edges connecting vertices corresponding to neighboring regions; see Figure 3.

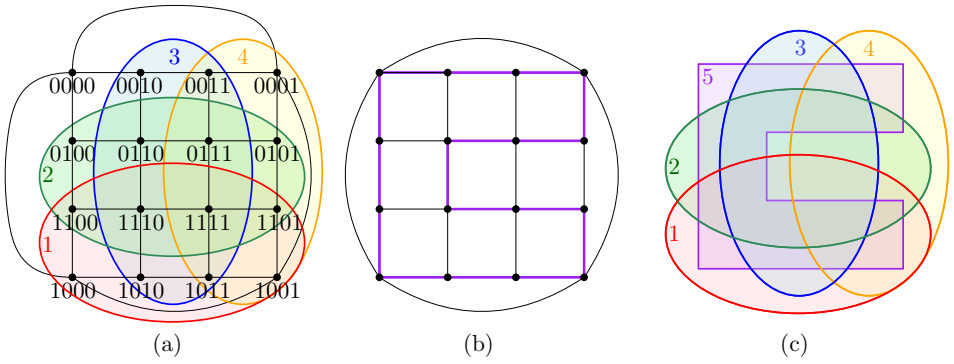


Figure 3 (a) A (monotone) 4-Venn diagram and its dual graph, a (monotone) 4-Venn quadrangulation; (b) a Hamilton cycle in the quadrangulation; (c) the 5-Venn diagram obtained from (a) by adding the curve corresponding to the Hamilton cycle in (b).

For this, we consider the *n-dimensional hypercube* Q_n, the graph formed by all bitstrings of length n, with an edge between any two bitstrings that differ in a single bit. The dual graph $Q(D)$ of a simple n-Venn diagram D satisfies the following properties:

① It is a connected subgraph of Q_n that is also spanning, i.e., all 2^n vertices are present. Specifically, the vertex in $Q(D)$ corresponding to a region of the diagram D is the characteristic vector of the inside/outside relation with respect to each curve, where 0 and 1 represent outside and inside, respectively.[1]
② It is a plane quadrangulation, i.e., it is drawn in the plane without edge crossings, and every face (including the outer face) is a 4-cycle.
③ For every position $i \in \{1, \ldots, n\}$ and every bit $b \in \{0, 1\}$, the subgraph of $Q(D)$ induced by all vertices x with $x_i = b$ is connected.

Conversely, the dual of any subgraph of Q_n satisfying these properties is an n-Venn diagram. We refer to subgraphs of Q_n satisfying ①–③, which are the duals of n-Venn diagrams, as *n-Venn quadrangulations*. We refer to these as *monotone*, if they satisfy the following additional condition, where the *weight* $w(x)$ of a bitstring x is the number of 1s in x:

③' Every vertex x with weight $0 < w(x) < n$ has a neighbor of weight $w(x) - 1$ and a neighbor of weight $w(x) + 1$.

Monotone n-Venn quadrangulations are precisely the duals of monotone n-Venn diagrams. Note that ③' implies ③, as connectedness is ensured via the vertices 0^n and 1^n. Monotone n-Venn quadrangulations are special cases of *rhombic strips*, recently studied in [1, 2].

Extending a given diagram D by a single curve means that the new curve has to enter and leave every region of D exactly once, splitting it into two, one of them inside and the other one outside the newly added curve. This corresponds to traversing a Hamilton cycle in the dual graph $Q(D)$, i.e., a cycle in the graph that visits every vertex exactly once; see Figures 3(b) and (c). Conjecture 2.1 can thus be restated equivalently as follows: For $n \geq 2$, every n-Venn quadrangulation admits a Hamilton cycle.

2.2 The counterexamples

As Q_n is a bipartite graph, Venn quadrangulations are bipartite, too. A necessary condition for the existence of a Hamilton cycle in a bipartite graph is the existence of a perfect matching. We provide counterexamples to Conjecture 2.1 by constructing, for every $n \geq 6$, an n-Venn quadrangulation that has no perfect matching and thus no Hamilton cycle. One of our counterexamples for $n = 6$ is shown in Figure 4.

▶ **Theorem 2.2.** *For every $n \geq 6$ there is an n-Venn quadrangulation that has no perfect matching and hence no Hamilton cycle. Its dual is a simple n-Venn diagram that cannot be extended to a simple $(n+1)$-Venn diagram by adding a suitable curve.*

On a high level, to inductively construct counterexamples for larger n, we carefully glue together two sliced copies of the diagram with $n - 1$ curves, surrounding one of the copies with an additional curve. For slitting the copies, we identify a subregion along a curve that intersects all other curves. In the respective dual graph, this corresponds to a ladder-like subgraph (highlighted in green in Figure 4). Since we need to ensure that we preserve the

[1] When considering the diagram in the plane, then the inside and outside of a curve C are the bounded and unbounded region of $\mathbb{R}^2 \setminus C$, respectively.

local obstruction (i.e., the subgraph witnessing the non-existence of a perfect matching and thus the non-extendability, highlighted in blue and red in Figure 4), the ladder and this part of the diagram must not intersect.

One may wonder whether Winkler's conjecture can be salvaged for monotone diagrams, i.e., those that can be drawn with convex curves. Our next theorem shows that this is not possible; a counterexample for $n = 7$ is shown in Figure 5.

▶ **Theorem 2.3.** *For every $n \geq 7$ there is a monotone n-Venn quadrangulation that has no perfect matching and hence no Hamilton cycle. Its dual is a simple monotone n-Venn diagram that cannot be extended to a simple $(n+1)$-Venn diagram by adding a suitable curve.*

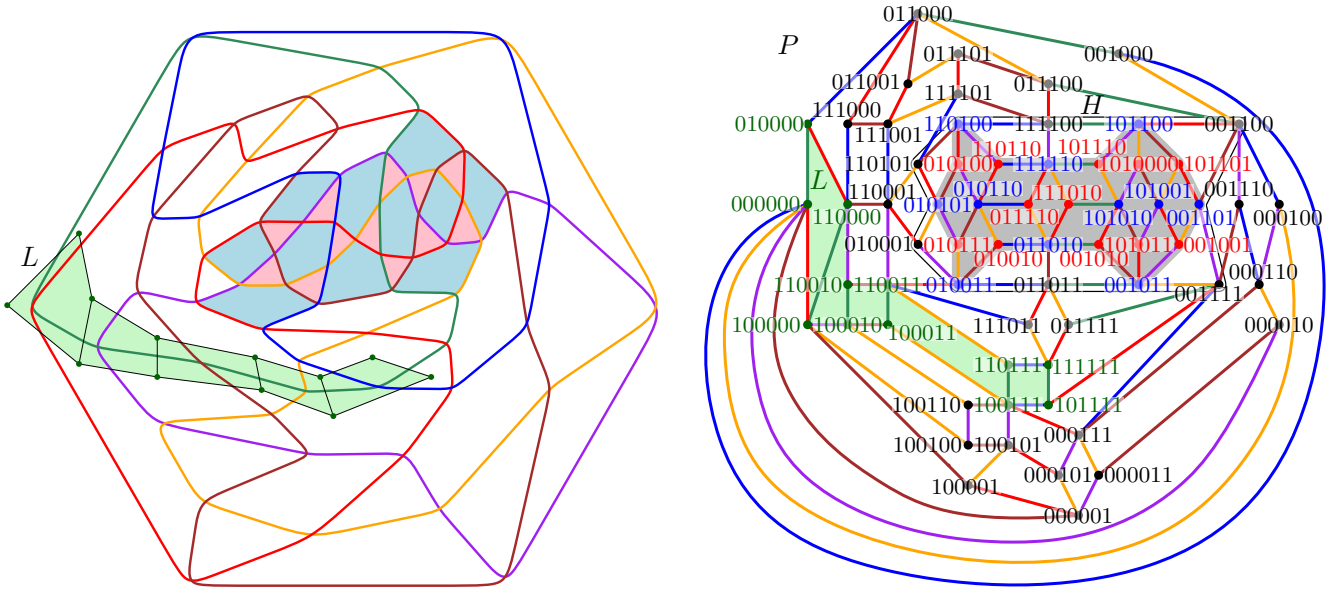


Figure 4 (Left) A counterexample to Winkler's conjecture for $n = 6$, namely a simple (non-monotone) 6-Venn diagram that cannot be extended to a simple 7-Venn diagram. Highlighted are 12 red regions that are surrounded by only 11 blue regions, which is a local obstruction for extendability.[2] (Right) The dual graph of the diagram on the left, namely a 6-Venn quadrangulation P that has no perfect matching and hence no Hamilton cycle. The highlighted subgraph H contains a subset of 12 red vertices in one partition class and their 11 blue neighbors, witnessing the non-existence of a perfect matching. The green subgraph L is a ladder that is used for extending the counterexample to all values of $n > 6$, and all those extensions contain a copy of H.

As mentioned before, for $n \leq 5$ all simple n-Venn diagrams are extendable, so Theorem 2.2 is best possible in the sense that the counterexamples for $n = 6$ are minimal. In fact, as part of our proof of Theorem 2.2 we computed all distinct 6-Venn diagrams and analyzed their properties. As it turns out, all 72 non-extendable diagrams for $n = 6$ are non-monotone, and so Theorem 2.3 is also best possible.

[2] This drawing and the one in Figure 5 were produced using a `SageMath` program [16] developed by Scheucher for his and Felsner's work [7] on pseudocircle arrangements, which we used with permission.

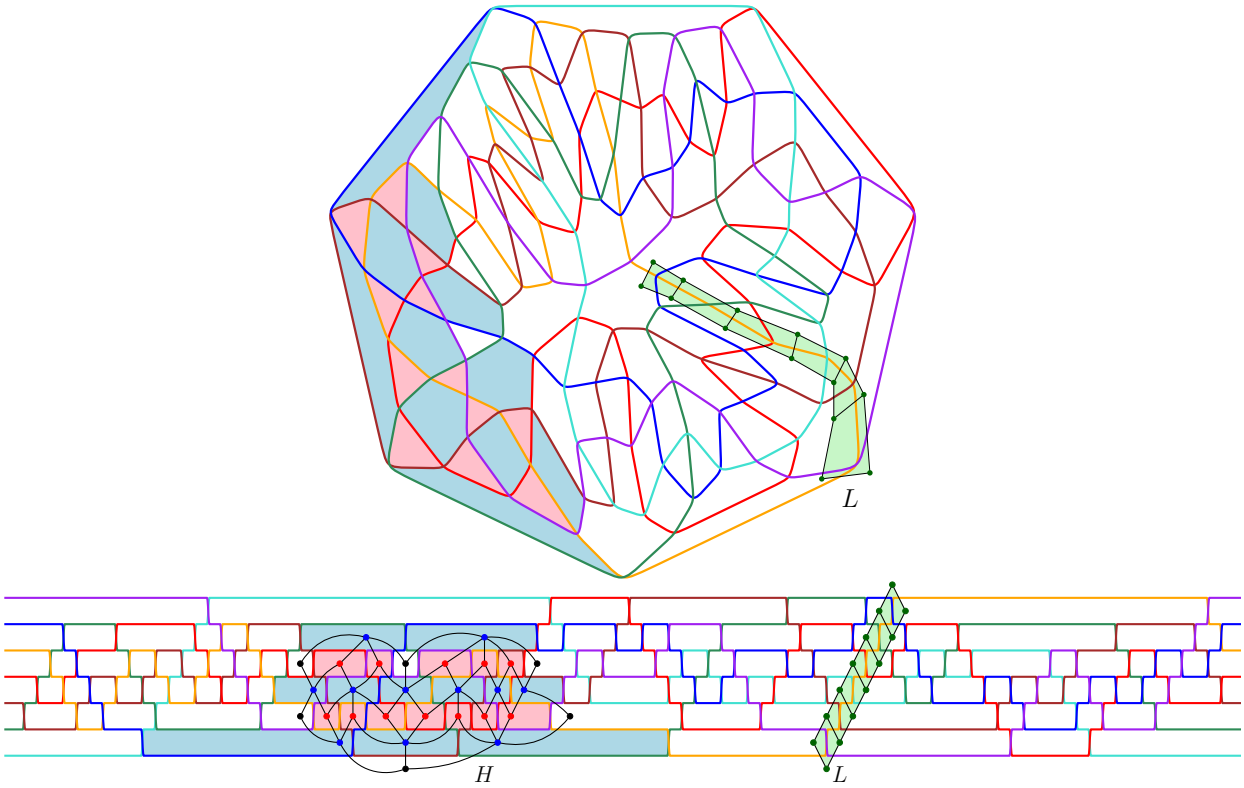


Figure 5 A simple monotone 7-Venn diagram that cannot be extended to a simple 8-Venn diagram. The bottom picture is the same diagram in a wire representation. The subgraph H of the dual Venn quadrangulation is the same as in Figure 4. It corresponds to the highlighted 12 red regions that are surrounded by only 11 blue regions, which is a local obstruction for extendability.

For our counting results (see Theorem 2.4 and Figure 6), we consider two diagrams the same if they differ only by mirroring and/or stereographic projection. Equivalently, we count their dual n-Venn quadrangulations up to graph isomorphism. Prior to this work, the number of simple diagrams was only known up to $n \leq 5$. Specifically, for each $n \leq 4$ there is only one, and for $n = 5$ there are 20 [6, 10]. An *exposed* diagram is one in which the outer region is bounded by all curves. It is not hard to see that any monotone diagram is exposed. Chilakamarri, Hamburger, and Pippert [5] constructed n-Venn diagrams for every n in which all regions are (combinatorial) triangles, quadrangles, or pentagons; in particular, they are not exposed for $n \geq 6$.

▶ **Theorem 2.4.** *There are* 3.430.404 *non-isomorphic 6-Venn quadrangulations (i.e., simple 6-Venn diagrams up to mirroring and/or stereographic projection). Of these,* 72 *do not have a Hamilton cycle (i.e., the diagram cannot be extended to a simple 7-Venn diagram), 60 do not have a Hamilton path, and 17 do not have a perfect matching. Furthermore, of the corresponding Venn diagrams, 157.619 are reducible, 32.255 are monotone and 267.510 are not exposed, and all those are extendable to a 7-Venn diagram.*

The programs we used for computing all simple 6-Venn diagrams and the result files with all 6-Venn quadrangulations, as well as the monotone 7-Venn quadrangulation corresponding to the diagram in Figure 5, are available for download [13].

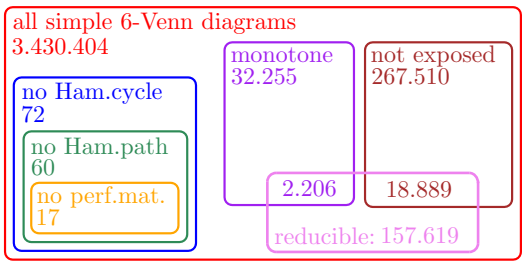


Figure 6 Census of 6-Venn diagrams and their properties stated in Theorem 2.4, visualized as an Euler diagram. The sizes of the boxes are not to scale.

3 Pruesse and Ruskey's conjecture

While working on Winkler's conjecture, Pruesse and Ruskey [14] considered the question of finding a Hamilton cycle not in the dual graph of a Venn diagram, but in the (primal) graph whose vertices are the crossings of the curves, and curve segments between consecutive crossings form the edges; see Figure 7. Note that these graphs may have multiple edges.

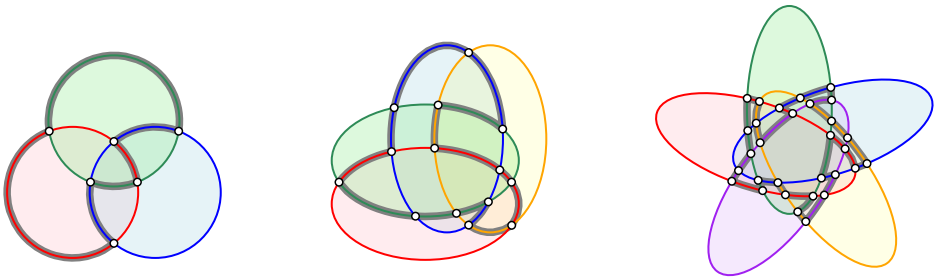


Figure 7 Hamilton cycles in the graphs of the Venn diagrams from Figure 1 (a)–(c).

▶ **Theorem 3.1** ([14, Cor. 1]). *For $n \geq 2$, the graph of any simple n-Venn diagram has a Hamilton cycle.*

Their proof demonstrates that the graph associated with a simple Venn diagram is 4-connected for $n \geq 3$, and thus a well-known theorem of Tutte [17] implies Hamiltonicity. Pruesse and Ruskey also conjectured that their result generalizes to non-simple diagrams.

▶ **Conjecture 3.2** ([14, Conj. 1]). *For $n \geq 2$, the graph of any (not necessarily simple) n-Venn diagram has a Hamilton cycle.*

3.1 The counterexamples

We disprove Conjecture 3.2 by providing counterexamples for all $n \geq 4$. Similarly to Theorem 2.2, our counterexamples have an even number of vertices and do not admit a perfect matching. One of the counterexamples for $n = 4$ is shown in Figure 8.

▶ **Theorem 3.3.** *For every $n \geq 4$, there is a non-simple n-Venn diagram whose graph has an even number of vertices and no perfect matching, and hence no Hamilton cycle.*

The counterexamples of Theorem 3.3, including the one in Figure 8, are monotone.

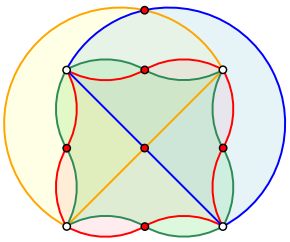


Figure 8 A counterexample to Conjecture 3.2 for $n = 4$. The 6 red vertices form an independent set, and the complement has only 4 vertices, witnessing the non-existence of a perfect matching.

4 Open questions

Computationally, it could be interesting to find all non-simple Venn diagrams for $n = 4$ or $n = 5$ curves and analyze their properties.

What can be said about the asymptotic number of n-Venn diagrams as n grows, or about the proportion of diagrams with certain properties (recall Figure 6)?

There is a large number of other intriguing problems on Venn diagrams, concerning symmetry, convexity, number of crossings, restrictions on the allowed shapes of the curves, to be found in the survey [15]. For example, is there a simple n-Venn diagram with rotational symmetry made of n congruent curves for every prime number n (see [8, 11, 12])?

References

1 Hugo A. Akitaya, Jean Cardinal, Stefan Felsner, Linda Kleist, and Robert Lauff. Facet-Hamiltonicity. In *Symposium on Discrete Algorithms (SODA)*, pages 5051–5064, 2025. doi:10.1137/1.9781611978322.171.

2 Nastaran Behrooznia, Sofia Brenner, Arturo Merino, Torsten Mütze, Christian Rieck, and Francesco Verciani. Listing faces of polytopes. In *Symposium on Discrete Algorithms (SODA)*, pages 6212–6222, 2026. doi:10.1137/1.9781611978971.222.

3 Bette Bultena, Branko Grünbaum, and Frank Ruskey. Convex drawings of intersecting families of simple closed curves. In *Canadian Conference on Computational Geometry (CCCG)*, 1999. URL: http://www.cccg.ca/proceedings/1999/c14.pdf.

4 Kiran B. Chilakamarri, Peter Hamburger, and Raymond E. Pippert. Hamilton cycles in planar graphs and venn diagrams. *J. Comb. Theory B*, 67(2):296–303, 1996. doi:10.1006/JCTB.1996.0047.

5 Kiran B. Chilakamarri, Peter Hamburger, and Raymond E. Pippert. Venn diagrams and planar graphs. *Geom. Dedicata*, 62(1):73–91, 1996. doi:10.1007/BF00240003.

6 Kiran B. Chilakamarri, Peter Hamburger, and Raymond E. Pippert. Analysis of Venn diagrams using cycles in graphs. *Geom. Dedicata*, 82(1-3):193–223, 2000. doi:10.1023/A:1005288325189.

7 Stefan Felsner and Manfred Scheucher. Arrangements of pseudocircles: Triangles and drawings. *Discret. Comput. Geom.*, 65(1):261–278, 2021. doi:10.1007/S00454-020-00173-4.

8 Jerrold R. Griggs, Charles E. Killian, and Carla D. Savage. Venn diagrams and symmetric chain decompositions in the boolean lattice. *Electron. J. Comb.*, 11(1), 2004. doi:10.37236/1755.

9 Branko Grünbaum. Venn diagrams. I. *Geombinatorics*, 1(4):5–12, 1992.

10 Peter Hamburger and Raymond E. Pippert. Simple, reducible Venn diagrams on five curves and Hamiltonian cycles. *Geom. Dedicata*, 68(3):245–262, 1997. doi:10.1023/A:1004924423245.

11 Charles E. Killian, Frank Ruskey, Carla D. Savage, and Mark Weston. Half-simple symmetric Venn diagrams. *Electron. J. Comb.*, 11(1), 2004. doi:10.37236/1839.

12 Khalegh Mamakani and Frank Ruskey. New roses: Simple symmetric Venn diagrams with 11 and 13 curves. *Discret. Comput. Geom.*, 52(1):71–87, 2014. doi:10.1007/S00454-014-9605-6.

13 Torsten Mütze. Simple 6-Venn diagrams: C++ program and data files. http://tmuetze.de/papers/venn.zip.

14 Gara Pruesse and Frank Ruskey. All simple Venn diagrams are Hamiltonian, 2015. doi:10.48550/arXiv.1504.06651.

15 Frank Ruskey and Mark Weston. A survey of Venn diagrams. *Electron. J. Combin.*, 4(1):Dynamic Survey 5, 1997. doi:10.37236/26.

16 Manfred Scheucher. Pseudocircle arrangements tools, 2024. GitHub repository: https://github.com/manfredscheucher/pseudocircle-arrangements-tools/.

17 William T. Tutte. A theorem on planar graphs. *Trans. Amer. Math. Soc.*, 82:99–116, 1956. doi:10.2307/1992980.

18 John Venn. On the diagrammatic and mechanical representation of propositions and reasonings. *Phil. Mag. S. 5.*, 9(59):1–18, 1880. doi:10.1080/14786448008626877.

19 Hassler Whitney. A theorem on graphs. *Ann. of Math. (2)*, 32(2):378–390, 1931. doi:10.2307/1968197.

20 Peter Winkler. Venn diagrams: Some observations and an open problem. *Congr. Numer.*, 45:267–274, 1984.

21 Peter Winkler. *Mathematical puzzles: A connoisseur's collection.* AK Peters Series, Taylor & Francis, 2004.

22 Peter Winkler. Puzzled: Where sets meet (Venn diagrams). *Commun. ACM*, 55(2):128, 2012. doi:10.1145/2076450.2076475.

Bouquet: A Visualization Tool for Symmetry Sets and Vineyards*

Erin Chambers[1], Christopher Fillmore[2], Shankha Shubhra Mukherjee[1], Rohit Roy[3], Elizabeth Stephenson[4], and Mathijs Wintraecken[3]

1 **University of Notre Dame**
echambe2@nd.edu, smukher4@nd.edu
2 **ISTA (Institute of Science and Technology Austria)**
cdfillmore@gmail.com
3 **Inria Centre Université Côte d'Azur**
rohit.a.roy@inria.fr, mathijs.wintraecken@inria.fr
4 **Orteliu**
elizasteprene@gmail.com

Abstract

The Persistent Homology Transform (PHT) has emerged as a powerful topological summary for shape analysis, capturing multiscale geometric information by scanning a shape along all possible directions. In this paper we study a close variant, namely the Radial Persistence Transform (RPT). The RPT maps a point x in the ambient space to the persistence diagram of the squared Euclidean distance function $d^2_{\mathbb{E}}(\cdot, x)|_{\mathcal{M}}$ restricted to a manifold $\mathcal{M}$. Previous results by Bruce, Giblin, and Gibson, and by Porteous, establish that the topology of this distance function, and hence the persistence diagram, undergoes changes if x traverses the symmetry set and focal set.

We present an interactive web-based 2D tool, Bouquet2D, designed to explore these transitions through the lens of vineyards (stacked persistence diagrams). By parameterizing x along a user-defined loop γ, our tool tracks the continuous evolution of persistence diagrams, revealing the complex "braiding" patterns and monodromy in the vineyard recently characterized in [4].

The tool features a real-time spline-based interface, allowing researchers to deform $\mathcal{M}$ and observe how geometric bifurcations manifest as topological events in the vineyard. By exhibiting the relation between the symmetry and focal set and the topology of the vineyard, this software provides a visual framework for investigating the origin of topology in vineyards.

1 Introduction

Vineyards, which are time-varying persistence diagrams arising from one-parameter families of filtrations, have found applications across topological data analysis [2, 7, 14]. By stability, points in persistence diagrams vary continuously with the parameter, tracing out curves called vines [5, 13]. When the parameter space is periodic, we obtain a closed vineyard.

Recent work has revealed much topological complexity in periodic vineyards. [1] exhibited closed vineyards with monodromy, while [4] showed that any order of monodromy occurs in some closed vineyard. Moreover, [4] showed that, given a knot, there exists a manifold and a family of distance functions restricted to the manifold such that the corresponding closed vineyard contains the given knot. The construction of [4] is performed in the context of the Radial Distance Transform (RDT), i.e. given a fixed manifold $\mathcal{M} \subset \mathbb{R}^d$ and a point

* This work has been supported by the ANR grant StratMesh, ANR-24-CE48-1899, by NSF award 2444309, and the welcome package from IDEX of the Université Côte d'Azur, ANR-15-IDEX-01.

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

$x \in \mathbb{R}^d$ (which will be varied later along a curve γ to generate a vineyard), one considers the persistence diagram of the (squared) Euclidean distance function restricted to $\mathcal{M}$, that is, $d^2_{\mathbb{E}}(\cdot, x)|_{\mathcal{M}}$.

On the other hand it is known [3, 11] that the bifurcation set of the function $d^2_{\mathbb{E}}(\cdot, x)|_{\mathcal{M}}$ is the union of the focal and symmetry sets (which will be defined below). The bifurcation set is the set of points where topological changes occur. This leads to the natural question:

How is the topology of the focal and symmetry sets (the bifurcation set) related to the topology of the vineyard?

In this paper, we present a 2D visualization tool hosted at `Bouquet2D`[1] to study this relationship for explicit examples. More precisely, our tool enables interactive study of how topological features—such as vine crossings, monodromy, and braiding (refer [4] for the definitions)—emerge and evolve, providing geometric intuition that guides theoretical analysis.

2 Preliminaries

▶ **Definition 2.1** (Restricted distance function)**.** For a $p \in \mathbb{R}^d$ we can consider the (Euclidean) distance function from this point $d_{\mathbb{E}}(\cdot, p) : \mathbb{R}^d \to \mathbb{R} : x \mapsto d_{\mathbb{E}}(x, p)$. Letting $\mathcal{M} \subset \mathbb{R}^d$ be a smooth manifold, we call the distance function restricted to $\mathcal{M}$, that is $d_{\mathbb{E}}(\cdot, p)|_{\mathcal{M}} : \mathcal{M} \to \mathbb{R} : x \mapsto d_{\mathbb{E}}(x, p)$, the restricted distance function.

2.1 Focal and Symmetry Set

To define the focal set in $\mathbb{R}^d$ is beyond the scope of this paper, so we will stick to the 2D case for which the tool is constructed. In 2D, the focal set is also called the *evolute* of the curve. Following the definition given by Porteous [12], the evolute is defined as the locus of the centers of curvature of the curve.

Let $\mathcal{M}$ be a smooth plane curve parameterized by arc length s, given by $\gamma(s)$. Let $\mathbf{t}(s) = \gamma'(s)$ be the unit tangent vector and $\mathbf{n}(s)$ be the unit normal vector obtained by rotating $\mathbf{t}(s)$ by 90° counter-clockwise.

▶ **Definition 2.2** (Curvature and Radius of Curvature)**.** The **curvature** $\kappa(s)$ is the rate of change of the tangent vector in the direction of the normal:

$$\mathbf{t}'(s) = \kappa(s)\mathbf{n}(s)$$

The **radius of curvature** $\rho(s)$ is the reciprocal of the curvature:

$$\rho(s) = \frac{1}{\kappa(s)}, \quad \text{for } \kappa(s) \neq 0$$

▶ **Definition 2.3** (Evolute)**.** The **evolute** (or focal set) of $\mathcal{M}$ is the curve $E(s)$ defined by the centers of osculating circles of the plane curve $\mathcal{M}$:

$$E(s) = \gamma(s) + \rho(s)\mathbf{n}(s)$$

where $\mathbf{n}(s)$ is the unit normal vector to the curve at $\gamma(s)$.

[1] https://www.rohitroy.me/symmetry-set-visualizer/bouquet2d.html

▶ **Definition 2.4** (Symmetry set, [3]). Let $\mathcal{M} \subset \mathbb{R}^d$ be a smooth manifold. The symmetry set $\mathrm{sym}(\mathcal{M})$ of $\mathcal{M}$ is the closure of the set of points p in $\mathbb{R}^d$, such that there is a radius r so that $B(p,r)$ is tangent to $\mathcal{M}$ in at least two places.

The local structure of the symmetry and focal set is determined by the classification of singularities [3].

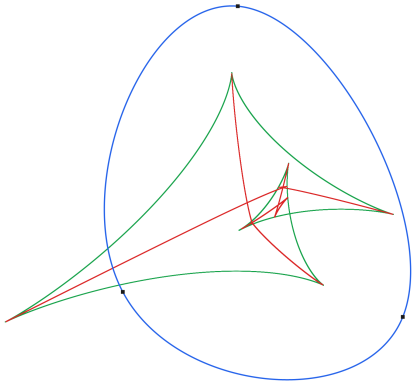


Figure 1 Visualizing the focal set (evolute) and symmetry set of a spline-defined curve $\mathcal{M}$. The green locus represents the evolute. The red locus represents the symmetry set.

2.2 Vineyards

A central component of our visualization is the concept of a *vineyard*, which captures the continuous evolution of persistence diagrams as the filtration function changes. Introduced in [5], vineyards provide a stable method for tracking topological features across a parameterized family of functions. Let $\mathcal{M} \subset \mathbb{R}^2$ be a smooth manifold (curve) and let $\gamma : S^1 \to \mathbb{R}^2$ be a closed curve. For each $t \in S^1$, define the radial distance function $f_t : \mathcal{M} \to \mathbb{R}$ by

$$f_t(x) = d^2_{\mathbb{E}}(x, \gamma(t))|_{\mathcal{M}}.$$

▶ **Definition 2.5** (Vineyard). The **vineyard** of the radial persistence transform, denoted $\mathcal{V}(\mathcal{M}, \gamma)$, is the subset of $S^1 \times \overline{\mathbb{R}}^2$ given by:

$$\mathcal{V}(\mathcal{M}, \gamma) = \bigcup_{t \in S^1} \{t\} \times \mathrm{Dgm}(f_t)$$

where $\mathrm{Dgm}(f_t)$ is the persistence diagram of the sublevel set filtration of f_t.

The set $\mathcal{V}(\mathcal{M}, \gamma)$ decomposes into curves called *vines*, which track individual persistence points as t varies. In our tool, as the center point $\gamma(t)$ traverses the closed curve, vines trace the evolution of birth and death times of homological features.

2.3 Monodromy

Following a vine in a vineyard for an entire period, that is, traversing S^1 once, does not necessarily lead to the point in the persistence diagram you set out with. If this is the case,

one says that there is monodromy. If $k > 1$ is the smallest number such that following a (or each) vine while traversing the loop k times you end up at the starting point in the persistence diagram, you say the monodromy is of order k.

3 Algorithm

3.1 Calculation of the Symmetry Set

The Symmetry Set (SS) is the locus of centers of circles bitangent to a curve at two or more points. We are following the algorithm described in [3] for the computation of the symmetry set. Consider sampled points $\mathcal{P} = \{p_1, p_2, \ldots, p_k\}$ from $\mathcal{M}$ and corresponding tangents $\{t_1, t_2, \ldots, t_k\}$ and normals $\{n_1, n_2, \ldots, n_k\}$ at these points. Algorithm 1 describes our implementation.

Bitangency Condition We identify potential bitangencies by detecting zero-crossings of the functions:

$$f_{\pm}(i, j) = (p_i - p_j) \cdot (t_i \pm t_j)$$

A sign change in $f_{\pm}$ between adjacent samples p_j, p_{j+1} indicates a local bitangent condition.

Intersection Calculation When a zero-crossing is detected, the center C is found by intersecting the normal lines $L_i = p_i + \lambda n_i$ and $L_j = p_j + \mu n_j$. We solve for λ by projecting the $(p_i - p_j)$ onto the combined normal vector:

$$\lambda = \frac{-(p_i - p_j) \cdot (n_i \pm n_j)}{\|n_i \pm n_j\|^2}$$

The candidate center is $C = p_i + \lambda n_i$. We retain C only if it satisfies the equidistance constraint $|\|C - p_i\| - \|C - p_j\|| < \epsilon_{rad}$, which ensures numerical stability against discretization artifacts.

3.2 Vineyard Computation

The vineyard $\mathcal{V}$ is constructed by stack-wise concatenation of persistence diagrams generated from a parameterized family of distance functions. Methods have been proposed by e.g. [8] to compute vineyards effectively, but an elementary approach suffices here.[2] In our tool, we parameterize the filtration by moving the source point $\gamma(t)$ along a circular path of user-defined radius and center, or an arbitrary user-drawn loop via splines [6].

Persistence For a fixed t, we compute persistence using a Union-Find based algorithm on the lower star filtration of the squared distance function $f_\theta(p) = d^2_{\mathbb{E}}(p, \gamma(t))|_{\mathcal{M}}$. As we iterate through points $p \in \mathcal{P}$ in increasing order of distance, components are merged. The "Elder Rule" is applied to pair the birth time (distance) of a merging component with the current distance as its death time.

Vine Tracking The tool captures the evolution of these birth-death pairs across m discrete samples of $t \in S^1$. The resulting collection of points in $(t, \text{birth}, \text{death})$ space forms the vines. By visualizing these vines, the user can observe the *braiding* of topological features, in accordance with [4].

[2] Because working with curves in 2D means the efficient union find algorithm suffices to compute the persistence diagram, which is sufficiently fast, not to require more sophisticated tools to further speed-up.

Algorithm 1: Discrete Symmetry Set Computation

Input: Sampled curve points $\mathcal{P} = \{p_1, \ldots, p_k\}$ with tangents $\{t_1, \ldots, t_k\}$ and normals $\{n_1, \ldots, n_k\}$. Thresholds $\epsilon_{chord}, \epsilon_{rad}$.

Output: List of Symmetry Set centers $\mathcal{S}$

$\mathcal{S} \leftarrow \emptyset$
for $i \leftarrow 1$ **to** k **do**
  $v^{\pm}_{prev} \leftarrow \text{NaN}$
  for $j \leftarrow i+1$ **to** k **do**
    $\Delta p \leftarrow p_i - p_j$
    if $\|\Delta p\|^2 < \epsilon_{chord}$ **then continue**
    `// Check tangent condition:` $\Delta p \cdot (t_i \pm t_j) = 0$
    if $|t_i \cdot t_j| < 1 - \epsilon$ **then**
      $v^{\pm}_{curr} \leftarrow \Delta p \cdot (t_i \pm t_j)$
      if $v^{\pm}_{prev} \cdot v^{\pm}_{curr} < 0$ **then**
        `// Sign change detected: Normals approx intersect`
        $C \leftarrow \text{ComputeIntersection}(p_i, n_i, p_j, n_j)$
        $r_i \leftarrow \|C - p_i\|, \quad r_j \leftarrow \|C - p_j\|$
        if $|r_i - r_j| < r_i \cdot \epsilon_{rad}$ **then**
          $\mathcal{S} \leftarrow \mathcal{S} \cup \{C\}$
      $v^{\pm}_{prev} \leftarrow v^{\pm}_{curr}$
return $\mathcal{S}$

4 Examples and Demonstration

To demonstrate the utility of our tool, we present a series of visualizations highlighting the interplay between the geometry of the input curve $\mathcal{M}$ and its vineyard topology.

4.1 Symmetry and Focal Sets of Deforming Ellipses

We begin by validating our tool against known geometric behaviors. Figure 2 illustrates the evolution of the focal and symmetry sets for a family of ellipse-like shapes, inspired by the work of [9]. As the ellipse is deformed from a convex shape into a non-convex "bean" shape, we observe the characteristic bifurcations in the symmetry set (red) and the formation of cusps in the focal set (green). The tool accurately captures the emergence of "swallowtail" singularities in the focal set, which correspond to the creation of inflection points on the curve, as shown in [10].

4.2 Complexity in Arbitrary Splines

The tool's robustness is further tested on complex, hand-picked spline curves (Figure 3). Even for highly non-convex shapes with multiple lobes, the discrete algorithm successfully resolves intricate symmetry set branches and the global structure of the focal set. Furthermore, the tool is capable of handling inputs with multiple connected components. As seen in the bottom right of Figure 3, the visualization captures not only the local symmetry within each individual curve but also the global bitangential interactions *between* the two disjoint shapes.

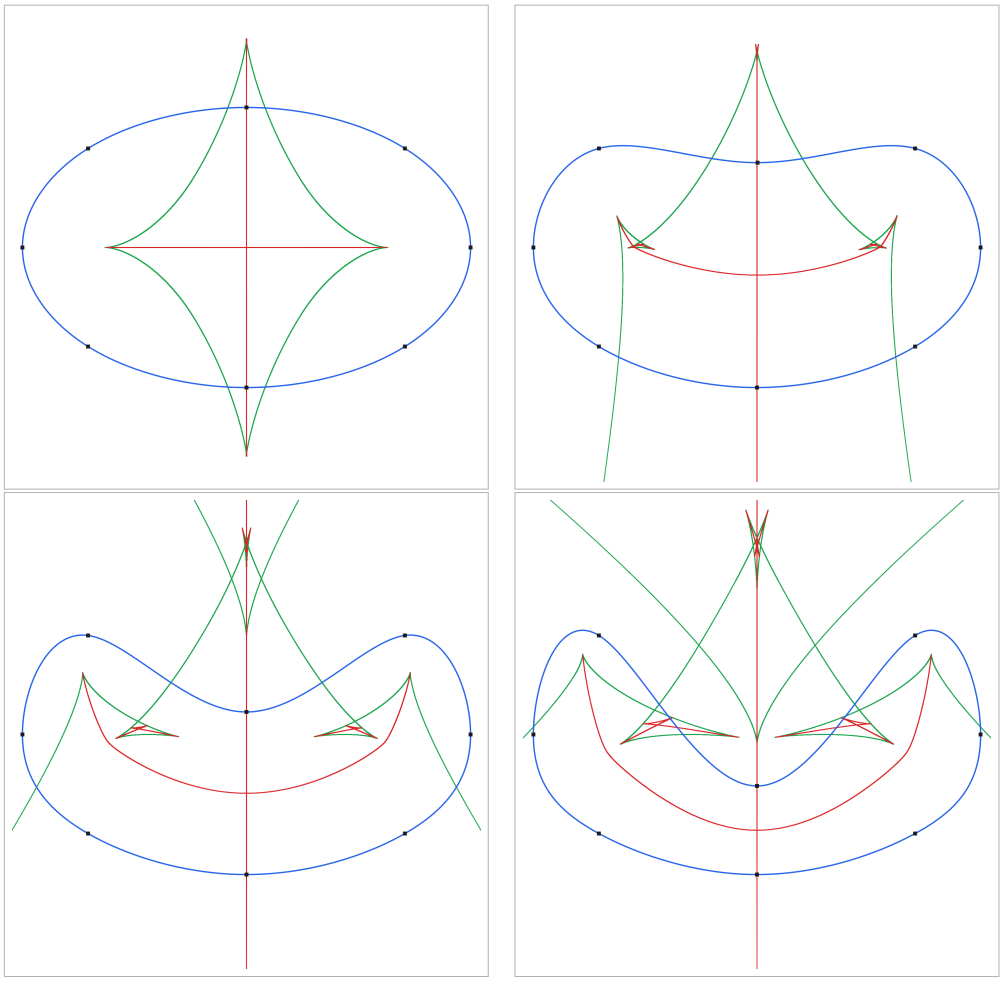


Figure 2 Focal (green) and symmetry (red) sets of ellipses undergoing deformation (inspired from [9]). The sequence captures the transition from a pure ellipse to a non-convex shape, showing the creation of cusps and self-intersections in the singularity sets.

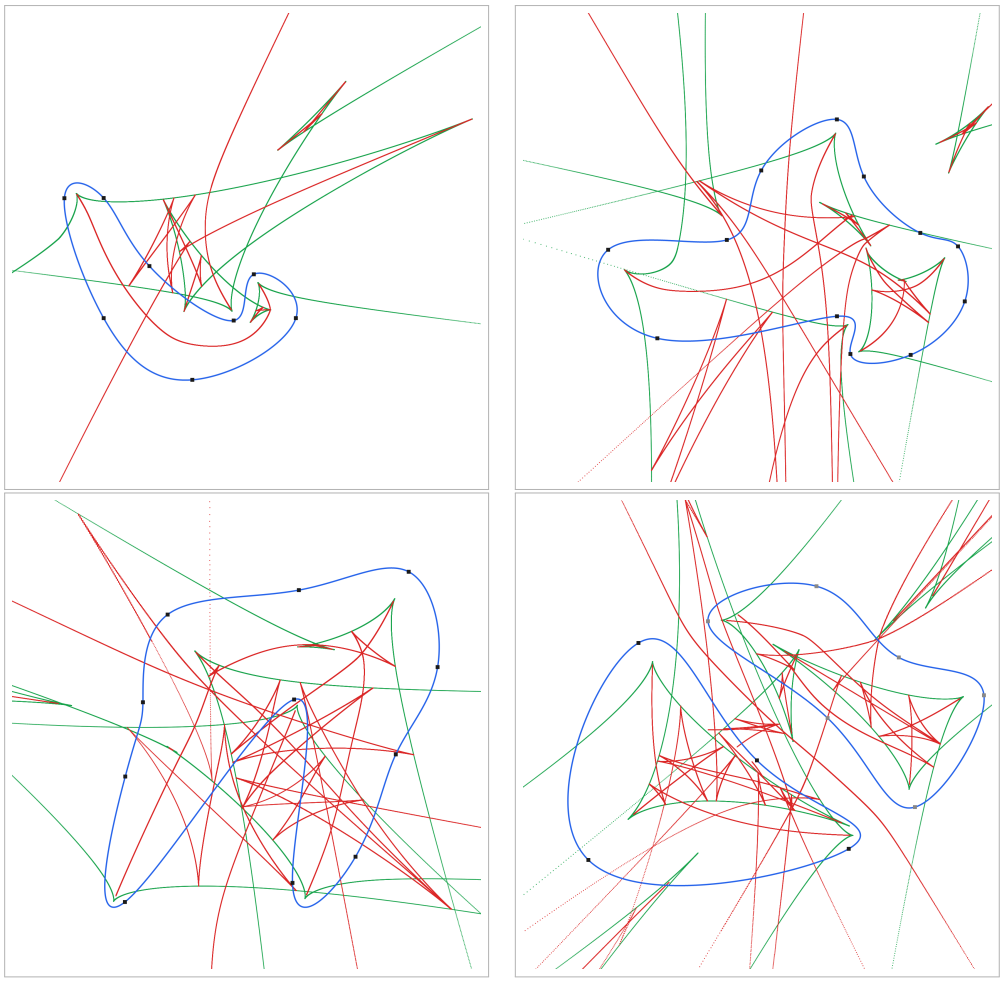


Figure 3 Complex symmetry and focal sets arising from arbitrary spline curves. The figure on the bottom right captures the symmetry and focal sets arising from 2 connected components.

4.3 Vineyards and Singularity Crossings

Finally, we examine the stability of the Radial Persistence Transform by computing the vineyard along a circular loop γ (shown in purple). Figure 4 explicitly visualizes that topological changes in the distance function occur precisely when the center of measurement crosses the focal set.

In the top row of Figure 4, the yellow center point traverses a region free of focal singularities, and the corresponding vine (blue curve in the 3D plot) remains continuous. As the center crosses the green focal curve (middle and bottom rows), we observe a distinct "break" or discontinuity in the vineyard. Figure 5 further explores this by varying the radius of the loop γ, showing how the vines evolve as the measurement loop encompasses different topological features of the ambient space.

Figure 6 shows a particularly compelling case, where the tool computes the vineyard for two disjoint connected components using extended persistence. This is particularly useful to track global spatial relationships along the observation loop.

Figure 7 is a demonstration of a vineyard where the measurement center moves along a arbitrary hand-drawn loop via splines; the resulting vineyard captures the topological response to a non-standard path, illustrating how custom trajectories can be used to probe specific regions of the ambient space.

5 Outlook: Feedback please!

We invite everyone to test the tool and provide feedback and comments on how it can be improved. Currently it supports 2D only. Future works will include extensions to higher dimensions.

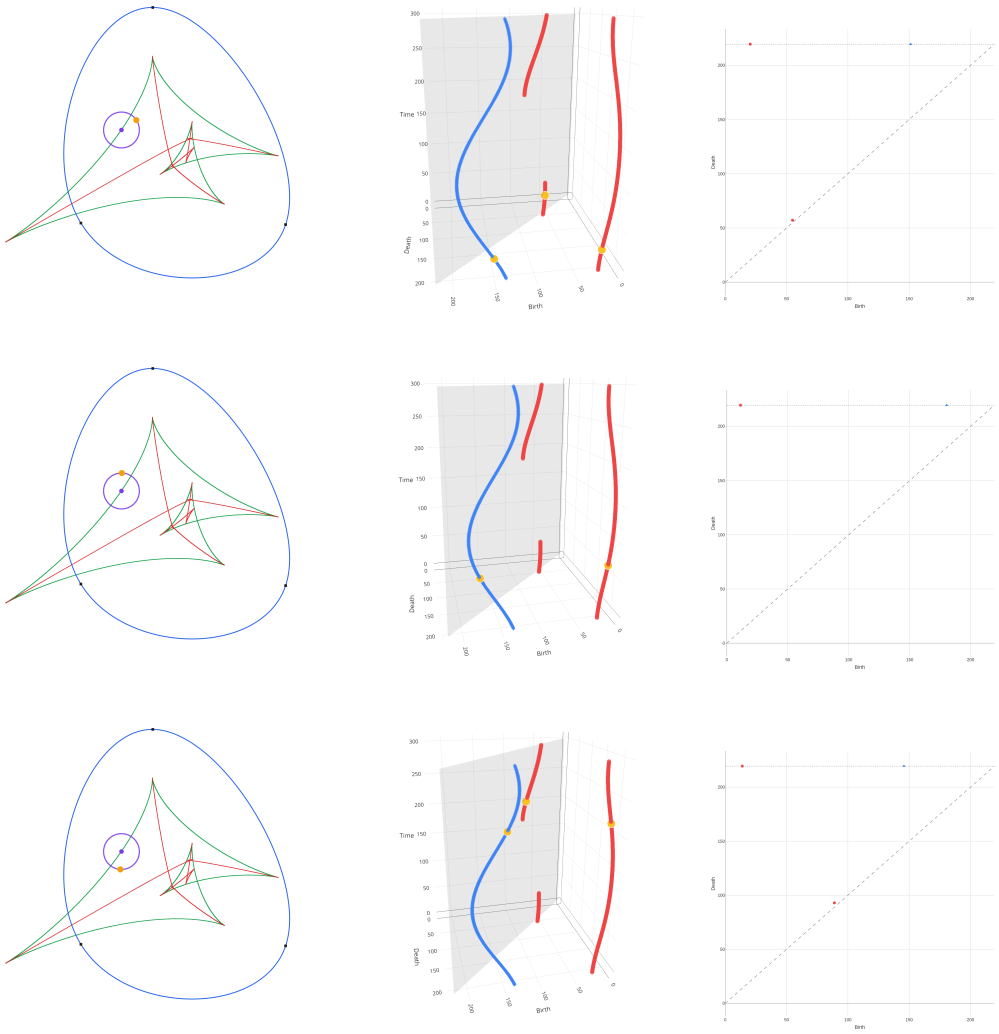


Figure 4 An example of a vineyard for a potato shaped figure along the purple loop. As the center (yellow point) crosses the focal set, there is a break in the vineyard, and when it crosses back again, the vineyard reappears.

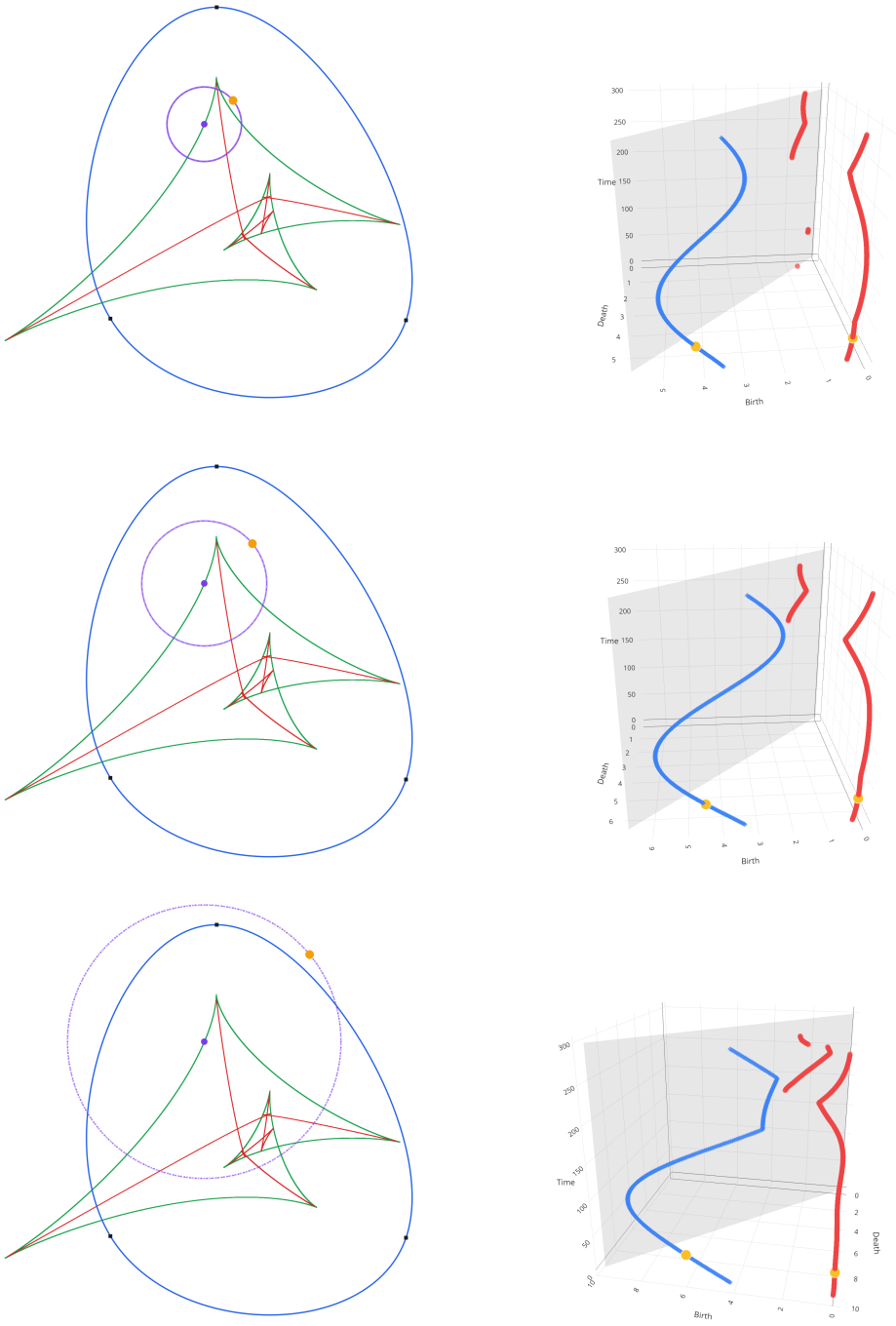


Figure 5 Demonstration about how the vineyard changes as the radius of the center circle changes.

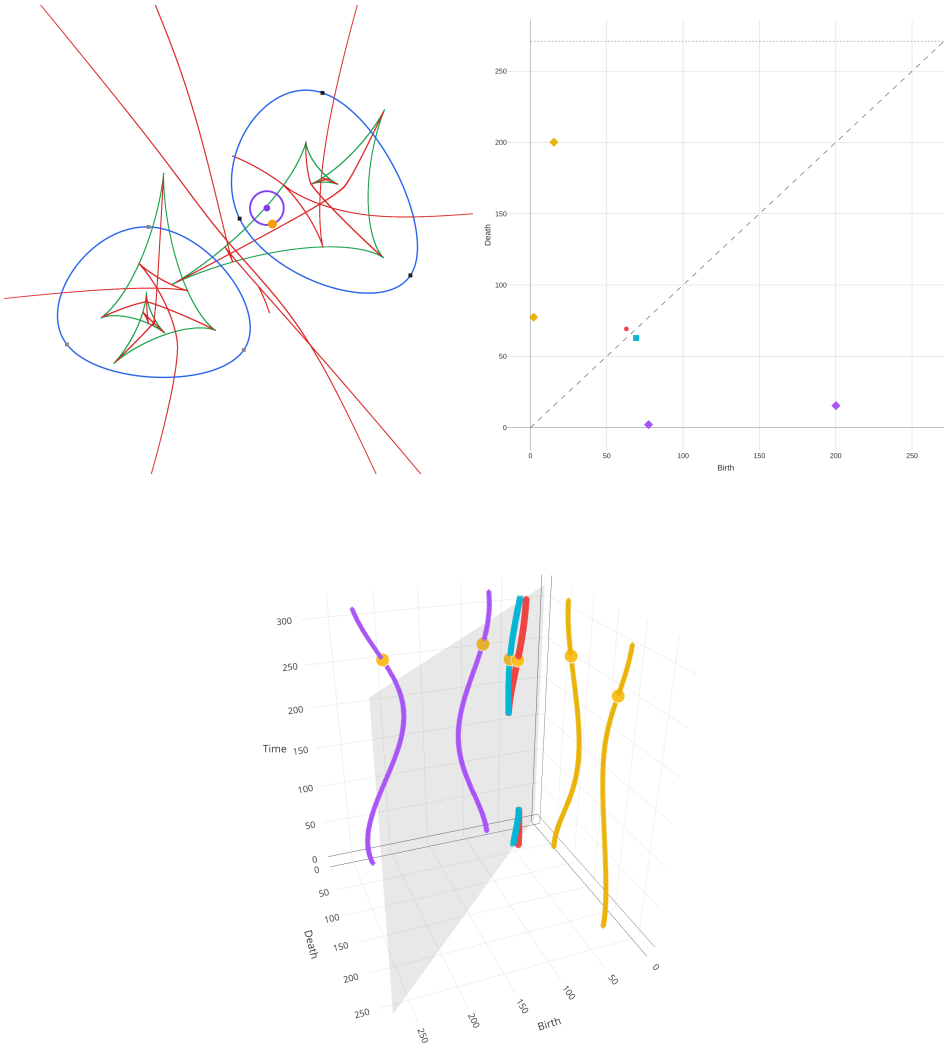


Figure 6 Demonstration of a vineyard for 2 disjoint connected components, with extended persistence.

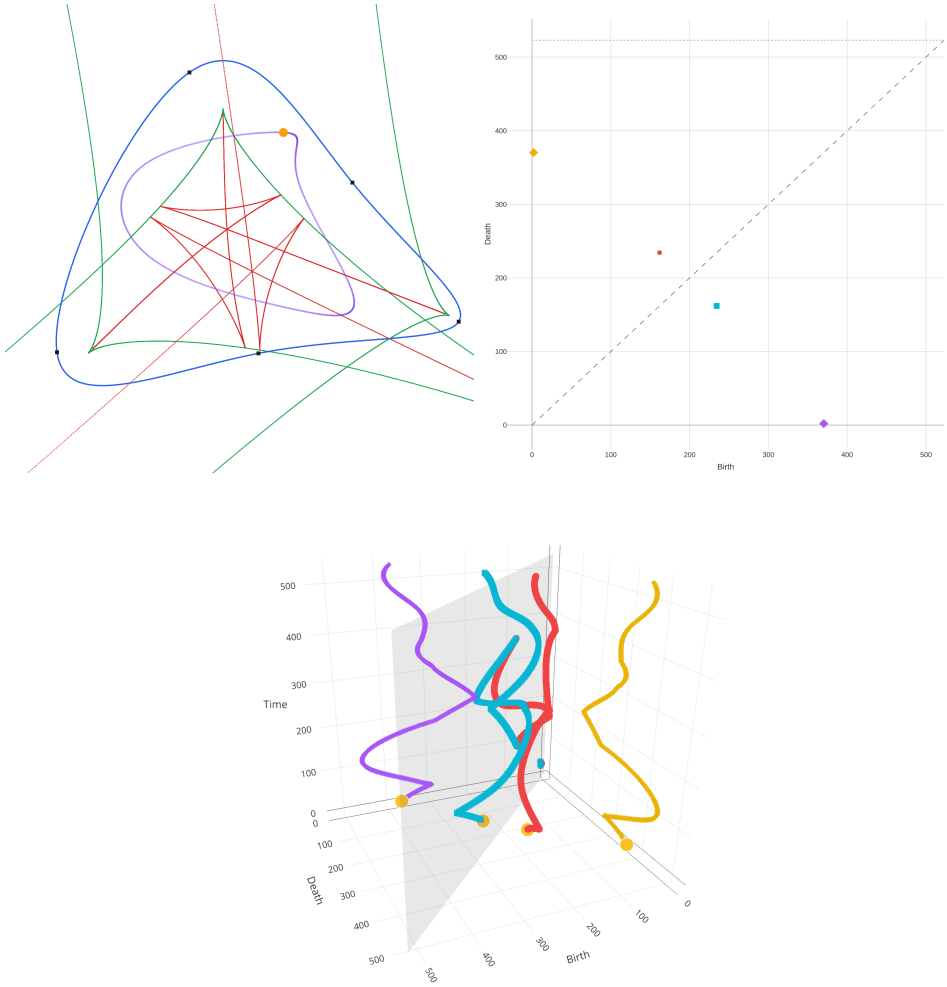


Figure 7 Vineyard for a arbitrary hand-drawn observation loop.

References

1 Shreya Arya, Barbara Giunti, Abigail Hickok, Lida Kanari, Sarah McGuire, and Katharine Turner. Decomposing the persistent homology transform of star-shaped objects, 2024. URL: https://arxiv.org/abs/2408.14995, doi:10.48550/ARXIV.2408.14995.

2 Mattia G. Bergomi and Adriano Baratè. Homological persistence in time series: an application to music classification. Journal of Mathematics and Music, 14(2):204–221, May 2020. URL: http://dx.doi.org/10.1080/17459737.2020.1786745, doi:10.1080/17459737.2020.1786745.

3 James W. Bruce, Peter J. Giblin, and Christopher G. Gibson. Symmetry sets. Proceedings of the Royal Society of Edinburgh: Section A Mathematics, 104A:179–204, 1985.

4 Erin W. Chambers, Christopher Fillmore, Elizabeth Stephenson, and Mathijs Wintraecken. Braiding vineyards. In Proceedings of the 2026 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA), page 6240–6263. Society for Industrial and Applied Mathematics, January 2026. doi:10.1137/1.9781611978971.225.

5 David Cohen-Steiner, Herbert Edelsbrunner, and Dmitriy Morozov. Vines and vineyards by updating persistence in linear time. In Proceedings of the twenty-second annual symposium on Computational geometry, SoCG06, page 119–126. ACM, June 2006. URL: http://dx.doi.org/10.1145/1137856.1137877, doi:10.1145/1137856.1137877.

6 Carl de Boor. A Practical Guide to Splines. Springer New York, 1978. URL: http://dx.doi.org/10.1007/978-1-4612-6333-3, doi:10.1007/978-1-4612-6333-3.

7 Shannon Dee Algar, Débora C. Corrêa, and David M. Walker. On detecting dynamical regime change using a transformation cost metric between persistent homology diagrams. Chaos: An Interdisciplinary Journal of Nonlinear Science, 31(12), December 2021. URL: http://dx.doi.org/10.1063/5.0073247, doi:10.1063/5.0073247.

8 Tamal K. Dey, Tao Hou, and Salman Parsa. Revisiting graph persistence for updates and efficiency, 2023. URL: https://arxiv.org/abs/2302.12796, doi:10.48550/ARXIV.2302.12796.

9 Peter Giblin. Symmetry sets and medial axes in two and three dimensions. In Roberto Cipolla and Ralph Martin, editors, The Mathematics of Surfaces IX, pages 306–321, London, 2000. Springer London.

10 Peter Giblin and André Diatta. Symmetry sets and medial axes in two dimensions, 2003. URL: https://www.researchgate.net/profile/Peter-Giblin-2/publication/221403718_Symmetry_Sets_and_Medial_Axes_in_Two_and_Three_Dimensions/links/5cc0825b299bf120977d5eb5/Symmetry-Sets-and-Medial-Axes-in-Two-and-Three-Dimensions.pdf.

11 Ian R Porteous. The normal singularities of a submanifold. Journal of Differential Geometry, 5(3-4):543–564, 1971.

12 Ian R Porteous. Geometric differentiation: for the intelligence of curves and surfaces. Cambridge University Press, 2001.

13 Katharine Turner. Representing vineyard modules. arXiv preprint arXiv:2307.06020, 2023.

14 Jaejun Yoo, Eun Young Kim, Yong Min Ahn, and Jong Chul Ye. Topological persistence vineyard for dynamic functional brain connectivity during resting and gaming stages. Journal of Neuroscience Methods, 267:1–13, July 2016. URL: http://dx.doi.org/10.1016/j.jneumeth.2016.04.001, doi:10.1016/j.jneumeth.2016.04.001.

The Domino Problem is Decidable for Robust Tilesets

Nathalie Aubrun[1], Manon Blanc[1, 2, 3], and Olivier Bournez[3]

1 **Université Paris-Saclay, LISN**
`aubrun@lisn.fr`
2 **IT-Universitet i København**
`mbrb@itu.dk`
3 **École Polytechnique, LIX**
`bournez@lix.polytechnique.fr`

Abstract

One of the most fundamental problems in tiling theory is the domino problem: given a set of tiles and tiling rules, decide if there exists a way to tile the plane. The problem is known to be undecidable in general. We prove that the domino problem is decidable for robust tilesets, *i.e.* tilesets that either cannot tile the plane or can by provably satisfying some particular invariant. We establish that several famous tilesets considered in the literature are robust. As a side effect of our work, we provide a sound, relatively complete method for proving that a tileset can tile the plane. Our analysis also provides explanations for the similarities between proofs in the literature for various tilesets, as well as of phenomena that have been observed experimentally in the systematic study of tilesets using computer-assisted methods.

Related Version https://link.springer.com/book/9783032156402 and https://arxiv.org/abs/2402.04438

1 Introduction

Wang tiles were introduced in the early '60s to study the undecidability of the $\forall\exists\forall$ fragment of first-order logic [7]. They consist of unit square tiles with colours on their edges. A tiling of the infinite Euclidean plane is obtained using arbitrarily many copies of the tiles, with no overlap, in the given finite tileset. The tiling is *valid* if everywhere the contiguous edges have the same colour. The *domino problem* is the decision problem which takes as input τ a finite set of Wang tiles, and outputs `Yes` if there exists a valid tiling by τ and `No` otherwise. A famous result by Berger states that the Domino problem is undecidable [1]. Berger's original proof relies on the construction of an aperiodic tileset, that is used to perform Turing machine computations inside tilings. All known proofs rely on the existence of an aperiodic tileset, even if the techniques can be different. For most of the known aperiodic tilesets, once the tileset has been explicitly built, proving aperiodicity is not that complicated, but proving a tiling exists may reveal more difficulty. Yet, no general technique to prove an aperiodic tileset admits a valid tiling is known. One objective of this paper is to set a formalism unifying proofs of the existence of tilings for aperiodic tilesets.

Using the formalism of transducers, we introduce two notions of *robust* tilesets (Definitions 3.1 and 3.3), according to whether some semantic or provably invariant holds. These concepts are motivated by our exploration of Robinson's tileset with transducers but are not specific to it. Indeed, an analogous semantic statement can be written for every tileset for which one can prove it admits a valid tiling: Propositions 3.2 and 3.2. We do so for Robinson's tileset, Jeandel-Rao's tileset and Kari's tilesets.

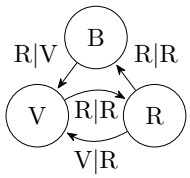
42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

2 Wang tilesets as transducers

2.1 Transducers and meta-transducers

▶ **Definition 2.1** (Transducer). Let Σ be a finite alphabet. A *transducer* is a triple (Q, Σ, δ) with Q a finite set of states, Σ a finite alphabet and $\delta \subseteq Q \times Q \times \Sigma \times \Sigma$ a set of transitions.

The transducer $\mathcal{T}_1$ with $Q = \{V, B, R\}$, $\Sigma = Q$ is pictured on the right: The fact that $(q, r, w, w') \in \delta$ is represented by an edge between q and r labelled by $w|w'$. To simplify notations we use *meta-transducers* rather than transducers. They allow us to deal not only with single letters but also with words of arbitrary finite length as labels of transitions.



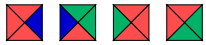
▶ **Definition 2.2** (Meta-transducer). A *meta-transducer* is a triple (Q, Σ, δ) with Q a finite set of states, Σ a finite alphabet and $\delta \subseteq Q \times Q \times \Sigma^* \times \Sigma^*$ a set of transitions, or edges, with $|w|, |w'| \geq 1$ where $w, w' \in \Sigma^*$ are words and $|w| = |w'|$, where $|w|$ denotes the length of w.

The transition relation δ^* is the smallest relation, containing δ, such that, whenever $(q, r, x, y) \in \delta^*$ and $(r, s, w, w') \in \delta$, then $(q, s, xw, yw') \in \delta^*$. If $\mathcal{T}$ is a (meta-)transducer, for any $w, w' \in \Sigma^*$, we write $w\mathcal{T}w'$ if there exist $q, s \in Q$ and $w, w' \in \Sigma^*$ s.t. $(q, s, w, w') \in \delta^*$. This is equivalent to say that there exists a sequence $q_0, \dots, q_n$ s.t. $(q_i, q_{i+1}, w_i, w'_i)$ for every $i = 0 \dots n-1$, where $w = w_1 \dots w_n$, $w' = w'_1 \dots w'_n$ and the w_i, w'_i are words on Σ.

Consider $\mathcal{T} = (Q, \Sigma, \delta)$ a meta-transducer. A transition $(q, q, w, w') \in \delta$ is called a *loop*. Moreover, if the word w' is a cyclic permutation of w, i.e. if there exist two non-trivial words u, v such that $w = uv$ and $w' = vu$, we say that the loop (q, q, w, w') is a *cyclic loop*. Finally, a loop $(q, q, w, w) \in \delta$ is called a *periodic loop*. Every periodic loop is a cyclic loop, but the converse is not true. A transducer *contains a loop* if there exists some $h : [[0 \dots m]] \to Q$ such that $(h(i), h(i+1), w_i, w'_i) \in \delta$ with $h(m+1)$ abusively defined as $h(0)$.

2.2 Tiles and transducers

A *Wang tileset* T is a triple (H, V, τ) where H is a finite set that corresponds to the horizontal colours, V is a finite set that corresponds to vertical colours and $\tau \subseteq H^2 \times V^2$ is the set of tiles. We denote $t = (w, e, s, n) \in \tau$ *a tile* in T where w (resp. e, s, n) is the colour on the left (resp. right, bottom, top) edge. For a tile t, we also write t_w, t_e, t_s and t_n the colours of its edges. A Wang tileset expressed as triple (H, V, τ) exactly matches the definition of a transducer of Section 2.1 where tiles are transitions, horizontal colours H are states Q and vertical colours V are letters of the alphabet Σ. Here is an example a of meta-transducer derived from the tileset τ composed by the four tiles pictured on the top:



From now on we will use tileset or transducer interchangeably without any possible confusion. A natural operation on transducers, denoted by $\cup$, is the union, defined as expected. Any transducer is thus the union of its edges. A *tiling* x by τ (or by T) is a mapping $x : \mathbb{Z}^2 \to \tau$ such that adjacent edges share the same colour, i.e. for every $(i, j) \in \mathbb{Z}^2$, $x(i, j)_n = x(i, j+1)_s$ and $x(i, j)_e = x(i+1, j)_w$. The tiling x is *periodic* with period $(a, b) \in \mathbb{Z}^2 \backslash \{(0, 0)\}$ iff $x(u, v) = x(u+a, v+b)$ for every $(u, v) \in \mathbb{Z}^2$. If there exists a periodic valid tiling with tiles from τ, then there exists a *doubly periodic* valid tiling, i.e. a tiling x such that, for some $a, b > 0, x(u, v) = x(u+a, v) = x(u, v+b)$ for all $(u, v) \in \mathbb{Z}^2$ (see e.g. [4]). A tileset τ is called *aperiodic* iff the two following conditions are satisfied (i) there exists a valid tiling and (ii) there does not exist any periodic valid tiling. A tileset τ is *compatible* iff the associated transducer $\mathcal{T}$ contains a loop.

▶ **Definition 2.3** (Notation $\mathcal{T} \circ \mathcal{T}'$)**.** If (Q, Σ, δ) and (Q', Σ', δ') are two meta-transducers, then the *composition of* $\mathcal{T}$ *and* $\mathcal{T}'$ is the meta-transducer $(Q \times Q', \Sigma \times \Sigma', \delta'')$ where $|w| = |w'|$ and $((w, w'), (e, e'), s, n') \in \delta''$ iff we have $(w, e, s, n) \in \delta$ and $(w', e', s', n') \in \delta'$ and $n = s'$.

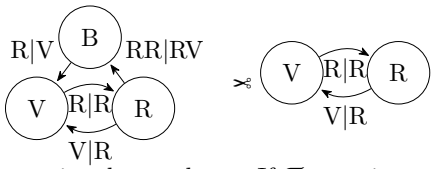
▶ Proposition 2.4. The composition of two meta-transducers is a meta-transducer.

As a corollary, if we start with two Wang tilesets, their composition is still a Wang tileset.

As a direct consequence of the definition, we relate the compatibility of the composition $\mathcal{T}^n$ with the existence of a valid tiling of a horizontal strip.

▶ Proposition 2.5. Consider a transducer $\mathcal{T}$. There exist $w, w' \in V^{\mathbb{Z}}$ such that $w\mathcal{T}^n w'$ iff there exists a valid tiling by τ of $\mathbb{Z} \times [0; n-1]$ the horizontal strip of height n.

▶ **Definition 2.6** (Notation $\mathcal{T} \succ\!\!\!\circ \mathcal{T}'$)**.** Given two transducers $\mathcal{T}$ and $\mathcal{T}'$, we write $\mathcal{T} \succ\!\!\!\circ \mathcal{T}'$ iff every edge of the transducer $\mathcal{T}'$ is an edge of $\mathcal{T}$. See example below.



As such, when we write $\mathcal{T} \succ\!\!\!\circ \mathcal{T}'$ and the states of $\mathcal{T}$ are of height n, then $\mathcal{T}'$ is necessarily made of states of height n.

▶ Proposition 2.7. Let τ be a tileset and $\mathcal{T}$ the associated transducer. If $\mathcal{T}$ contains a cyclic loop of size (or order) N then $\mathcal{T}^m$ contains a periodic loop for some $1 \leq m \leq N$.

▶ Proposition 2.8. Consider a transducer $\mathcal{T}$. Then $\mathcal{T}^n$ contains a loop iff there exists a valid tiling by τ of $\mathbb{Z} \times [0; n-1]$ the horizontal strip of height n. Then $\mathcal{T}^n$ contains a loop of order N iff there exists a valid tiling by τ of $\mathbb{Z} \times [0; n-1]$ the horizontal strip of height n using a pattern repeated infinitely with period N.

3 Two notions of robustness for tilesets

Consider a finite set S of transducers. Consider the signature made of the symbols of S (with arity 0) and the symbol $\circ$ (of arity 2). A *composition pattern* F over S is a term over this signature. For example $H \circ \tau \circ H$ is a composition pattern over $\{H, \tau\}$. Such a composition pattern is interpreted as expected: $\circ$ corresponds to composition.

▶ **Definition 3.1.** A tileset τ is **semantically robust** (or *inductive*) if there exists a family of transducers $(\mathcal{T}_n)_n$, of heights $(g(n))_n$, with $g : \mathbb{N}^* \to \mathbb{N}^*$ increasing and injective with $g(1) = 1$ and some (fixed) composition pattern $F = F(\mathcal{T}_1, \mathcal{T}_{n-k}, \ldots, \mathcal{T}_n)$ over $\mathcal{T}_1, \mathcal{T}_{n-k}, \ldots, \mathcal{T}_n$ for some integer k, such that: (1.) Initialisation: $\tau \succ\!\!\!\circ \mathcal{T}_1$, $\ldots$, $\tau^{g(k+1)} \succ\!\!\!\circ \mathcal{T}_{k+1}$; (2.) F provides an invariant: $\forall n, \quad F(\mathcal{T}_1, \mathcal{T}_{n-k}, \ldots, \mathcal{T}_n) \succ\!\!\!\circ \mathcal{T}_{n+1}$; (3.) For all n, $\mathcal{T}_n$ contains a loop.

▶ **Theorem 3.2.** *A tileset τ admits a tiling iff it is semantically robust.*

There might be a strong difference between the fact that $\forall n, F(\mathcal{T}_1, \mathcal{T}_{n-k}, \ldots, \mathcal{T}_n) \succ\!\!\!\circ \mathcal{T}_{n+1}$ holds for some composition pattern F, written as $\models \exists F, \forall n, F(\mathcal{T}_1, \mathcal{T}_{n-k}, \ldots, \mathcal{T}_n) \succ\!\!\!\circ \mathcal{T}_{n+1}$ and the fact that we can prove it holds, written $\vdash \exists F, \forall n, F(\mathcal{T}_1, \mathcal{T}_{n-k}, \ldots, \mathcal{T}_n) \succ\!\!\!\circ \mathcal{T}_{n+1}$. This is for instance the case if a tileset admits a tiling, but we cannot prove the existence of any valid tiling. The same phenomenon happens for Turing machines, for which non-termination may hold while there is no proof of it. In the meantime we introduce the concept of *provably robust* tileset to add this condition.

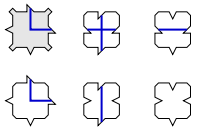
▶ **Definition 3.3.** A tileset τ is **provably robust** if there exists a family of transducers $(\mathcal{T}_n)_n$, of respective heights $(g(n))_n$, with $g : \mathbb{N}^* \to \mathbb{N}^*$ increasing and injective, $g(1) = 1$ and

some (fixed) composition pattern $F = F(\mathcal{T}_1, \mathcal{T}_{n-k}, \ldots, \mathcal{T}_n)$ over $\mathcal{T}_1, \mathcal{T}_{n-k}, \ldots, \mathcal{T}_n$ for some integer k, (1) Initialisation: $\tau \succ_{\mathrm{s}} \mathcal{T}_1, \ldots, \tau^{g(k+1)} \succ_{\mathrm{s}} \mathcal{T}_{k+1}$ (2) F provides an invariant **holding provably**: $\forall n, F(\mathcal{T}_1, \mathcal{T}_{n-k}, \ldots, \mathcal{T}_n) \succ_{\mathrm{s}} \mathcal{T}_{n+1}$ (3) For all n, $\mathcal{T}_n$ contains a loop.

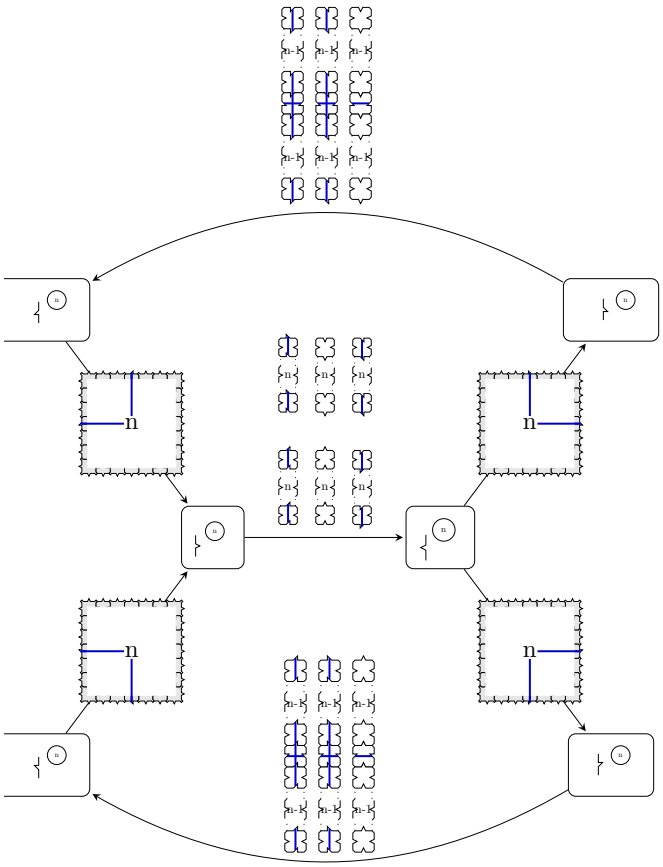
▶ Proposition 3.4. If a tileset τ is provably robust then it admits a tiling.

▶ Proposition 3.5. A periodic tileset is provably robust.

4 Example of Robust Tilesets: Robinson's tileset



A famous tileset is Robinson's tileset [6] which simplifies the construction of Berger [1]. This aperiodic Wang tileset can be represented more compactly with the geometric tiles pictured here, which may be rotated and reflected. Formally, they are not Wang tiles but they can be transformed in such, using Robinson's arguments [6]. We convert this tileset into a transducer by defining two alphabets for horizontal and vertical colours: $H := \{$ ⌒ , ⌒ , ⌒ , ⌒ , ⌒ , ⌒ $\}$; $V := \{$ ⊢ , ⊣ , ⊢ , ⊢ , ⊣ , ⊣ $\}$. Robinson's tileset can be associated with the transducer $\mathcal{T}_{Robi}$:



For more readability, in this figure, we label a transition from a state q to a state q' by the corresponding tile from Robinson's tileset with the proper orientation.

▶ Proposition 4.1. For all $n \geq 1$, $\mathcal{T}_n$ contains a loop.

▶ Proposition 4.2. Robinson's tileset is provably robust.

5 Example of Robust Tilesets: Jeandel-Rao's tileset

Jeandel Rao's aperiodic tileset ([3]) $\mathcal{T}$ is the Wang tileset composed with the 11 tiles:

▶ **Theorem 5.1.** *Jeandel Rao's aperiodic tileset $\mathcal{T}$ is provably robust.*

6 Example of Robust Tilesets: Kari's tilings

In this section, we briefly present a technique due to Kari to encode piecewise rational affine maps computations inside Wang tilings. We encourage the readers to refer to [5] for details about the construction.

▶ **Definition 6.1.** The tileset τ_{f_i} corresponding to a rational affine map $f_i(\vec{x}) = M\vec{x}+\vec{b}$ and its domain cube U_i is all the tiles (pictured below), where $k \in \mathbb{Z}$, $\vec{x} \in U_i$, and $C_{f_i,k}(\vec{x})$ is a rational number depending on $f_i, \vec{x}$ and k (details can be found in [5]) and $B_k(x) = \lfloor kx \rfloor - \lfloor (k-1)x \rfloor$.

$$\begin{array}{ccc} & B_k(f_i(\vec{x})) & \\ C_{f_i,k-1}(\vec{x}) & \square & C_{f_i,k}(\vec{x}) \\ & B_k(\vec{x}) & \end{array}$$

Every tile as pictured to the left locally computes f_i with some error since we have $f_i\left(B_k(\vec{x})\right) + C_{f_i,k-1}(\vec{x}) = B_k\left(f_i\left(\vec{x}\right)\right) + C_{f_i,k}(\vec{x})$. Moreover, because f_i is rational, there are only finitely many such tiles (even though there are infinitely many $k \in \mathbb{Z}$ and $\vec{x} \in U_i$). For fixed $\vec{x} \in U_i$ the tiles for consecutive $k \in \mathbb{Z}$ match so that a horizontal row can be formed whose top and bottom labels bear the balanced representations of $\vec{x}$ and $f_i(\vec{x})$.

Kari considers the tileset τ_f obtained as the union of the τ_{f_i}, with additional rules to ensure that only tiles from the same τ_{f_i} appear on a given row. Tilings can also be associated with Turing machines. In this context, a Turing machine is said to be non-robust if it does not halt and furthermore does so non-provably.

▶ **Theorem 6.2.** *Consider a Turing machine $\mathcal{M}$. Then $\mathcal{M}$ is provably robust and not terminating iff $\tau_{f_\mathcal{M}}$ is provably robust.*

7 Conclusions and discussions

Going back to the domino problem, we have the following theorem:

▶ **Theorem 7.1.** *The domino problem is decidable for robust tilesets.*

The method we described to prove that a given tileset can tile the plane is based on proving invariants of the same form as in Definition 3.1. The method of finding an invariant of the proper form for some composition pattern F to prove that a tileset can tile the plane can be seen as a sound and relatively complete method. Jeandel-Rao's tileset $\mathcal{T}$ considered in Section 5 arose from a systematic computer-assisted proof of tilesets by considering an increasing number of tiles, conducted in [3]. The authors proved (using the help of about 23 CPU years of computing done on a cluster of parallel machines) that there is no aperiodic Wang set with 10 tiles or less. Our study provides theoretical explanations for the experimental facts mentioned in [3, Section 8].

Our settings pave the way for important and fascinating developments. First, when a tileset is robust, we know it is decidable. But then it makes sense to discuss the complexity of the domino problem and not only its decidability. This can be done by quantifying the robustness in an approach similar to the one done in [2]. Furthermore, we think that similar statements can be obtained not only about the fact that the plane can be tiled but also about the fact that the plane can be tiled aperiodically.

References

1 R. Berger. The undecidability of the domino problem. *Memoirs American Mathematical Society*, 66, 1966.

2 Manon Blanc and Olivier Bournez. Quantifiying the robustness of dynamical systems. relating time and space to length and precision. In *Computer Science Logic CSL'24*, Naples, Italy, February 2024.

3 Emmanuel Jeandel and Michael Rao. An aperiodic set of 11 wang tiles. *Advances in Combinatorics*, 2021.

4 Jarkko Kari. A small aperiodic set of wang tiles. *Discrete Mathematics*, 160(1-3):259–264, 1996.

5 Jarkko Kari. The Tiling Problem Revisited. In Jérôme Olivier Durand-Lose and Maurice Margenstern, editors, *MCU*, volume 4664 of *Lecture Notes in Computer Science*, pages 72–79. Springer, 2007.

6 R. M. Robinson. Undecidability and nonperiodicity for tilings of the plane. *Inventiones Mathematicae*, 12:177–209, 1971.

7 Hao Wang. Proving theorems by pattern recognition - ii. *Bell System Technical Journal*, 40(1):1–41, January 1961. URL: `http://dx.doi.org/10.1002/j.1538-7305.1961.tb03975.x`, `doi:10.1002/j.1538-7305.1961.tb03975.x`.

Drone Air Traffic Control: Tracking a Set of Moving Objects with Minimal Power*

Sándor P. Fekete[1,2], Malte Hoffmann[1], Chek-Manh Loi[1], and Michael Perk[1]

1 **Department of Computer Science, TU Braunschweig**
s.fekete@tu-bs.de, {mhoffman,loi,perk}@ibr.cs.tu-bs.de
2 **L3S, Germany**

Abstract

A common sensing problem is to use a set of stationary tracking locations to monitor a collection of moving devices: Given n objects that need to be tracked, each following its own trajectory, and m stationary traffic control stations, each with a sensing region of adjustable range; how should we adjust the individual sensor ranges in order to optimize energy consumption? We present an algorithm based on geometric insights that is able to find optimal solutions for the min max variant of the problem, which aims at minimizing peak power consumption. Instances with 500 moving objects and 25 stations can be solved in the order of seconds for scenarios that take minutes to play out in the real world, demonstrating real-time capability of our methods.

Related Version *Full version and code* [14–16]

1 Introduction

Keeping track of a collection of moving objects is a fundamental problem with a long history in Computational Geometry. In air traffic control, stationary centers monitor planes with powerful tracking devices to coordinate overall motion. With the growing ubiquity of drones, air traffic control needs further development [4,17], such as the use of less powerful tracking stations: How should we adjust sensing radii over time to minimize power consumption?

Formally, this is the *Kinetic Disk Covering Problem* (KDC), see Figure 1 for an illustration: Given a set $\mathcal{P} = \{p_1, \ldots, p_n\}$ of n points that move at constant speeds along linear trajectories $p_i(t) : [0,1] \to \mathbb{R}^2$ over the time interval $[0,1]$ as well as a set $\mathcal{Y} \subset \mathbb{R}^2$ of m centers, the KDC problem asks for an assignment of radii $r_i(t)$ to each center $y_i \in \mathcal{Y}$ at each time t such that all points are covered by at least one disk and the maximum total area of all disks at any time is minimized, i.e., $\min \max_{t\in[0,1]} \sum_{i=1}^{m} \pi r_i(t)^2$. The problem of finding an area-optimal assignment for a (stationary) set of points is known as the *Disk Covering Problem* [1] (DC) and known to be NP-hard. In an optimal solution for the KDC, the radius of each disk is determined by the farthest point assigned to it. We call these *supporting points*. A solution to the KDC is thus a list of k assignments $\{A_{t_0}, A_{t_1}, \ldots, A_{t_k}\}$ with $0 = t_0 \le t_1 \le \ldots \le t_k = 1$. Each assignment $A_t : (\mathcal{Y} \to \mathcal{P})$ assigns supporting points to centers for a specific time window. The radius of the disk centered at y_i at time $t \in [t_j, t_{j+1})$ is simply the distance from y_i to its supporting point, i.e., $r_i(t) = \left\| y_i - A_{t_j}(y_i)(t) \right\|$.

In this work, we present exact and heuristic methods for the KDC. Our approaches start from a stationary solution at time $t = 0$ and use geometric insights to extend it over time (thus producing a feasible solution for all $t \in [0,1]$). We can then locate the time t' at which

* Supported by the German Research Foundation (Deutsche Forschungsgemeinschaft, DFG) as part of project *Computational Geometry: Solving Hard Optimization Problems* (CG:SHOP) - 444569951.

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

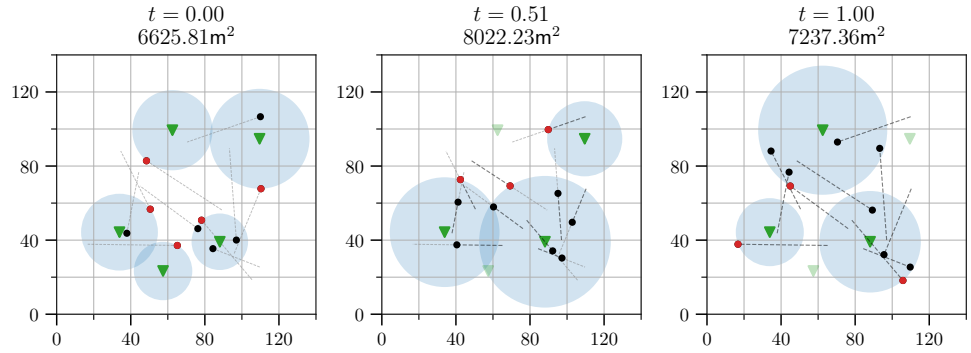


Figure 1 A solution for an instance of the Kinetic Disk Covering Problem for $n = 10$ and $m = 5$. Stations are shown as green triangles, points move along the dotted lines. We start with an IP solution at time $t = 0$. Our algorithm can find a solution over time, that uses at most 8022.23 m^2 of area over the whole time interval. This solution is matched by a lower-bound computed by solving the static DC problem at this time step.

the peak total area is attained and attempt to compute an alternative feasible assignment at t' that, when combined with the current solution, produces an improved overall solution with a smaller maximum area. Furthermore, by formulating the DC as a integer program (IP) we obtain provable lower bounds for the KDC and can certify optimality for many instances; see our empirical evaluation in Section 3.

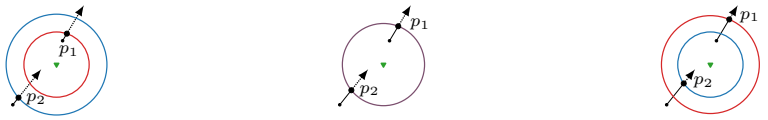
Related Work The DC [1] and its generalization the non-uniform minimum-cost multi-cover (MCMC) problem [12, 13] are known to be NP-hard. Recently, there has been some work on optimally multi-covering fixed points with disks of varying sizes [10]. Other work examines multi-covering points with the minimum number of unit disks, with centers at arbitrary locations [9]. Research on fundamental concepts for kinetic data structures [2] enabled algorithms for k-center and k-means problem for both kinetic stations and objects [3].

2 Algorithmic Approaches

In this section, we discuss how to extend a solution to the (stationary) DC at time t to a solution for the KDC. Further geometric insights allow us to improve extended solutions and design an iterative algorithm to solve the KDC problem. The proofs of the lemmas presented in this section are omitted due to space constraints and can be found in the full version [15].

Stationary Disk Cover We implement two strategies to solve the stationary Disk Cover problem (DC); (1) an IP formulation that can solve the DC problem to provable optimality, and (2) a nearest-neighbor heuristic.

Ensuring Feasibility Every solution to the (stationary) DC problem at time t can be extended to a solution for the KDC problem over time by maintaining the same assignment of objects to stations and altering the supporting point of each station as needed to ensure coverage. This yields a feasible solution for the KDC problem. Figure 2 shows, how initially the supporting point of the station y is p_2. At time $t \approx 0.41$ the disks defined by p_1 and p_2 have the same radius. After this point in time, p_1 becomes the supporting point of the disk.



(a) $t = 0.2$ **(b)** $t \approx 0.41$ **(c)** $t = 0.6$

Figure 2 The supporting point of station y changes from p_1 to p_2 to ensure coverage.

▶ **Lemma 1.** *In $\mathcal{O}(mn^2)$ time, all $\mathcal{O}(mn^2)$ possible supporting point changes when extending a stationary solution to $t \in [0, 1]$ can be computed.*

Our algorithm also handles degenerate cases where the supporting point might not change or multiple objects might have the same distance to a station, see Figure 3.

Figure 3 Degenerate cases after which the supporting point might change.

Improving Extended Solutions It is easy to see that only ensuring feasibility can lead to suboptimal solutions. To reduce the overall cost, we consider a second type of event that changes the assignment of objects to stations, see Figure 4a. Intuitively, we want to identify points in time in which it is cheaper to handover a point from one station to another.

(a) **(b)**

Figure 4 (a) Station y_2 takes over a point p_2. The objective values of the red and blue disks are equal at this point in time. We change the supporting point of both y_1 and y_2 at the same time. (b) Generalization of the case depicted on the left to more than two stations.

▶ **Lemma 2.** *In $\mathcal{O}(m^2n^3)$ time all $\mathcal{O}(m^2n^3)$ possible handovers of objects between two stations (when extending a stationary solution to $t \in [0, 1]$) can be computed.*

The concept of handing over objects between stations can be generalized to three or more stations, see Figure 4b. Furthermore, a chain of stations can successively take over objects from each other as time progresses. Figure 5 shows how initially the optimal solution may consist of blue disks covering all objects. At $t = 0.5$ the optimal solution switches to red disks, and the previously used blue disks are no longer part of the solution. This process can be extended to arbitrarily long chains of objects and stations. In the worst case, detecting

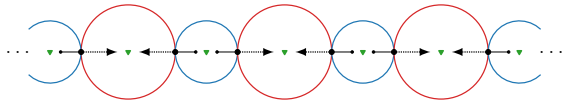


Figure 5 Multiple stations and objects in the same event point. Either the blue or the red disk cover the objects, but not both.

these types of events requires comparing every subset of stations with every other subset, resulting in $\Omega(2^m)$ possible events.

Iterative Algorithm We present an algorithm that uses the beforementioned geometric observations to compute heuristic and optimal solutions for the KDC. It starts by computing a stationary solution at $t = 0$ using any of the methods presented above. In the case of the IP formulation, the algorithm also provides a valid lower bound on the optimal solution which can be used to determine the quality of the solution during the solving process.

We then extend the stationary solution over time by applying the adjustments from Lemma 1, thereby obtaining a feasible KDC solution for the interval $[0, 1]$. To speed up the computation, we cache the next supporting point change for each center y and update the cache when assignments change. The algorithm then proceeds by identifying the point $t' \in [0, 1]$ where the current solution has maximum total area. This point must lie at the intersection of two intervals in which the assignment of objects to stations changes, since the objective value within each interval is an upward-opening parabola. At this point in time we compute a new stationary solution. If the stationary solution has a smaller objective value than the current solution, we can improve our current solution at time t' and potentially over the whole interval $[0, 1]$ or any set of subintervals. This is done by again extending the solution at time t' both in the positive and negative time direction. Afterwards we merge two solutions into a single valid solution, by taking the lower envelope of both functions. If the stationary solution has equal or higher objective value than the current solution, we terminate the procedure. Additionally, if the integer programming formulation is used, we can also terminate if the solution is within a desired optimality gap of the lower bound.

Improvements The first improvement is to the extension of solutions by considering handovers of objects between two stations, see Lemma 2. Again an efficient caching scheme can be used to speed up the computation. The second improvement targets solutions that are obtained while extending the solution. If the supporting point of a center y_i is also covered by another disk y_j, we remove it from y_i, reducing the radius of y_i and therefore the total area of the disks. We can repeat this process until no further improvement is possible. Finally, we reduce the time spent for extending and combining solutions. This can be done by simultaneously extending and combining the stationary solution (which is better than the current solution) only until it intersects the current solution, see Figure 6.

3 Evaluation

Experiments were carried out on desktop workstations equipped with AMD Ryzen 9 7900 (12×3.7 GHz) CPUs and 96 GB of RAM. Code and data are available online [14]. We use Gurobi [11] (version 12) solver for the integer program, with a default optimality gap of 0.01% and a time limit of 600 s. To detect numerical issues for degenerate cases, we added checks to ensure correctness of our implementation. For the instance generation, we used

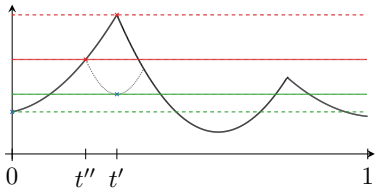


Figure 6 Iterative approach to improve the min max power consumption over time. Green and red lines denote lower and upper bounds on the optimal solution.

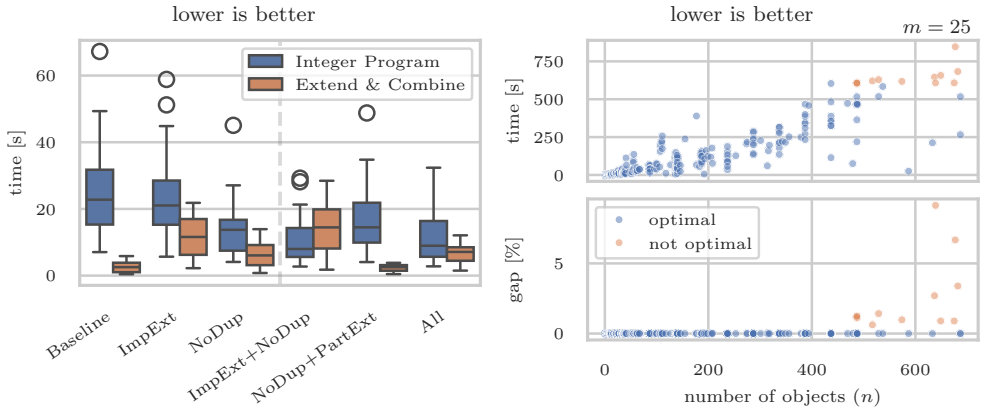
line segments with lengths uniformly distributed in $[25, 50]$ in a 100×100 grid. In real-world applications where objects are drones have velocities up to $100\,\mathrm{km\,h^{-1}}$, this resembles scenarios that take between 15 and 30 minutes to play out on a $100\,\mathrm{km}^2$ canvas. In total, we have the following instances sets.

`fix` We generate 25 instances with $m = 25$ and $n = 500$.

`fix_n` Fix number of objects to $n = 500$. For $m \in \{5, 10, \ldots, 50\}$ we generate 10 instances.

`fix_m` Fix number of sensors to $m = 25$. For $n \in \{50, 100, \ldots, 500\}$ we generate 10 instances.

`pub` Instances from well-known publicly available benchmarks used in [8], such as point sets from the CG:SHOP challenges [5, 6] and [7, 18, 19]. We scaled point sets to a fixed size and sampled $m = 25$ centers and point pairs (for trajectories) at random.



(a) Improvement Strategies **(b)** Public Instances

Figure 7 **(a)** Evaluation of different improvement strategies on the `fix` benchmark set. The results show that enabling all three solution strategies yields the best performance. **(b)** Performance of the exact version of `IP` on the `pub` dataset. Despite the variety and degeneracies, almost all instances can be solved to provable optimality.

Improvement Strategies We evaluate the influence of three proposed improvement strategies to the algorithm from Section 2. We denote the removal of duplicates as `NoDup`, the improved extension in Lemma 2 as `ImpExt` and partial extension as `PartExt`. Our baseline is the exact algorithm that uses the IP solver. We compare the time to compute the stationary IP solutions and the time to extend and combine solutions, see Figure 7a.

Our experiments show that both `ImpExt` and `NoDup` have a positive effect on the time spent in the IP solver and a negative effect on the time to extend and combine solutions. The use of `PartExt` however, reduces the time spent during extension and combination

significantly, which mitigates the negative effect resulting in the best configuration that achieves an overall runtime reduction of about 39 % compared to the baseline. We use the best configuration that enables all three improvement strategies for the following evaluations.

Influence of m and n We evaluate the influence of the number of objects n and the number of stations m on the performance of the algorithm. We execute the algorithm from Section 2 with both an IP solver and the nearest neighbor heuristic to solve the DC problem in each iteration. We denote the IP-based algorithm as `IP` and the nearest neighbor based algorithm as `NN`. Additionally, we implemented another heuristic that divides the time into $k = 10$ same sized intervals and computes the nearest neighbor solution at each border. The algorithm `FixedNN` then extends all solutions and reports the lower envelope. Figure 8 shows the performance of the different algorithms. We note that one instance of `IP` on the `fix_n` benchmark was restarted with a different seed due to numerical instability in the non-exact extension step that caused our checks to fail.

Overall, `IP` is able to solve all instances to optimality in a matter of seconds, demonstrating real-time capability for scenarios that take minutes to play out in the real world.

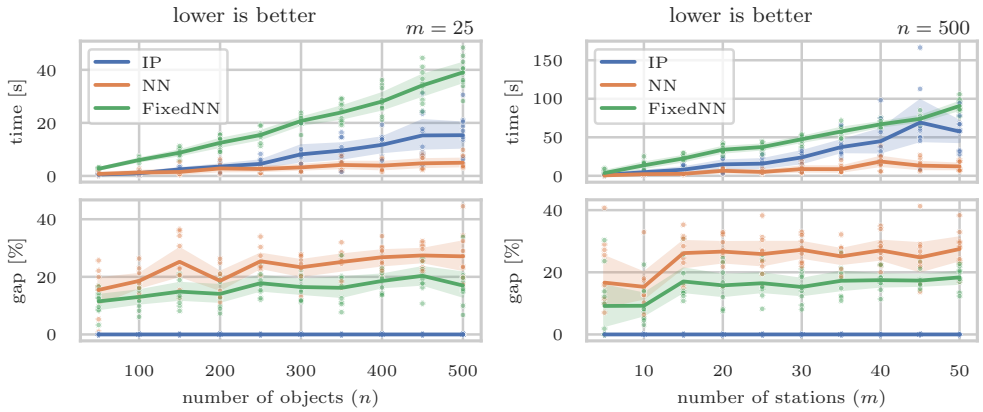


Figure 8 Performance of `IP`, `NN` and `FixedNN` on instances from the `fix_m` and `fix_n` datasets. While `IP` is significantly slower than `NN`, it produces optimal solutions for all instances. `FixedNN` is slightly better in terms of solution quality than `NN` but significantly slower.

Instances from Literature We further assess the algorithm's performance on the `pub` instance set, as shown in Figure 7b and illustrated by Figure 9. The presence of degeneracies, originating from real-world data, led to various numerical challenges in our implementation. To address these issues, we developed an alternative version of `IP` utilizing exact number types for computations. Our results show that 290 out of 302 instances with up to 700 objects could be solved to provable optimality within 600 s of computation time.

4 Conclusions

We provide novel insights into tracking large sets of moving objects from a set of fixed observation stations with optimal energy. While this problem is hard in theory, our practical methods can compute provably optimal coverage in a matter of seconds, demonstrating perspectives for real-world capability. There are many promising directions for future work;

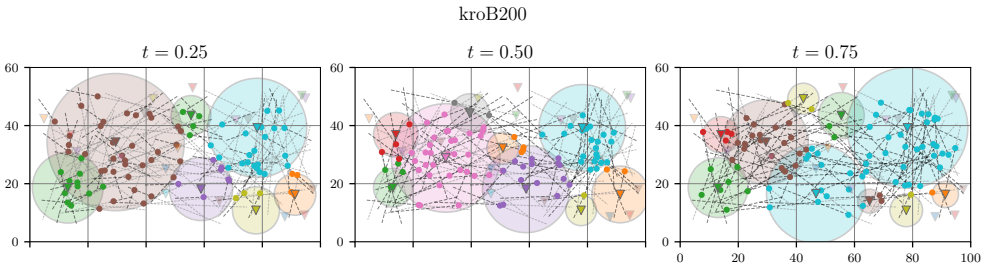


Figure 9 A solution of an instance from `pub` instance set based on *kroB200* TSPLIB [18].

deployment requires addressing practical issues, such as communication delays, online trajectory updates, and three-dimensional coverage. If the stations can be repositioned, the problem becomes even more complex, but also opens up new possibilities for optimization that may further reduce energy consumption. Uncertainty in trajectories and alternative coverage models, such as probabilistic sensing, also present interesting avenues for further research that could enhance the robustness and applicability of our methods in real-world scenarios.

References

1 Helmut Alt, Esther M. Arkin, Hervé Brönnimann, Jeff Erickson, Sándor P. Fekete, Christian Knauer, Jonathan Lenchner, Joseph S. B. Mitchell, and Kim Whittlesey. Minimum-cost coverage of point sets by disks. In *Symposium on Computational Geometry (SoCG)*, pages 449–458, 2006. doi:10.1145/1137856.1137922.

2 Julien Basch, Leonidas J. Guibas, and John Hershberger. Data structures for mobile data. *J. Algorithms*, 31(1):1–28, 1999. doi:10.1006/JAGM.1998.0988.

3 Sergei Bespamyatnikh, Binay K. Bhattacharya, David G. Kirkpatrick, and Michael Segal. Mobile facility location. In *International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications (DIAL-M)*, pages 46–53, 2000. doi:10.1145/345848.345858.

4 Adam P. Cohen, Susan A. Shaheen, and Emily M. Farrar. Urban air mobility: History, ecosystem, market potential, and challenges. *IEEE Transactions on Intelligent Transportation Systems*, 22(9):6074–6087, 2021. doi:10.1109/TITS.2021.3082767.

5 Erik D Demaine, Sándor P Fekete, Phillip Keldenich, Dominik Krupke, and Joseph S. B. Mitchell. Computing convex partitions for point sets in the plane: The CG:SHOP Challenge 2020. *arXiv preprint arXiv:2004.04207*, 2020.

6 Erik D Demaine, Sándor P. Fekete, Phillip Keldenich, Dominik Krupke, and Joseph S. B. Mitchell. Area-optimal simple polygonalizations: The CG Challenge 2019. *Journal of Experimental Algorithmics (JEA)*, 27(2):1–12, 2022.

7 Günther Eder, Martin Held, Steinþór Jasonarson, Philipp Mayer, and Peter Palfrader. Salzburg database of polygonal data: Polygons and their generators. *Data in Brief*, 31:105984, 2020. doi:10.1016/j.dib.2020.105984.

8 Sándor P. Fekete, Phillip Keldenich, and Michael Perk. Exact Algorithms for Minimum Dilation Triangulation. In *Symposium on Computational Geometry (SoCG)*, pages 48:1–48:18, 2025. doi:10.4230/LIPIcs.SoCG.2025.48.

9 Xuening Gao, Longkun Guo, and Kewen Liao. Fast approximation algorithms for multiple coverage with unit disks. In *Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, pages 185–193, 2022.

10 Mariem Guitouni, Chek-Manh Loi, Sándor P. Fekete, Michael Perk, and Aaron T. Becker. Multi-covering a point set by m disks with minimum total area. In *ICRA*, pages 3000–3006. IEEE, 2025. doi:10.1109/ICRA55743.2025.11127835.

11 Gurobi Optimization, LLC. Gurobi Optimizer Reference Manual, 2025. URL: https://www.gurobi.com.

12 Ziyun Huang, Qilong Feng, Jianxin Wang, and Jinhui Xu. PTAS for minimum cost multi-covering with disks. In *Symposium on Discrete Algorithms (SODA)*, pages 840–859, 2021.

13 Ziyun Huang, Qilong Feng, Jianxin Wang, and Jinhui Xu. PTAS for minimum cost multi-covering with disks. *SIAM Journal on Computing*, 53(4):1181–1215, 2024.

14 Chek-Manh Loi and Michael Perk. Drone air traffic control: Tracking a set of moving objects with minimal power, 2025. doi:10.5281/zenodo.17119781.

15 Chek-Manh Loi, Michael Perk, Malte Hoffmann, and Sándor P. Fekete. Drone air traffic control: Tracking a set of moving objects with minimal power. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2026. To appear.

16 Chek-Manh Loi, Michael Perk, Malte Hoffmann, and Sándor Fekete. Drone air traffic control: Tracking a set of moving objects with minimal power, 2026. doi:10.48550/arXiv.2603.05286.

17 Franco Mazzenga, Romeo Giuliano, and Alessandro Vizzarri. 5G-based synchronous network for air traffic monitoring in urban air mobility. *IEEE Access*, 12:188542–188559, 2024. doi:10.1109/ACCESS.2024.3513212.

18 G. Reinelt. TSPLIB–A Traveling Salesman Problem Library. *ORSA Journal of Computing*, 3(4):376–384, 1991.

19 Andre Rohe. VLSI data set. URL: https://www.math.uwaterloo.ca/tsp/vlsi/index.html.

Line Segment Visibility in Simple Polygons: Exact, Robust, Scalable Computation and Applications*

Sándor P. Fekete[1,2], Prahlad Narasimhan Kasthurirangan[3], Phillip Keldenich[1], Fabian Kollhoff[1], Chek-Manh Loi[1], and Michael Perk[1]

1 **Department of Computer Science, TU Braunschweig**
`s.fekete@tu-bs.de, {keldenich,kollhoff,loi,perk}@ibr.cs.tu-bs.de`
2 **L3S, Germany**
3 **Department of Applied Mathematics and Statistics, Stony Brook University**
`prahladnarasim.kasthurirangan@stonybrook.edu`

Abstract

The weak visibility polygon of a line segment s inside a simple polygon P, denoted by $V_P(s)$, is the region of the polygon that is visible from at least one point on s. Given its fundamental nature in computational geometry, several algorithms have been proposed to compute weak visibility polygons efficiently. In this work, we present an implementation of an optimal linear-time algorithm for computing the weak visibility polygon of a segment inside a triangulated simple polygon. Our implementation provides exact, robust geometric primitives and optimizations to handle large inputs with more than 18 000 000 vertices. We demonstrate two concrete applications: (1) construction of window partitions, a standard data structure in visibility algorithms, and (2) support for optimal minimum-link path queries between two points in a simple polygon, the latter serving as a direct use case of the former. Experimental results confirm that the end-to-end runtime scales linearly with the size of the polygon and is dominated by the cost of computing the triangulation, validating the practicality and scalability of the approach.

Related Version *Full version and code* [11, 12]

1 Introduction

In this paper, we provide a practical solution for a fundamental problem of Computational Geometry: Compute the *weak visibility polygon of a segment* ℓ within a simple polygon P. This task occurs in a wide variety of problems, such as in robot navigation or optimal surveillance, e.g., in the classical Watchman Problem; see the surveys by Mitchell [22, 21]. In theory, the weak visibility polygon of a segment can be computed in linear time, combining Chazelle's $\mathcal{O}(n)$ triangulation algorithm with an $\mathcal{O}(n)$ algorithm by Guibas, Hershberger, Leven, Sharir and Tarjan [15]. However, to this date, we are unaware of *any* open source implementation. Our implementation has the following properties.

- It is *scalable*: We can solve instances with 18 000 000 points within seconds.
- It is *robust* and *exact*: We use exact arithmetic for all computations.
- It is also *available*: It comes as a CGAL package that can be used out of the box.

In addition, we showcase two major use cases of our code for large-scale instances: We show how to compute a *window partition* of a polygon P, as well as computing the *minimum link* path between two points in P, for polygons with up to 2 000 000 vertices.

* Supported by the German Research Foundation (Deutsche Forschungsgemeinschaft, DFG) as part of project *Computational Geometry: Solving Hard Optimization Problems* (CG:SHOP) - 444569951. Work by PNK was carried out during a research stay at TU Braunschweig.

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

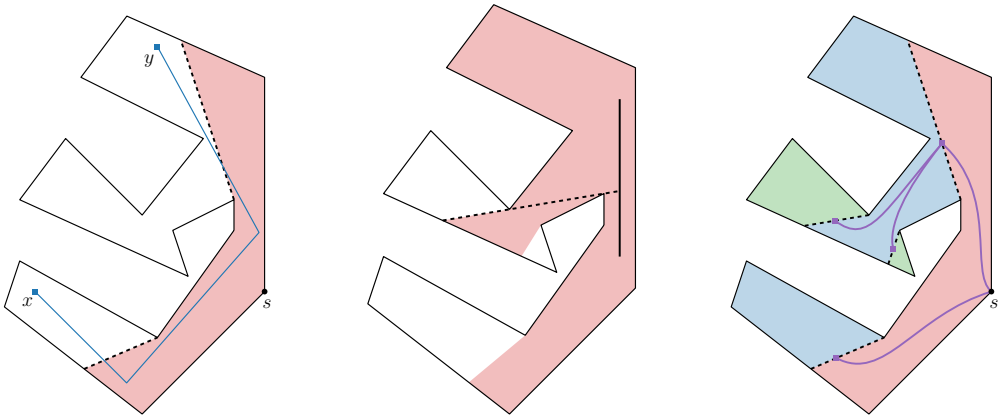


Figure 1 *Left:* $V_P(s)$ is given in red and the windows are marked using dashed lines. A minimum link path between x and y is marked in blue; $d_L(x, y) = 3$. *Middle:* A segment $\ell \in P$ is marked with a solid black line and $V_P(\ell)$ is marked in red. *Right:* The window partition $\mathcal{W}(s)$ and the window tree T_s of s. Points with link distance one are colored red, with two are colored blue, and with green are colored green. T_s is marked in purple.

Related Work Due to restricted space, we focus on weak visibility polygons of segments inside simple polygons; the interested reader can find a more general overview of visibility problems in [27, 23, 24, 13, 9]. Given a triangulation of a polygon, Guibas et al. [15] presented an $\mathcal{O}(n)$ algorithm that computes the visibility polygon. A number of algorithms have been proposed for data structures that can efficiently compute these for a query segment ℓ, each with its own trade-offs in terms of preprocessing time, query time, and space complexity [1, 3, 5, 8, 15]. Implementing these more recent results with improved output sensitive query time with exact predicates seems extremely challenging. It is also unclear if these will lead to improvements in runtime in practice (see Section 3 for experiments).

Computing segment visibility is a crucial subroutine in several geometric data structures; perhaps the most well-known one being window partitions [25]. These can be used to efficiently compute minimum link paths between two points. For a comprehensive overview of minimum link paths, we refer the reader to [18, Chapter 12] and [19].

2 Overview

We implemented the algorithm by Guibas et al. [15] as well as the window partition construction by Suri [25] which allows exact and approximate minimum link path computation. All our implementations use exact number types and handle real-world degenerate cases.

Weak Visibility of a Segment Let P be a simple polygon with n vertices. Two points x and y are *visible* to each other if the line segment $xy \subset P$. The *visibility polygon of a point* $s \in P$, denoted by $V_P(s)$, is the set of all points in P visible to s (see Figure 1, *Left*). Given a segment $\ell = pq$ inside of P, the *weak visibility polygon* $V_P(\ell)$ is the set of all points in P that are visible from any point on ℓ, i.e., $V_P(\ell) \coloneqq \{p \in P \mid \exists s \in \ell : p \in V_P(s)\}$, see Figure 1 *Middle*. Constructing a triangulation $\mathcal{T}$ of a polygon P takes linear time using Chazelle's algorithm [7]. Thus, computing $V_P(\ell)$ for a segment ℓ in P takes linear time – first triangulate P [7], then use the algorithm from Guibas et al. [15], which we describe in the full version [11]. As we are unaware of any implementation of [7], we compute a

constrained Delaunay triangulation (CDT) of P using CGAL [2]. Due to the lack of more direct polygon triangulations or other types of constrained triangulations implemented in CGAL, this is, to the best of our knowledge, the recommended way of obtaining a polygon triangulation using CGAL. Although CDT can incur a worst-case cost of $\mathcal{O}(n^2)$ [4], they perform very well in practice. Our implementation of [15] nevertheless runs in linear time on the computed triangulation. For this runtime, we also implemented a specialized data structure called a *finger tree* [17, 16] (for details see the full version [11]). A finger tree is a balanced search tree augmented with pointers (*fingers*) to its extremal elements, enabling access and splits in $\mathcal{O}(\log \delta)$ time, where δ is the distance to the nearest finger. This yields an amortized constant time per operation in the algorithm and thus overall linear runtime. A simpler alternative is to use a standard balanced binary search tree such as CGAL's `Multiset` (a red-black tree), which supports the same operations in $\mathcal{O}(\log n)$ time, resulting in an $\mathcal{O}(n \log n)$ algorithm. To the best of our knowledge, all previous implementations of finger trees are in functional programming languages like Haskell or Scala [16, 14]; we used C++ for their iterative versions.

Minimum Link Paths and Window Partitions For two points $x, y \in P$, the *link distance* $d_L(x, y)$ between them is the minimum number of edges in any polygonal path within P between x and y (see Figure 1, *Left*). Consider a point $s \in P$ and its visibility polygon $V_P(s)$. The vertices of $V_P(s)$ are either vertices of P or the shadows of reflex vertices of P seen by s. The edges of $V_P(s)$ between reflex vertices and their shadows are called *windows*. A window w is a chord of P and divides P into two parts; define $P(w, s)$ to be the part of P that does not contain s. The *window partition* $\mathcal{W}_P(s)$ is the partition of P obtained by recursively computing the visibility polygons of the windows of $V_P(s)$ within their respective subpolygons $P(w, s)$. The *window tree* T_s of $\mathcal{W}(s)$ is rooted at s and has the windows as its vertices, with an edge between two vertices whenever their regions share a common boundary; see Figure 1, *Right*. Following Suri [25], we use $\mathcal{W}(\cdot)$ to compute exact and approximate minimum link paths; see the full version [11] for details. The approximation algorithm precomputes window partitions from k random seed points and answers a query by finding the best path among all seed trees, guaranteeing an additive error of at most four links. We improved the visibility polygon routine so it runs in time linear in the number of triangles of $\mathcal{T}$ that intersect $V_P(\ell)$, allowing the window-partition construction to run in overall linear time.

3 Experimental Evaluation

In this section, we present the experimental evaluation of our implementations, see Figure 2 for some example outputs. All experiments were run on otherwise idle workstations equipped with AMD Ryzen 7 5800X CPUs and 128 GiB of DDR4-3200 memory, running Ubuntu Linux 24.04.3 LTS. The core routines were implemented in C++17, compiled against CGAL 6.0.1 and Boost 1.83.0 using g++ 13.3.0. Our instances and code are publicly available [12].

3.1 Weak Line Segment Visibility

For evaluating our implementation, we consider the following research questions.

RQ1 How does our implementation scale to large inputs, and how is the total runtime distributed onto the individual subroutines?

RQ2 Is there a measurable difference between using finger trees and CGAL's multiset? Which of the two approaches is faster?

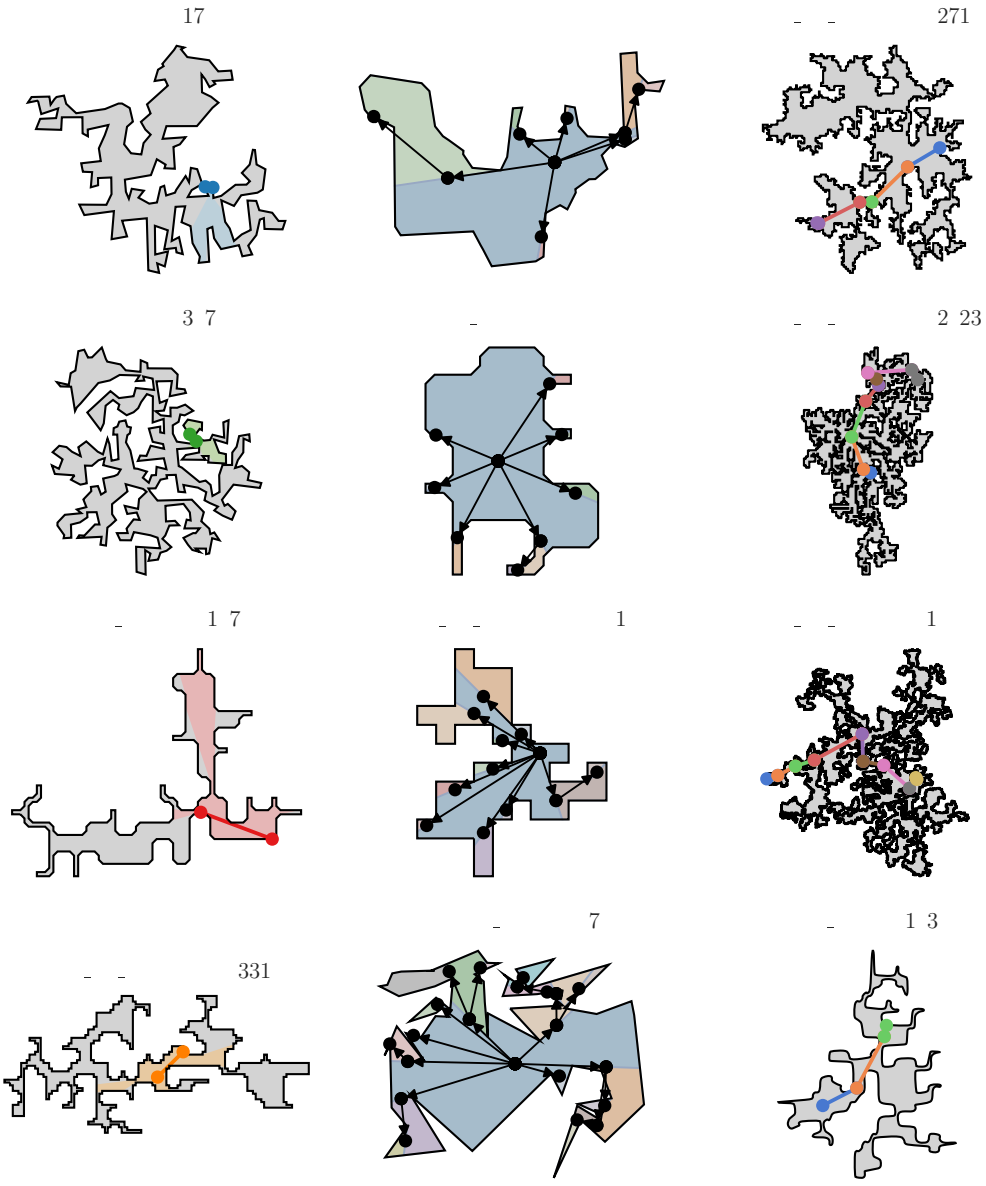


Figure 2 Example outputs. *Left Column:* Weak visibility polygon of a segment. *Middle:* Window partition from a source point s. *Right:* Minimum link path between two points s and t.

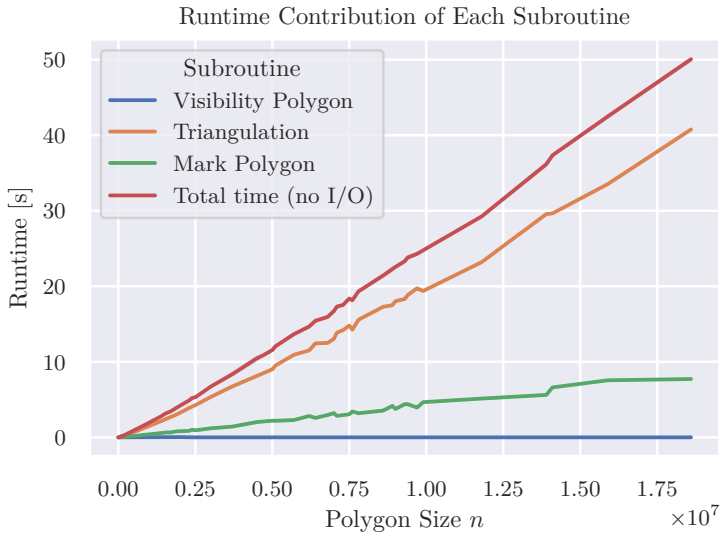


Figure 3 The runtime of different subroutines (triangulating the polygon, marking faces in the polygon, and computing the visibility polygon itself).

RQ3 How does the performance of our implementation depend on the size of the output?

To address these questions, we randomly generate segments for simple polygons from the Salzburg Database of Geometric Inputs[1] [10], the largest of which has 18 556 888 vertices. To generate segments, we sample a source point uniformly at random within the given polygon, compute the visibility polygon of that point, and then either sample a target point in the visibility polygon or pick a random vertex of it. Using each polygon and each approach twice results in a total of 24 096 instances. For each such instance, and each of the two shortest path tree data structures, we compute the exact visibility polygon five times, measuring the runtime (both total and of various subroutines). Figure 3 depicts the resulting runtimes. We can see that the overall runtime is dominated by computing a triangulation of the input polygon, which we do via CGAL's constrained Delaunay triangulation (CDT). Even CGAL's `mark_in_domain` routine, which simply flags the CDT's triangles that constitute the polygon, is typically more expensive than the actual visibility computation. The runtime for the visibility computation seems to be almost unaffected by the size of the input polygon because it mostly depends on the size of the output polygon; see Figure 4, *Left*. On our inputs, there is a small but detectable runtime advantage resulting from using a finger tree compared to CGAL's `Multiset`; see Figure 4, *Right*. This is despite the fact that funnels are rarely large; when building a funnel by left insertion as part of a preliminary micro-benchmark, we found that our finger tree implementation already becomes significantly faster at around 15 funnel vertices. This threshold might be reached even earlier if the comparison operation is particularly expensive. In light of this observation, in the remaining experiments, we used our finger tree-based funnel.

[1] We used all valid simple polygons not labeled as "contrived".

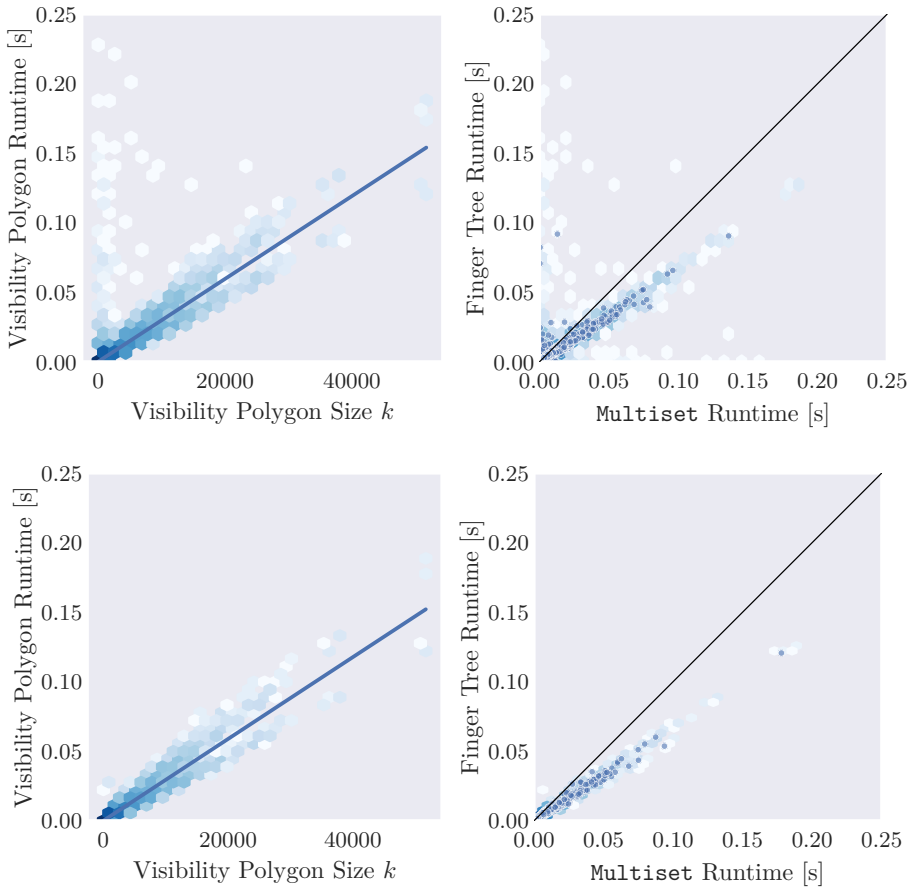


Figure 4 *Top Left:* Runtime dependence on the size of the output visibility polygon, excluding preprocessing. The hexagonal bin plot shows the underlying distribution similar to a scatter plot, for which we have too many data points; darker hues correspond to a higher concentration of data points, the gray area contains none. *Top Right:* Plot of the runtime using CGAL's `Multiset` and our finger tree implementation with a random sample of data points; the finger tree is better for points below the black diagonal. *Bottom:* The noise in the plot can be attributed to random disturbances such as interrupts, page faults on code pages hitting the network file system, and similar issues; it completely disappears if we drop the worst two out of five runs for each (instance, method) pair.

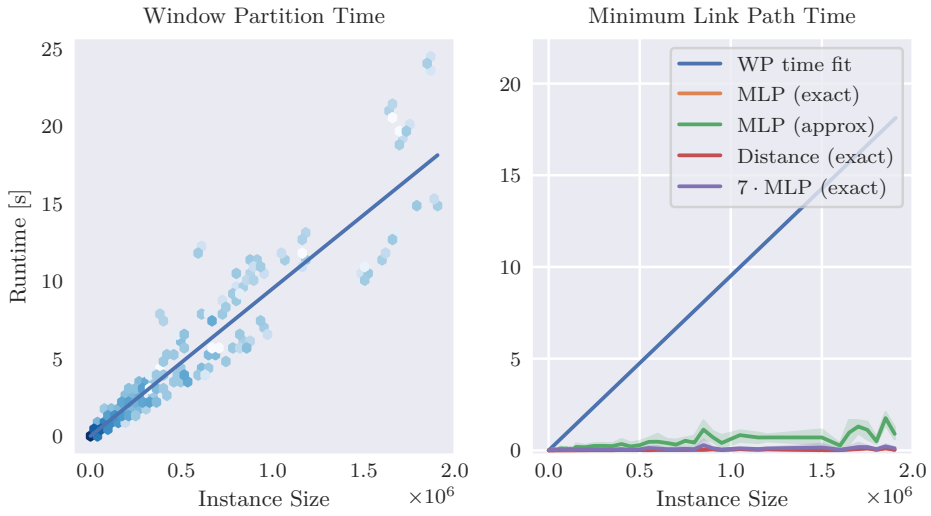


Figure 5 *Left:* The runtime distribution of our window partition implementation, which overall appears to scale linearly, but does have a significant amount of outliers; these may be related to the cost of exact computations arising more commonly in polygons with many collinear vertices. *Right:* Comparison of the individual components, showing only the regression line for the window partition time to improve readability. It can be seen that, once a window partition is computed, computing the link distance is very cheap. A single path reconstruction has almost negligible cost, whereas $k = 7$ reconstructions for the approximate approach are more noticeable, but still clearly dominated by the window partition time.

3.2 Minimum Link Paths

Computing a window partition is a basic routine that is useful in different contexts. Furthermore, we only need one window partition from a source point s to compute shortest link-distance paths to any target t, and only a constant number to approximate shortest link-distance paths for any source-target pair inside our polygon. We therefore study the performance of our window partition on its own, as well as in the context of minimum link paths, addressing the following research questions.

RQ4 How does our window partition implementation scale to large polygons?

RQ5 How does it compare to the other parts of our minimum link path implementation?

RQ6 How does the approximate algorithm perform w.r.t. runtime and solution quality?

To answer these questions, we randomly selected a sample of 1000 simple polygons with at most 2 000 000 vertices from [10]. For each such polygon P, we generate 10 source points $s \in P$ uniformly at random and computed a window partition for each source. Furthermore, for each source point s, we generate a random target point $t \in P$ outside of the visibility polygon of s and compute an exact minimum link s-t-path, as well as an approximate minimum link path using $k = 7$ randomly chosen source points for the window partitions in our approximation. The resulting runtimes are depicted in Figure 5. For each path computed using the approximation algorithm, Figure 6 shows the absolute additive error depending on the number of window partitions k; the quality of the approximation notably improves with higher k. One should, however, note that increasing k makes both preprocessing and path reconstruction more expensive, both scaling linearly in k.

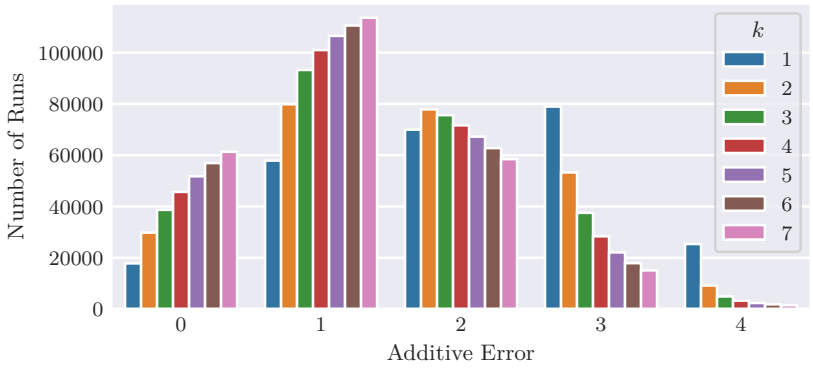


Figure 6 The additive approximation error of our approximate minimum link paths, depending on k, the number of source points used to compute window partitions. In particular, for $k \geq 4$, the majority of queries result in an error of 0 or 1.

Overall, we find that both our window partition and minimum link path implementation scale well to large polygons; in fact, in our experiments, time spent inputting, verifying and outputting polygons was typically orders of magnitude higher than pure computation time. If we want to query minimum link paths from a single source s, each subsequent query helps amortizing the cost of computing a window partition; similarly, for doing many queries for different sources and targets in the same polygon, the approximate algorithm can be used to amortize the cost of computing a constant number of window partitions. Depending on the relative importance of runtime versus solution quality, the approximation may be preferable.

4 Conclusion

We presented a scalable, robust and exact implementation for computing the weak visibility polygon of a line segment in a simple polygon, which can handle large inputs with more than 18 000 000 vertices on commodity hardware. The bottleneck of the implementation is the computation of a triangulation of the input polygon. We also demonstrated two concrete applications: computing the window partition, and computing the link distance between two points inside a simple polygon. Our experiments confirm that the visibility and link-path computations themselves are efficient and output-sensitive, while the approximate minimum link path algorithm already achieves an additive error of at most 1 for the majority of queries with $k \geq 4$ random source points. For further improvement on the algorithm engineering side, we could parallelize the shortest path tree and window partition implementations.

An immediate generalization is to non-simple polygons. Suri and O'Rourke [26] studied algorithms for computing visibility polygons with holes. Following the approach described in [6], we can extend our implementation to handle polygons with holes by first splitting the polygon into simple subpolygons and then computing the union of these visibility polygons. The same can be considered for link distance in polygons with holes [20, 19].

References

1 Boris Aronov, Leonidas J. Guibas, Marek Teichmann, and Li Zhang. Visibility queries and maintenance in simple polygons. *Discrete and Computational Geometry*, 27(4):461–483, 2002. doi:10.1007/S00454-001-0089-9.

2 Jean-Daniel Boissonnat, Olivier Devillers, Monique Teillaud, and Mariette Yvinec. Triangulations in CGAL. In *Symposium on Compuational Geometry (SoCG)*, page 11–18, 2000. doi:10.1145/336154.336165.

3 Prosenjit Bose, Anna Lubiw, and J. Ian Munro. Efficient visibility queries in simple polygons. *Computational Geometry*, 23(3):313–335, 2002. doi:10.1016/S0925-7721(01)00070-0.

4 Francisc Bungiu, Michael Hemmer, John Hershberger, Kan Huang, and Alexander Kröller. Efficient computation of visibility polygons. 2014. arXiv:1403.3905.

5 Mojtaba Nouri Bygi and Mohammad Ghodsi. Weak visibility queries in simple polygons. In *Canadian Conference on Computational Geometry (CCCG)*, 2011. URL: http://www.cccg.ca/proceedings/2011/papers/paper95.pdf.

6 Mojtaba Nouri Bygi and Mohammad Ghodsi. Weak visibility queries of line segments in simple polygons and polygonal domains. 2013. arXiv:1310.7197.

7 Bernard Chazelle. Triangulating a simple polygon in linear time. *Discrete and Computational Geometry*, 6:485–524, 1991. doi:10.1007/BF02574703.

8 Danny Z. Chen and Haitao Wang. Weak visibility queries of line segments in simple polygons. *Computational Geometry*, 48(6):443–452, 2015. doi:10.1016/j.comgeo.2015.02.001.

9 Mark de Berg, Otfried Cheong, Marc van Kreveld, and Mark Overmars. *Computational Geometry: Algorithms and Applications*. Springer, 2008. doi:10.1007/978-3-540-77974-2.

10 Günther Eder, Martin Held, Steinþór Jasonarson, Philipp Mayer, and Peter Palfrader. Salzburg database of polygonal data: Polygons and their generators. *Data in Brief*, 31:105984, 2020. doi:10.1016/j.dib.2020.105984.

11 Sándor P. Fekete, Prahlad N. Kasthurirangan, Phillip Keldenich, Fabian Kollhoff, Chek-Manh Loi, and Michael Perk. Line segment visibility in simple polygons: Exact, robust, scalable computation and applications. In *Symposium on Compuational Geometry (SoCG)*, 2026. to appear.

12 Sándor P. Fekete, Prahlad N. Kasthurirangan, Phillip Keldenich, Fabian Kolhoff, Chek-Manh Loi, and Michael Perk. Line segment visibility in simple polygons, 2025. doi:10.5281/zenodo.17800685.

13 Submir Kumar Ghosh. *Visibility Algorithms in the Plane*. Cambridge University Press, 2007.

14 Andrew Gibiansky. Finger trees, 2014. URL: https://andrew.gibiansky.com/blog/haskell/finger-trees/.

15 Leonidas J. Guibas, John Hershberger, Daniel Leven, Micha Sharir, and Robert Endre Tarjan. Linear-time algorithms for visibility and shortest path problems inside triangulated simple polygons. *Algorithmica*, 2:209–233, 1987. doi:10.1007/BF01840360.

16 Ralf Hinze and Ross Paterson. Finger trees: a simple general-purpose data structure. *Journal of Functional Programming*, 16(2):197–217, 2006. doi:10.1017/S0956796805005769.

17 Scott Huddleston and Kurt Mehlhorn. A new data structure for representing sorted lists. *Acta informatica*, 17(2):157–184, 1982.

18 Anil Maheshwari, Jörg-Rüdiger Sack, and Hristo N. Djidjev. Link distance problems. In Jörg-Rüdiger Sack and Jorge Urrutia, editors, *Handbook of Computational Geometry*, pages 519–558. North Holland / Elsevier, 2000. doi:10.1016/B978-044482537-7/50013-9.

19 Joseph S. B. Mitchell, Valentin Polishchuk, and Mikko Sysikaski. Minimum-link paths revisited. *Computational Geometry*, 47(6):651–667, 2014. Special issue on EuroCG'11.

20 Joseph S. B. Mitchell, Günter Rote, and Gerhard J. Woeginger. Minimum-link paths among obstacles in the plane. In *Symposium on Computational Geometry (SoCG)*, pages 63–72. ACM, 1990. doi:10.1145/98524.98537.

21 Joseph SB Mitchell. Shortest paths and networks. In *Handbook of Discrete and Computational Geometry*, pages 445–466. CRC Publishing, 1997.

22 Joseph SB Mitchell et al. Geometric shortest paths and network optimization. In *Handbook of Computational Geometry*, volume 334, pages 633–702. North Holland, 2000.

23 Joseph O'Rourke. *Art gallery theorems and algorithms*. Oxford University Press, Inc., USA, 1987.

24 Jörg-Rüdiger Sack and Jorge Urrutia. *Handbook of computational geometry*. Elsevier, 1999.

25 Subhash Suri. On some link distance problems in a simple polygon. *IEEE Transactions on Robotics and Automation*, 6(1):108–113, 1990. doi:10.1109/70.88124.

26 Subhash Suri and Joseph O'Rourke. Worst-case optimal algorithms for constructing visibility polygons with holes. In Alok Aggarwal, editor, *Proceedings of the Second Annual ACM SIGACT/SIGGRAPH Symposium on Computational Geometry, Yorktown Heights, NY, USA, June 2-4, 1986*, pages 14–23. ACM, 1986. doi:10.1145/10515.10517.

27 Csaba D Toth, Joseph O'Rourke, and Jacob E Goodman. *Handbook of discrete and computational geometry*. CRC press, 2017.

Ordinal Geometric Thickness of Complete and Complete Bipartite Graphs*

Patrizio Angelini[1], Michael A. Bekos[2], Luca Grilli[3], and Aikaterini Maria Ntasiou[2]

1 **John Cabot University, Rome, Italy**
pangelini@johncabot.edu

2 **Department of Mathematics, University of Ioannina, Greece**
{bekos,m.ntasiou}@uoi.gr

3 **Department of Engineering, University of Perugia, Italy**
luca.grilli@unipg.it

1 Introduction

Thickness [16] is a combinatorial measure defined as the minimum number of planar subgraphs into which a given graph can be decomposed. Its exact value is known only for restricted graph classes, such as complete [2] and complete bipartite graphs [5]. Since determining the thickness of general graphs is NP-hard [12], most of the research has primarily focused on establishing bounds for this parameter; see [13] for a survey.

The geometric analogue of thickness is known as *geometric thickness* [6, 9] and is defined as the minimum integer k such that there exists a straight-line drawing of the given graph whose edge set can be partitioned into k planar layers. In other words, here it is additionally required all planar layers to arise from a single straight-line drawing (with fixed vertex positions). Clearly, the thickness of a graph is a lower bound for its geometric thickness; relationships with other graph parameters are studied in [7, 8, 10, 17].

In this work, we introduce and study a variant of the geometric thickness problem in which the edge lengths are strictly increasing from one layer to the next. This variant is motivated by zoom-based graph visualizations where at any given zoom level, only edges of certain lengths are visible, while shorter edges are hidden (to avoid cluttering the visualization) and longer ones are suppressed (as they fall outside the zoom lens); see Figure 1.

▶ **Definition 1.1.** The *ordinal geometric thickness* $\theta_o(G)$ of a graph $G = (V, E)$ is the smallest integer t such that there exist a straight-line drawing Γ of G and a partition of its edges into t layers $E_1, \ldots, E_t$, where each layer induces a planar drawing in Γ and for each pair of layers E_i and E_j with $1 \leq i < j \leq t$ the edges in E_i have length strictly smaller than those in E_j. If $\theta_o(G) = 2$, we call the edges in E_1 *short* and those in E_2 *long*.

Equivalently, the layers are strictly ordered by edge length; for every pair of edges e and f with $e \in E_i$, $f \in E_j$ and $i < j$, we have $\max_{e \in E_i} |e| < \min_{f \in E_j} |f|$, where $|.|$ denotes the length of the edge. Problems of realizing graphs whose edge lengths satisfy prescribed ordinal constraints (without imposing planarity) have been investigated in various contexts in the literature; see, e.g., [1, 3, 4, 14].

* M. A. Bekos and A. M. Ntasiou is supported by the HFRI grant 26320. This work started at the Summer Workshop on Graph Drawing 2025. We thank the organizers and participants for their helpful feedback and discussions.

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

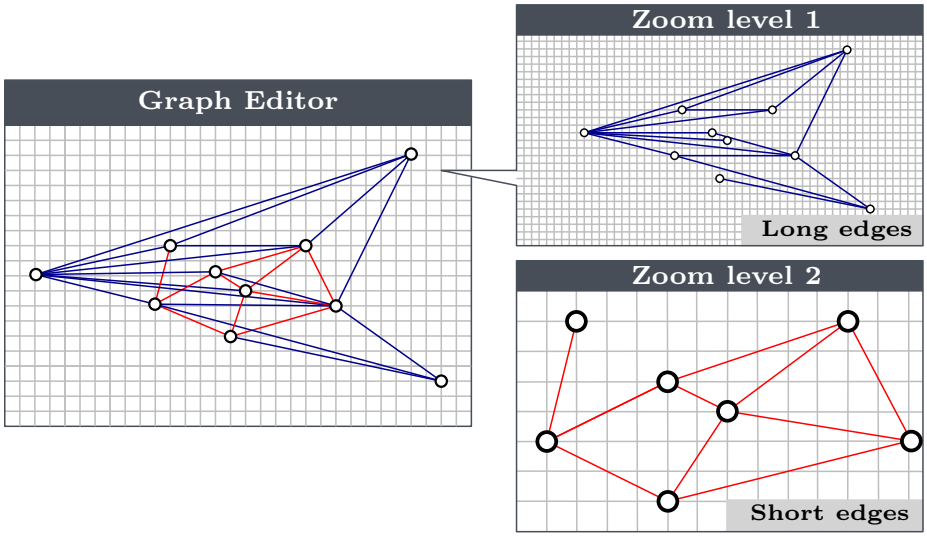


Figure 1 Visualizing a graph with ordinal geometric thickness 2 at different zoom levels.

2 Properties of Graphs with Ordinal Geometric Thickness 2

In this section, we study properties of a straight-line drawing Γ of a graph G with $\theta_o(G) = 2$, allowing the ordinal geometric thickness of such a graph to be treated combinatorially.

▶ Property 1. The crossing graph of Γ is bipartite, where the vertices of the crossing graph are the edges of Γ and two vertices are adjacent if and only if the corresponding edges cross.

Proof. Assume to the contrary that there is an odd cycle $\mathcal{C}$ in the crossing graph of Γ. Since Γ is a straight-line drawing with ordinal geometric thickness 2, it follows that any two neighboring vertices in the crossing graph of Γ correspond to two edges of Γ that cross such that one is short and the other is long. Since $\mathcal{C}$ is of odd length, it follows that there exist two edges of Γ that are either both short or both long and cross in Γ; a contradiction. ◀

▶ Property 2. Drawing Γ does not contain three mutually crossing edges.

Proof. Such edges induce a 3-cycle in the crossing graph of Γ, which violates Property 1. ◀

▶ Property 3. There is no K_5 in Γ whose vertices are in convex positions.

Proof. Assume to the contrary that Γ contains a K_5 whose vertices are in convex positions. Let $\mathcal{R}$ be the convex region of Γ formed by these vertices. The edges of the K_5 that lie in the interior of $\mathcal{R}$ induce a 5-cycle in the crossing graph of Γ, which violates Property 1. ◀

▶ Property 4. Assume that Γ contains a 3-cycle C consisting of two short edges and a long edge. Let u be the vertex of C that is incident to the two short edges. Then, every long edge incident to u lies entirely outside the interior of C in Γ.

Proof. Assume to the contrary that u is incident to a long edge, say (u, v), that has a part in the interior of C in Γ. Let (u, w) and (u, z) be the two short edges of C incident to u. Since (u, v) is longer than (u, w) and (u, z), (u, v) crosses (w, z) in Γ. This is a contradiction to the fact that Γ has ordinal geometric thickness 2, since (u, v) and (w, z) are long. ◀

Figure 2 Illustrations of the (a) B- and (b) W-configurations.

▶ Property 5. Assume that Γ contains a 3-cycle C consisting of three short edges. Then, there is no long edge in the closed triangular region of Γ defined by C.

Proof. This is because an edge in the interior of C is shorter than the longest edge of C. ◀

▶ Property 6. There is no K_4 in Γ composed of exactly two long edges that are independent.

Proof. Assume to the contrary that there is a K_4 with exactly two long edges e and e' that are independent. Since Γ is geometric, two edges sharing a common vertex cannot cross in Γ.Since every short edge of the K_4 (i.e., a different one than e and e') shares a vertex with both e and e', the drawing of the K_4 in Γ is planar, such that, w.l.o.g., e is on the outer 3-cycle, while e' in its interior. This violates Property 4 of Γ. ◀

▶ Property 7. Let C_s and C_l be two cycles in Γ consisting only of short and long edges, respectively. Then, Γ does not contain two neighboring vertices u and v such that u is in the interior of both C_s and C_l, while v is in the exterior of both C_s and C_l.

Proof. Since u is in the interior of both C_s and C_l, while v is in the exterior of both C_s and C_l, it follows by Jordan curve theorem that the edge (u, v) crosses both a short edge (of C_s) and a long edge (of C_l), contradicting the fact that Γ has ordinal geometric thickness 2. ◀

▶ Property 8. Let C be a cycle whose boundary forms a convex region in Γ. Let x and (w, z) be a vertex and an edge lying in the interior of C in Γ. If x is connected to the endpoints of a boundary edge (u, v) of C by crossing twice the edge (w, z), then x, u, v, w and z are in convex position and thus they cannot induce a K_5.

Proof. To prove the statement, it suffices to show that all interior angles of the region formed by x, u, v, w and z are strictly less than 180°. This is obvious for the angles at u and v, because u and v are corners of C in Γ, which is convex. Since the edges (x, u) and (x, v) cross (w, z), the angle at x is internal to the triangular region formed by x, w and z and the angles at w and z are internal to the triangular regions formed by x, u, w and x, v, z (or vice versa). Hence, these three angles are all strictly less than 180°. ◀

Finally, there exist two well-known 1-plane graphs (i.e., graphs embedded in the plane with at most one crossing per edge) known in the literature as B- and W-configurations [15] admitting no straight-line drawing [11, 15] (see Figure 2). Hence, the next property follows.

▶ Property 9. Drawing Γ contains neither a B- nor a W-configuration.

3 Ordinal Geometric Thickness: Complete & Complete Bipartite Graphs

In this section, we first investigate the ordinal geometric thickness of the complete n-vertex graph K_n. For small values of n, we establish tight bounds. Subsequently, we demonstrate an algorithm that yields straight-line drawings of K_n with certain ordinal geometric thickness.

▶ **Lemma 3.1.** *The ordinal geometric thickness of K_8 minus an edge is 2.*

Proof. Since $K_8 - e$ is not planar, its (ordinal) geometric thickness cannot be 1. Figure 3, on the other hand, demonstrates a drawing of $K_8 - e$ in which the red and the blue edges form two planar subdrawings, such that the length of each red edge is strictly smaller than that of each blue edge. This establishes that $\theta_o(K_8 - e) = 2$, as desired. ◀

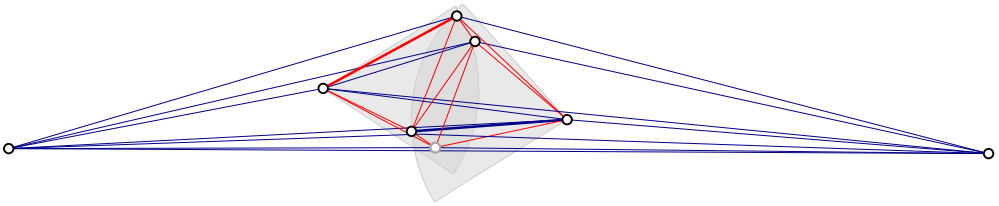


Figure 3 A straight-line drawing of $K_8 - e$ and a partition of its edges into two layers (blue and red) each inducing a planar subdrawing such that the longest red edge is shorter than the shortest blue edge (both bold). The missing edge connects the topmost vertex with the gray one.

▶ **Corollary 3.2.** *The ordinal geometric thickness of K_5, K_6 and K_7 is 2.*

We prove that the ordinal geometric thickness of K_8 is 3. Since K_8 is of geometric thickness 2 [6], this establishes that ordinal geometric thickness can be strictly greater than geometric thickness. By Lemma 3.1, K_8 is a minimal separating example for the two parameters.

▶ **Lemma 3.3.** *The ordinal geometric thickness of K_8 is 3.*

Proof sketch. Assume to the contrary that $\theta_o(K_8) = 2$, i.e, K_8 admits a straight-line drawing Γ and a partition of its edges into two layers, where each layer induces a planar subdrawing in Γ and the edges in the first layer (short edges) are all shorter than those in the second layer (long edges). We say that the vertices incident to the outer face of Γ are at *level 0*, while vertices incident to the outer face of the subdrawing of Γ obtained by deleting all vertices of levels $0, \ldots, i-1$ are at *level i*. Since Γ is a drawing of K_8, the boundary of each level is convex. Hence, each level of Γ, except the last one, has at least three vertices. By Property 3, each level of Γ, including the last, cannot have more than four vertices. We denote by $\langle i, j, k\rangle$ with $3 \leq i, j \leq 4$ and $i + j + k = 8$ the case in which levels zero, one and two of Γ have i, j and k vertices, respectively. It follows that in order to prove the lemma, it suffices to consider four cases in total, namely $\langle 4, 3, 1\rangle$, $\langle 3, 3, 2\rangle$, $\langle 3, 4, 1\rangle$ and $\langle 4, 4, 0\rangle$.

Case $\langle 4, 3, 1\rangle$: The sole level-two vertex u lies in the interior of triangle T formed by the three level-one vertices. Consider the edges of u towards the four level-zero vertices of Γ. Each of these edges crosses an edge of T; see Figure 4a. By the pigeonhole principle, there is an edge of T crossed by two edges emanating from u, which contradicts Property 8 of Γ.

Case $\langle 3, 3, 2\rangle$: Let $\{u_1, u_2, u_3\}$, $\{u_4, u_5, u_6\}$ and $\{u_7, u_8\}$ be the vertices of levels zero, one and two, respectively. Since Γ satisfies Property 8, the three edges connecting each of u_7 and

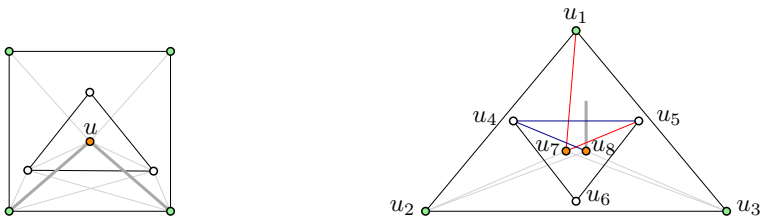


Figure 4 Illustration for Cases (a) $\langle 4, 3, 1\rangle$ and (b) $\langle 3, 3, 2\rangle$. Vertices of level zero, one and two are green-, white- and orange-filled, respectively; edges colored red are short, while blue are long.

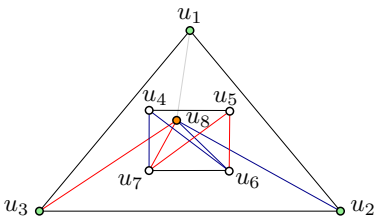


Figure 5 Illustration for Case $\langle 3, 4, 1\rangle$; edges colored red are short, while blue are long.

u_8 to the level-zero vertices must each cross a different edge of the triangle formed by the three level-one vertices. Also, since each of u_7 and u_8 has three edges towards the three level-one vertices, two out of these six edges, say w.l.o.g. (u_4, u_8) and (u_5, u_7), necessarily cross. Assume w.l.o.g. that (u_4, u_8) is long, while (u_5, u_7) is short; see Figure 4b. Up to renaming, let (u_1, u_7) be the edge incident to u_7 that crosses (u_4, u_5). This edge is short, as it crosses the long edge (u_4, u_8). This in turn implies that (u_4, u_5) is long. Now consider the edge that connects u_8 with one of the vertices of level zero by crossing the edge (u_4, u_5). This edge crosses both the short edge (u_5, u_7) and the long edge (u_4, u_5), which is a contradiction.

Case $\langle 3, 4, 1\rangle$: Let $\{u_1, u_2, u_3\}$ and $\{u_4, u_5, u_6, u_7\}$ be the vertices of levels zero and one, respectively; see Figure 5. The sole level-two vertex is u_8 and lies in the interior of the convex quadrilateral region Q formed by the level-one vertices. Since Q is convex, its two diagonals (u_4, u_6) and (u_5, u_7) necessarily cross in its interior. Assume w.l.o.g. that u_8 lies in the triangular region defined by vertices u_4, u_5 and the crossing point of the two diagonals of Q. Up to renaming, assume that (u_4, u_6) is long, whereas (u_5, u_7) is short. Since (u_7, u_8) crosses (u_4, u_6), which is long, it follows that (u_7, u_8) is short. Symmetrically, (u_6, u_8) is long. Since Γ satisfies Property 8, the three edges incident to u_8 towards the level-zero vertices cross distinct edges of Q. However, (u_6, u_7) cannot be crossed by an edge incident to u_8 towards the level-zero vertices, as otherwise this edge together with (u_4, u_6) and (u_5, u_7) would form three mutually crossing edges in Γ, violating Property 2. So, we can again assume w.l.o.g. that (u_1, u_8) crosses (u_4, u_5), (u_2, u_8) crosses (u_5, u_6) and (u_3, u_8) crosses (u_4, u_7). Since (u_2, u_8) crosses (u_5, u_6), it also crosses (u_5, u_7), which is a short edge. So, (u_2, u_8) is long, while (u_5, u_6) is short. Symmetrically, (u_3, u_8) is short, while (u_4, u_7) is long. The proof is then completed by bringing to a contradiction the two main subcases: **(a)** (u_1, u_7) intersects an edge of Q, **(b)** (u_1, u_7) does not intersect any of the edges of Q.

Case $\langle 4, 4, 0\rangle$: Let $\{u_1, u_2, u_3, u_4\}$ and $\{u_5, u_6, u_7, u_8\}$ be the vertices of levels zero and one, respectively. Note that there is no level-two vertex. Denote by Q_0 and Q_1 the quadrilaterals formed by the level-zero and the level-one vertices, respectively. Since Q_0 is convex, its

two diagonals (u_1, u_3) and (u_2, u_4) necessarily cross in its interior. Let w.l.o.g. (u_1, u_3) be long, while (u_2, u_4) be short. Similarly, the two diagonals (u_5, u_7) and (u_6, u_8) of Q_1 cross inside Q_1. Edges (u_1, u_3) and (u_2, u_4) define 4 triangular regions in Q_0. Note that no two vertices of Q_1 lie in two non-adjacent triangular regions of Q_0, as otherwise the two diagonals of Q_0 together with the edge connecting them would form three mutually crossing edges in Γ violating Property 2. Since Q_1 lies in the interior of Q_0, we distinguish the following subcases: **(a)** all four vertices of Q_1 lie inside a single triangular region of Q_0, **(b)** three vertices of Q_1 lie in a triangular region of Q_0, while the remaining one lies inside an adjacent triangular region, **(c)** two vertices of Q_1 lie inside a triangular region of Q_0, while the remaining two vertices of Q_1 lie inside an adjacent triangular region. Subcase (c) can be easily ruled out, as the two diagonals of Q_1 together with the edge of Q_0 separating the two adjacent triangular regions form three mutually crossing edges in Γ violating Property 2. The proof is then completed by bringing to a contradiction subcases (a) and (b).

From the analysis above, we conclude that $\theta_0(K_8) > 2$. To show that $\theta_0(K_8) = 3$, consider the drawing of K_7 with ordinal geometric thickness 2 (see Figure 3). Add a new vertex on a third layer, placing it sufficiently far from the existing vertices of K_7 so that every edge incident to the new vertex is longer than any edge in the drawing of K_7. This construction yields a drawing of K_8 with ordinal thickness 3. ◀

In the next two theorems, we provide upper bounds on the ordinal geometric thickness of the complete graph K_n for $n \geq 9$ and of the complete bipartite graph $K_{n,m}$.

▶ **Theorem 3.4.** *For $n \geq 7$, the ordinal geometric thickness of K_n is at most $\lceil \frac{n-3}{2} \rceil$.*

Proof sketch. By Corollary 3.2, we focus on the case $n > 7$. Assuming w.l.o.g. that n is odd, we present an incremental drawing algorithm maintaining the following invariants: **(I.1)** all vertices have distinct integer y-coordinates that range from 1 to the number of vertices of the drawn graph, **(I.2)** the two topmost vertices are the leftmost and the rightmost, **(I.3)** the y-coordinates of the neighbors of the leftmost (rightmost) vertex increase when its incident edges are traversed in counterclockwise (clockwise) order. We start with the drawing of K_7 depicted in Figure 6. The remaining $n-7$ vertices are added in pairs. For each pair, its two vertices are placed one and two units above the current drawing, and sufficiently to the left and right so that all invariants are maintained and all edges incident to these two vertices are longer than any edge in the current drawing. By I.3, edges incident to the two new vertices do not cross each other. Also, each pair increases the ordinal geometric thickness of the drawing by one, which implies the claimed bound. ◀

▶ **Theorem 3.5.** *The ordinal geometric thickness of $K_{n,m}$ is at most $\lceil \frac{\min\{n,m\}}{2} \rceil$.*

Proof. We adjust an algorithm by Dillencourt, Eppstein, and Hirschberg [6] and achieve the same upper bound. Let the vertex set of $K_{n,m}$ be $\{u_1, \ldots, u_n\} \cup \{v_1, \ldots, v_m\}$ and assume w.l.o.g. that m is even and $m \leq n$. Place $u_1, \ldots, u_n$ on the y-axis with $y(u_i) = i$ for $i = 1, \ldots, n$. Place $v_1, \ldots, v_m$ in pairs on the x-axis, i.e., $y(v_i) = 0$ for $i = 1, \ldots, n$. Initially set $x(v_1) = -1$ and $x(v_2) = 1$. For $1 < j < \frac{m}{2}$, suppose we have drawn the subgraph induced by $u_1, \ldots, u_n$ and $v_1, \ldots, v_{2j}$ satisfying: **(J.1)** v_{2j-1} and v_{2j} are the leftmost and rightmost vertices, and **(J.2)** (u_n, v_{2j-1}) and (u_n, v_{2j}) are the longest edges and have equal length. These invariants hold for $j = 1$. Place v_{2j+1} and v_{2j+2} at $(-\lceil d \rceil, 0)$ and $(\lceil d \rceil, 0)$, where d is the length of (u_n, v_{2j-1}). This preserves J.1 and J.2, and ensures that the shortest new edges, (u_1, v_{2j+1}) and (u_1, v_{2j+2}), are longer than d. The edges incident to v_{2j+1} and v_{2j+2} do not cross each other. Thus, the ordinal geometric thickness increases by 1 at each step. ◀

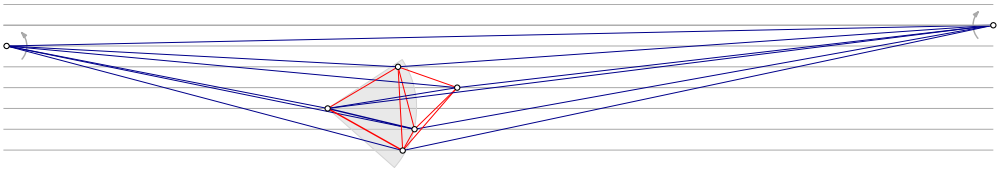


Figure 6 A drawing of K_7 satisfying Invariants I.1–I.3.

4 Future Directions

An immediate goal is to establish improved bounds on the ordinal geometric thickness of K_n and $K_{n,m}$. Since $\lceil \frac{n}{4} \rceil$ is the best upper bound on the geometric thickness of K_n [6], there may be room for improving our upper bound in Theorem 3.4. It would also be interesting to identify classes of graphs that admit drawings of low ordinal geometric thickness. Finally, one may consider variants of the model that allow bends on the edges.

References

1 Md. Jawaherul Alam, Stephen G. Kobourov, Sergey Pupyrev, and Jackson Toeniskoetter. Weak unit disk and interval representation of graphs. In Ernst W. Mayr, editor, *WG 2015*, volume 9224 of *LNCS*, pages 237–251. Springer, 2015. doi:10.1007/978-3-662-53174-7_17.

2 Valerii Borisovich Alekseev and V. S. Gončakov. The thickness of an arbitrary complete graph. *Mathematics of the USSR-Sbornik*, 30(2):567–577, 1976. doi:10.1070/SM1976v030n02ABEH002267.

3 Patrizio Angelini, Michael A. Bekos, Martin Gronemann, and Antonios Symvonis. Geometric representations of dichotomous ordinal data. In Ignasi Sau and Dimitrios M. Thilikos, editors, *WG 2019*, volume 11789 of *LNCS*, pages 205–217. Springer, 2019. doi:10.1007/978-3-030-30786-8_16.

4 Patrizio Angelini, Sabine Cornelsen, Carolina Haase, Michael Hoffmann, Eleni Katsanou, Fabrizio Montecchiani, Raphael Steiner, and Antonios Symvonis. Geometric realizations of dichotomous ordinal graphs. In Oswin Aichholzer and Haitao Wang, editors, *SoCG 2025*, volume 332 of *LIPIcs*, pages 9:1–9:16. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2025. doi:10.4230/LIPICS.SOCG.2025.9.

5 Lowell W. Beineke, Frank Harary, and John W. Moon. On the thickness of the complete bipartite graph. *Mathematical Proceedings of the Cambridge Philosophical Society*, 60(1):1–5, 1964. doi:10.1017/S0305004100037385.

6 Michael B. Dillencourt, David Eppstein, and Daniel S. Hirschberg. Geometric thickness of complete graphs. *J. Graph Algorithms Appl.*, 4(3):5–17, 2000. doi:10.7155/JGAA.00023.

7 Vida Dujmovic and David R. Wood. Graph treewidth and geometric thickness parameters. *Discret. Comput. Geom.*, 37(4):641–670, 2007. doi:10.1007/S00454-007-1318-7.

8 Christian A. Duncan. On graph thickness, geometric thickness, and separator theorems. *Comput. Geom.*, 44(2):95–99, 2011. doi:10.1016/J.COMGEO.2010.09.005.

9 Christian A. Duncan, David Eppstein, and Stephen G. Kobourov. The geometric thickness of low degree graphs. In Jack Snoeyink and Jean-Daniel Boissonnat, editors, *SoCG 2004*, pages 340–346. ACM, 2004. doi:10.1145/997817.997868.

10 Stephane Durocher, Ellen Gethner, and Debajyoti Mondal. Thickness and colorability of geometric graphs. *Comput. Geom.*, 56:1–18, 2016. doi:10.1016/J.COMGEO.2016.03.003.

11 Seok-Hee Hong, Peter Eades, Giuseppe Liotta, and Sheung-Hung Poon. Fáry's theorem for 1-planar graphs. In Joachim Gudmundsson, Julián Mestre, and Taso Viglas, editors,

COCOON 2012, volume 7434 of *LNCS*, pages 335–346. Springer, 2012. doi:10.1007/978-3-642-32241-9_29.

12 Anthony Mansfield. Determining the thickness of graphs is NP-hard. *Mathematical Proceedings of the Cambridge Philosophical Society*, 93(1):9–23, 1983. doi:10.1017/S030500410006028X.

13 Petra Mutzel, Thomas Odenthal, and Mark Scharbrodt. The thickness of graphs: A survey. *Graphs Comb.*, 14(1):59–73, 1998. doi:10.1007/PL00007219.

14 Dominik Peters. Recognising multidimensional Euclidean preferences. In Satinder Singh and Shaul Markovitch, editors, *AAAI-17*, pages 642–648, 2017. doi:10.1609/AAAI.V31I1.10616.

15 Carsten Thomassen. Rectilinear drawings of graphs. *J. Graph Theory*, 12(3):335–341, 1988. doi:10.1002/JGT.3190120306.

16 William T. Tutte. The thickness of a graph. *Indagationes Mathematicae (Proceedings)*, 66:567–577, 1963. doi:10.1016/S1385-7258(63)50055-9.

17 David R. Wood. Geometric thickness in a grid. *Discret. Math.*, 273(1-3):221–234, 2003. doi:10.1016/S0012-365X(03)00240-1.

Oracle Separations for $\mathbb{R}$PH

Thekla Hamm[1], Lucas Meijer[2], Till Miltzow[3], and Subhasree Patro[4]

1 **Eindhoven University**
2 **Utrecht University**
3 **Utrecht University**
4 **Eindhoven University**

Abstract

While theoretical computer science primarily works with discrete models of computation, like the Turing machine and the wordRAM, there are many scenarios in which introducing real computation models is more adequate. For example, when working with continuous probability distributions for say smoothed analysis, in continuous optimization, computational geometry or machine learning. Furthermore, there are various problems earlier known to be NP-hard that have been found to be $\exists\mathbb{R}$-complete. We want to investigate the relation between real models of computation and discrete models of computation. We do this by means of oracle separation results.

In this paper, we define the notion of a *real Turing machine* as an extension of the (binary) Turing machine by adding a real tape. On this real tape, each memory cell can hold a real number. (This is equivalent to BSS-machines.) Using those machines, we define and study the real polynomial hierarchy $\mathbb{R}$PH. We say that a language $L \subseteq \{0,1\}^*$ is in $\mathbb{R}$PH if there is a constant $k \geq 1$, a polynomial q, polynomial-time real Turing machine M, and (alternating) quantifiers $Q_i \in \{\exists, \forall\}$ such that

$$x \in L \Longleftrightarrow Q_1 u_1 \in \mathbb{R}^{q(|x|)} Q_2 u_2 \in \mathbb{R}^{q(|x|)} \cdots Q_k u_k \in \mathbb{R}^{q(|x|)} M(x, u_1, \ldots, u_k) = 1.$$

We refer to this as the real polynomial hierarchy, as the language L is binary, but we quantify over real numbers. The kth level of the hierarchy is denoted by $\Sigma_k\mathbb{R}$, and $\Pi_k\mathbb{R}$ respectively, depending on whether the first quantifier is existential or universal.

We are interested in $\mathbb{R}$PH as the first level of the hierarchy corresponds to the well-known complexity class $\exists\mathbb{R}$. It is known that NP $\subseteq \exists\mathbb{R} \subseteq$ PSPACE and furthermore PH $\subseteq \mathbb{R}$PH $\subseteq$ PSPACE. We are interested to know if any of those inclusions are tight. In the absence of unconditional separations of complexity classes, we turn our attention to oracle separation. Oracle machines can ask questions of the form $x \in O$ for some oracle O. We develop a technique that allows us to transform oracle separation results from the classical binary world to the real world. As an application of our technique, we show that there are oracles such that:

- $\mathbb{R}\text{PH}^O$ is a proper subset of PSPACE^O,
- Σ_{k+1}^O is not contained in $\Sigma_k\mathbb{R}^O$, for all $k \geq 0$,
- $\Sigma_k\mathbb{R}^O$ is a proper subset of $\Sigma_{k+1}\mathbb{R}^O$, for all $k \geq 0$,
- BQP^O is not contained in $\mathbb{R}\text{PH}^O$.

Our results indicate that $\exists\mathbb{R}$ is strictly less powerful than PSPACE and that there is a separation between the different levels of the real polynomial hierarchy. Additionally, we bound the power of real computations by showing that NP-hard problems are not likely to be solvable using polynomial time on a realRAM. Furthermore, our oracle separations indicate that polynomial-time quantum computing cannot be simulated on a real Turing machine.

Acknowledgments. L.M. and T.M. are generously supported by the Netherlands Organisation for Scientific Research (NWO) under project no. VI.Vidi.213.150. S.P. is cofunded by the European Union (ERC, ASC-Q, 101040624).

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

1 Introduction

1.1 Motivation

In theoretical computer science, discrete models of computation are dominant. They capture many algorithmic problems accurately and have a rich structural theory [3]. Yet, in many contexts, continuous numbers are more appropriate or even necessary to be used as part of the model of computation. The primary examples are computational geometry [14], machine learning [1], continuous optimization [2] and working with probabilities [13]. Yet, working with continuous numbers and thus inherently with infinite precision makes many researchers uneasy. The reasons for that are multitude. Firstly, it is clearly an unrealistic model of computation, as a physical computer can only work with finite precision. Secondly, it might be possible to exploit the real number computations to say solve problems much faster using real numbers compared to using binary numbers. For example, one such exploitation might be the ability to solve PSPACE-complete problems in polynomial time using real numbers. For instance, Shamir [12] shows how to factor integers, and Schönhage showed how to solve general PSPACE-complete problems [11]. However, both algorithms use the rounding operation in a clever way. Since then, the realRAM is considered not to have the rounding operation [9]. And without rounding, we are not aware of any other anomalies or exploitations with real number computations. Still, the fact that researchers have not found any new abnormalities does not mean that they do not exist. At last, we want to point out that real Turing machines are often considered with access to arbitrary real constants. In this case, it was shown that those models of computation are at least as powerful as P/`poly` [10] and the best upper bound is PSPACE/`poly` [8].

We want to show *limitations* on real number computations. Most known methods to separate complexity classes unconditionally use the diagonalization method [3]. As the diagonalization method is limited in scope, we turn our attention to oracle separations.

1.2 The Setup

Model of Real Computation. There are several ways to define computations with real numbers. The most common way to work with real numbers is the realRAM model of computation [9]. Its advantage is that it closely resembles modern computers and thus allows fine-grained computational analysis. In other words, we can distinguish between linear, quadratic time computations etc. However, that model of computation is very involved, specifically as it allows random access. (This means that the content of one word register is used to address the content of another register.)

The second common model is the Blum Shub Smale model (or BSS model) of computation, which allows for real inputs and real outputs [5, 4]. Its advantage is that it is closer to a Turing machine and thus more minimal. The downside is that it does not closely correspond to modern computer architectures. The research line building upon the BSS model often focuses on real input, where we focus on binary inputs $\{0,1\}^*$. Furthermore, originally the BSS model allowed the machine to have access to arbitrary real constants, whereas we are constant free.

As there are many definitions of BSS machines, compare [5] and [4] for example, and as we want to work with both real and binary input, we give our own definition of a real Turing machine. For us a real Turing machine has two tapes, one binary tape and one real tape. Furthermore, we are allowed the usual arithmetic operations: addition, subtraction, multiplication, and division. See Section A.1 (full version) for a detailed definition. The real

Turing machine we define is convenient for our purpose, and we show that it is equivalent to the previous models up to polynomial factors. Therefore, readers familiar with any other standard model of real computation may skip Section A.1.

Real Polynomial Hierarchy. We say that a language $L \subseteq \{0,1\}^*$ is in the real polynomial hierarchy $\mathbb{R}\mathrm{PH}$ if there is a constant $k \geq 1$, a univariate polynomial q, a polynomial-time real Turing machine M, and a sequence of (alternating) quantifiers $Q_1, \ldots, Q_k \in \{\exists, \forall\}$ such that

$$x \in L \Longleftrightarrow Q_1 u_1 \in \mathbb{R}^{q(|x|)} Q_2 u_2 \in \mathbb{R}^{q(|x|)} \cdots Q_k u_k \in \mathbb{R}^{q(|x|)} M(x, u_1, \ldots, u_k) = 1.$$

The kth level of the hierarchy $\mathbb{R}\mathrm{PH}$ consists of all languages decidable in this way using k quantifiers. Languages on the kth level are distinguished based on their first quantifier into $\Sigma_k\mathbb{R}$ (existential), and $\Pi_k\mathbb{R}$ (universal).

One can also define the *truly* real polynomial hierarchy $\mathsf{T}\mathbb{R}\mathrm{PH}$ in which even languages (rather than just computations) are over real numbers and not only over Boolean values. The truly real polynomial hierarchy is interesting in its own right, but not the topic of this article. It was considered in Chapter 21 by Blum, Cucker, Shub and Smale in the context of machines that are not allowed to do multiplication [4, Chapter 21]. For general real Turing machines, we only found a paper by Bürgisser and Cucker [6] that defines this hierarchy.

To us, the complexity classes $\mathrm{P}\mathbb{R} = \Sigma_0\mathbb{R}$ and $\exists\mathbb{R} = \Sigma_1\mathbb{R}$ are of primary importance. The complexity class $\mathrm{P}\mathbb{R}$ and $\exists\mathbb{R}$ are the real analogs of P and NP. The reason we find them most interesting is that there are more problems studied for those two classes compared to other levels of the real polynomial hierarchy. See Section 1.4 (full version) for some selected examples of problems in $\mathrm{P}\mathbb{R}$ and see Section 1.5 (full version) for an introduction to the existential theory of the reals $\exists\mathbb{R}$.

Oracles. When we consider an oracle model of computation, we assume that there is a set O and our model of computation is allowed to make queries of the form $x \in O$ and receives the answer in one time step. See Section A.3 (full version) for formal details, and Section 1.8 (full version) for more context on why oracle separations are arguably considered a step towards separating complexity classes. We want to say already here, that oracle separation results serve as a weak *indication* that complexity classes are separate. We discuss in Section 1.6 (full version) how to judge this "indication".

1.3 Results

Our first result separates the real polynomial hierarchy from PSPACE using random oracles.

▶ **Theorem 1.1.** *For a random binary oracle O, $\mathrm{PSPACE}^O \not\subseteq \mathbb{R}\mathrm{PH}^O$ with probability 1.*

To understand the significance of the theorem, let us consider the complexity class $\exists\mathbb{R} = \Sigma_1\mathbb{R}$, which is on the first level of the hierarchy. It is known that $\mathrm{NP} \subseteq \exists\mathbb{R} \subseteq \mathrm{PSPACE}$. Theorem 1.1 gives the first indication that $\exists\mathbb{R}$ is not equal to PSPACE. Again, looking at the gap between NP and PSPACE, we see that there are infinitely many layers of the polynomial hierarchy in between.

As $\exists\mathbb{R}$-hard problems are arguably harder to solve than problems in NP it is plausible that $\exists\mathbb{R}$ could for instance contain more levels of the polynomial hierarchy. The next theorem hints that this is not the case.

▶ **Theorem 1.2.** *For each $k \geq 0$, for a random binary oracle O, with probability 1 $\Sigma_{k+1}^O \not\subseteq \Sigma_k\mathbb{R}^O$.*

Note that the previous theorem implies the following corollary by choosing $k = 0$. (We denote by P$\mathbb{R}$ polynomial-time computations on a real Turing machine with binary input, also denoted as $\mathrm{BP}^0(P_{\mathbb{R}})$ in the literature.)

▶ **Corollary 1.3.** *There exists a binary oracle O such that $NP^O \not\subseteq P\mathbb{R}^O$.*

This hints that polynomial time realRAM computation is likely not capable of solving NP-hard problems. (Specifically, that the result by Bürgisser and Jindal [7] about NP-hardness of PosSLP is not true in full generality, which we discuss in the full version.) In other words, their assumptions are too strong. It is trivial that $\Sigma_k \subseteq \Sigma_k\mathbb{R}$, for every k, as we can restrict any real witness to be binary. This implies that we also have a separation of the different levels within ℝPH.

▶ **Corollary 1.4.** *For each $k \geq 0$, there exists a binary oracle O such that $\Sigma_k\mathbb{R}^O \subsetneq \Sigma_{k+1}\mathbb{R}^O$.*

At last, we turn our attention to quantum computing. We were wondering if $\exists\mathbb{R}$ would be powerful enough to simulate quantum computing. Quantum computing relies on quantum mechanics, where qubits can exist in superposition, representing a continuous range of states. Quantum algorithms manipulate these states through interference and entanglement, which allows for exploration of solution spaces differently than classical computers. Although the number of amplitudes to describe a state in superposition can be exponential, we hoped that one might be able to simulate those quantum computations using real numbers.

The idea was that the amplitudes of all quantum states could be stored by real numbers which could in turn be guessed by real Turing machines, i.e., BQP $\subseteq \exists\mathbb{R}$. Again, using oracle separation, we could show that this is not likely. We show an even stronger statement by comparing BQP with ℝPH.

▶ **Theorem 1.5.** *There exists an oracle O such that $BQP^O \not\subseteq \mathbb{R}PH^O$.*

The main message. The main message of our study is that working with this real model of computation *seems not* to give any great additional power to the binary counterparts. This confirms our intuition that most problems can be solved more or less equally fast on the wordRAM and the realRAM. This is in contrast to previous models, which appear significantly more powerful. For example, the simple addition of rounding allows a real Turing machine to compute PSPACE-complete problems in polynomial time. So, despite some models that work with real numbers being extraordinarily powerful, we convey that simply using the realRAM is not an all-powerful model that can solve problems unrealistically fast.

Foundations. In order to have a self-contained work, we give a detailed account with elaborate definitions of our model of real computation and other basic concepts, see **??**.

References

1 Ethem Alpaydin. *Introduction to machine learning*. MIT press, 2020.
2 Niclas Andréasson, Anton Evgrafov, and Michael Patriksson. *An introduction to continuous optimization: foundations and fundamental algorithms*. Courier Dover Publications, 2020.
3 Sanjeev Arora and Boaz Barak. *Computational Complexity - A Modern Approach*. Cambridge University Press, 2009.
4 Lenore Blum, Felipe Cucker, Michael Shub, and Steve Smale. *Complexity and real computation*. Springer-Verlag, New York, 1998. With a foreword by Richard M. Karp.
5 Lenore Blum, Mike Shub, and Steve Smale. On a theory of computation and complexity over the real numbers: NP-completeness, recursive functions and universal machines. *Bull. Amer. Math. Soc. (N.S.)*, 21(1):1–46, 1989.

6 Peter Bürgisser and Felipe Cucker. Exotic quantifiers, complexity classes, and complete problems. *Found. Comput. Math.*, 9(2):135–170, 2009.

7 Peter Bürgisser and Gorav Jindal. On the hardness of PosSLP. In *Proceedings of the 2024 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1872–1886. SIAM, Philadelphia, PA, 2024.

8 Felipe Cucker and Dima Grigoriev. On the power of real Turing machines over binary inputs. *SIAM Journal on Computing*, 26(1):243–254, 1997.

9 Jeff Erickson, Ivor van der Hoog, and Tillmann Miltzow. Smoothing the gap between NP and ER. In *2020 IEEE 61st Annual Symposium on Foundations of Computer Science*, pages 1022–1033. IEEE Computer Soc., Los Alamitos, CA, 2020.

10 Pascal Koiran. Computing over the reals with addition and order. *Theor. Comput. Sci.*, 133(1):35–47, 1994.

11 Arnold Schönhage. On the power of random access machines. In *International Colloquium on Automata, Languages, and Programming*, pages 520–529. Springer, 1979.

12 Adi Shamir. Factoring numbers in O(log n) arithmetic steps. 1977.

13 Daniel A Spielman and Shang-Hua Teng. Smoothed analysis of algorithms: Why the simplex algorithm usually takes polynomial time. *Journal of the ACM (JACM)*, 51(3):385–463, 2004.

14 Csaba Toth, Joseph O'Rourke, and Jacob Goodman. *Handbook of discrete and computational geometry*. CRC press, 2017.

Dynamic Level Planarity Testing

Jonathan Højlev[1], Simon D. Fink[2], and Eva Rotenberg[3]

1 **Technical University of Denmark**
`jonathan.hoejlev@gmail.com`
2 **Algorithms and Complexity Group, TU Wien, Austria**
`sfink@ac.tuwien.ac.at`
Vienna Science and Technology Fund (WWTF) grant [10.47379/ICT22029] and Austrian Science Fund (FWF) grant [10.55776/Y1329]
3 **IT University of Copenhagen**
`erot@itu.dk`

Abstract

We study the Dynamic Level Planarity problem, where a directed graph G with vertices assigned to horizontal lines called *levels* is subject to insertions and deletions, and we at every step want to check whether it can be drawn planarly while respecting the leveling. We build upon a 2-SAT model of Level Planarity by Randerath et al. [24], constructing an *implication graph* that models Level Planarity as graph connectivity problem. Applying known results on dynamic graph connectivity together with a careful amortized analysis, we obtain efficient algorithms that solve the incremental, decremental, and fully-dynamic settings for Level Planarity. To the best of our knowledge, this is the first ever dynamic algorithm for a constrained planarity variant.

1 Introduction

Planarity testing is a fundamental problem both in graph theory and practical graph drawing. Driven by applications, many variants of the problem have been considered. Constrained Planarity studies graph planarity under additional constraints such as hierarchies, clusterings, or upward-directed edges. An important and well-studied example is Level Planarity [5, 23], where vertices lie on fixed horizontal levels and edges should be drawn crossing-free and upwards. In 1998, Jünger et al. gave a linear-time algorithm for Level Planarity [22, 23], however their algorithm is quite involved and could not be implemented [2]. A slightly simpler linear-time algorithm has been proposed in a 2021 PhD thesis [2] but also not been put into practice. Because of these practical difficulties, various simpler, quadratic-time algorithms were proposed [14, 15, 24], but all with some flaws [11]. In general, Level Planarity has seen a surprising number of flawed publications, see [12, Remark 4.3] for an overview.

Another common and well-studied extension is Dynamic Planarity, where we want to maintain whether a graph is planar under the insertion or deletion of vertices and edges. A seminal paper from the 1990s by Di Battista et al. showed that incremental planarity can be checked in inverse Ackermann (denoted $\alpha(\cdot)$) amortized time per insertion using a data structure called SPQR-tree [6, 7]. A more general version was studied, resulting in a *fully-dynamic* algorithm allowing also for edge deletions, albeit with a worse running time of $O(\sqrt{n})$ per operation, based on a sparsification through separators technique [8, 9]; see also [20]. This result has more recently been improved to polylogarithmic time, again using SPQR-trees [19]. When restricting the problem to the maintenance of a certain given embedding, polylogarithmic algorithms are known for planar embeddings [18, 21, 26] as well as upward-planar embeddings of single-source graphs [25, 28]. See also [17] for an overview over other dynamic graph algorithms on planar graphs and [13] for a survey on dynamic graph algorithms.

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

We consider the combination of these two planarity variants, that is Dynamic Level Planarity, where we want to maintain whether a graph has a level planar drawing while undergoing insertions or deletions. For the incremental case, we obtain an approach with an almost-linear running time per insertion. In the decremental case, our running time is slightly worse, but still sub-quadratic. While this does not beat re-running the static linear-time level planarity algorithm [22, 23], the latter is rather complex and all prior attempts at an implementation have failed [3, 11], while our approach is much simpler and thereby more practical by improving upon the many static quadratic-time approaches to Level Planarity.[1] While doing so, we also obtain better running times for instances that have levels only containing a bounded number of vertices. We present the first non-trivial incremental and decremental algorithm for level planarity without a fixed embedding, and we are currently working an extension of these results to the fully-dynamic setting.

2 Preliminaries

Let $G = (V, E)$ be a simple directed acyclic graph and $k \in \mathbb{N}$. A *leveling* of G is a map $l : V \to \{1, \dots, k\}$ such that $l(y) > l(x)$ for all $(x, y) \in E$, and the leveling is *proper* if $l(y) - l(x) = 1$ for all $(x, y) \in E$. If $l(v) = j$, then v is called a level-j vertex. The set $V_j = l^{-1}(j)$ of all level-j vertices is called a *level*. Denote by E_j the set of edges between V_j and V_{j+1}. In this paper, we will only consider proper levelings. Note that any leveling can be made proper by subdividing edges, which increases graph size by a factor at most k. In an upcoming full version, we show that the running times we find transfer to non-proper instances with mostly unchanged runtimes.

A planar drawing of G is *level planar* if every vertex v has y-coordinate $l(v)$ and all edges are y-monotone. A level planar *embedding* of G is the equivalence class of all level planar drawings with the same vertex orders on all layers. We thus denote such embedding $\prec$ as the total order of vertices obtained by combining the orders of each level and setting $x \prec y$ whenever $l(x) < l(y)$. Let $\prec_j$ denote the order of vertices on level j.

2.1 Solving 2-SAT

Given a boolean formula C in conjunctive normal form and where each clause contains exactly two literals, also called a *2-CNF formula*, the problem of determining whether there exists an assignment of variables that satisfies the formula is called 2-SAT. Using the fact that a clause of the form $(x \vee y)$ is equivalent to $(\neg x \Rightarrow y)$ and $(\neg y \Rightarrow x)$, Tarjan et al. [1] construct a directed *implication graph* $\mathcal{G}(C)$ where each boolean variable x and its complement $\neg x$ are vertices, and each clause $(x \vee y)$—where x and y can represent either a variable or its negation—adds two directed edges to $\mathcal{G}(C)$: $(\neg x, y)$ and $(\neg y, x)$. They then show the following theorem which they use to get a linear-time algorithm to answer 2-SAT.

▶ **Theorem 2.1** ([1]). *The expression C is satisfiable if and only if no vertex x in $\mathcal{G}(C)$ has a directed path to its complement $\neg x$, which can be checked in $O(|\mathcal{G}(C)|)$ total time.*

[1] A straight-forward implementation of the 2-SAT formulation of Level Planarity is available from Fink et al. [11], see the `compute2SATClasses` function of [10]. While the incremental Union-Find we use is a common asset of many graph software libraries such as the OGDF [4], we refer to the survey [13] for an overview over practical implementations of fully-dynamic connectivity algorithms.

2.2 Modeling Level Planarity as Satisfiability

Recall that a level planar embedding is entirely specified by the vertex orders on each level. These orders can be described by a set of boolean variables $V' = \{uv \mid u, v \in V(G), u \neq v, l(u) = l(v)\}$, where uv is true if and only if u is to the left of v on their common level j, i.e. $u \prec_j v$. Using this modeling, Randerath et al. [24] show[2] that Level Planarity can be modeled through consistency (1), planarity (2), and transitivity (3) constraints:

$$\forall uv \quad \in V' \qquad\qquad : uv \quad \Leftrightarrow \neg vu \tag{1}$$

$$\forall uv, xy \in V', (u,x), (v,y) \in E : uv \quad \Leftrightarrow xy \tag{2}$$

$$\forall uv, vw \in V' \qquad\qquad : uv \wedge vw \Rightarrow uw \tag{3}$$

Each solution to this constraint system corresponds bijectively to a vertex order that represents a level planar embedding. If no solution exists, this is a proof that the graph is not level planar [24]. Although using all three constraints requires solving a 3-SAT problem, Randerath et al. [24] show that dropping transitivity and using only constraints (1) and (2) suffices for level planarity testingand only requires solving 2-SAT. Denote by $\mathcal{M}$ the 2-SAT instance with variables V' and clauses $(\neg uv \vee xy) \wedge (uv \vee \neg xy)$ for every $(uv \Leftrightarrow xy)$ biimplication from constraints (1) and (2). Observe that the implication graph $\mathcal{G}(\mathcal{M})$ has edges (uv, xy) and (vu, yx) for every $uv \Leftrightarrow xy$ constraint; see Figure 1. Using Theorem 2.1 on $\mathcal{M}$, that is checking the connected components of $\mathcal{G}(\mathcal{M})$, yields the following.

▶ **Theorem 2.2** ([3, 24]). *Given a proper level graph G with n vertices, level planarity testing can be done in $O(n^2)$ time.*

Interestingly, if we assume a bound on the size of each level V_j, the algorithm can get a sub-quadratic runtime, which to our knowledge is not noted anywhere in the literature so far.

▶ **Corollary 2.3.** *Given a proper level graph G with n vertices and at most n^w vertices on each level for some $0 \leq w \leq 1$, level planarity testing can be done in $O(n^{1+w})$ time.*

Proof. Since $|V_j| \leq n^w$ for each j, we have $n = \sum_j |V_j| \leq k \cdot n^w$, giving $k \leq n^{1-w}$; the bound is maximized at $k = n^{1-w}$. Thus $|V'(G)| = \sum_{j=1}^{k} 2\binom{|V_j|}{2} \leq k \cdot (n^w)^2 \leq n^{1-w} \cdot n^{2w} = n^{1+w}$, and Theorem 2.1 yields the desired runtime. ◀

2.3 Optimizations to the Implication Graph

Notice that there are many symmetries in $\mathcal{G}(\mathcal{M})$ that can be exploited to arrive at a simpler graph. Firstly, any directed edge in $\mathcal{G}(\mathcal{M})$ is accompanied by an edge in its reversed direction, so directed connectivity in this graph becomes equivalent to connectivity in the underlying undirected graph. As a bidirected edge denotes equivalence of the variables at its endpoints, we contract all edges from constraint (1) and turn all other (bidirected) edges into undirected ones to obtain the undirected graph H with vertex set V' and edges $\{(uv, xy) \mid uv, xy \in V', (u,x), (v,y) \in E(G)\}$. We call vu the *complement* of uv.

Another insight about H is that each of its connected components is accompanied by an isomorphic *mirror* version, where the labels of all vertices are inverted; see Figure 1. In a practical implementation, this can be used to further halve the size of the represented graph.

[2] A flaw in their proof found by Fink et al. [11] has been resolved by Brückner et al. [3], at the cost of not (directly) being able to obtain an embedding if the test succeeds.

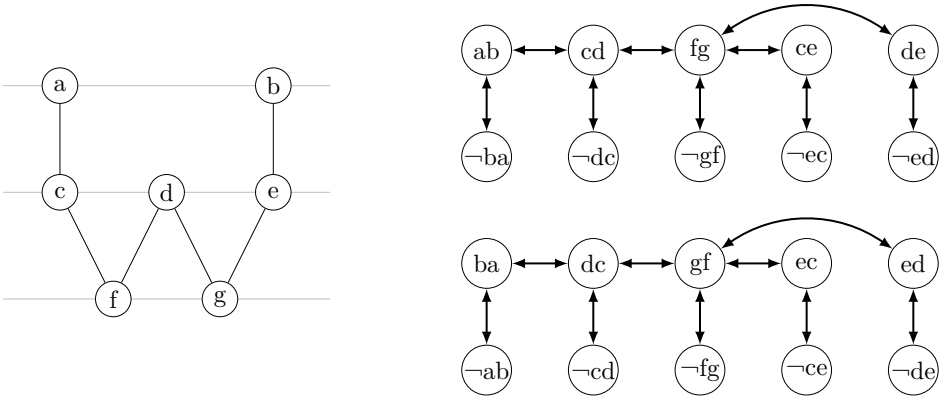


Figure 1 An example level planar graph G and the implication graph $\mathcal{G}(\mathcal{M})$ that is derived from its consistency and planarity constraints.

3 Incremental Level Planarity

The incremental variant of Level Planarity considers a level graph G to which vertices and edges are added. After each change, it should be reported whether the graph remained level planar or not. Recall that using the 2-SAT modeling reduces level planarity to a connectivity problem on its implication graph. In the incremental setting, we will now maintain this 2-SAT model under the insertions and perform the connectivity check for its implication graph incrementally by using the well-known Union-Find data structure by Tarjan [27].

More specifically, our algorithm for incremental level planarity testing preprocesses the graph G to construct its undirected implication graph H of its 2-SAT formula $\mathcal{M}$ and sets up a Union-Find data structure $\mathcal{U}$ modeling the connected components of H. Observe that a new vertex in G simply adds more variables to $\mathcal{M}$ (and thereby vertices in H) but, due to the optimizations of Section 2.3), no new clauses in $\mathcal{M}$ or edges in H. In contrast, a new edge in G adds new planarity constraints (2) to $\mathcal{M}$ and thus new edges to H. Again employing Theorem 2.1 to the updated instance, the question now becomes how to detect when the addition of such edge completes any path that connects a literal to its complement. We fortunately only need to recheck a bounded set of literals for this, as shown in the following.

▶ **Lemma 3.1.** *The addition of an edge to G breaks level planarity if and only if one of the edges consequently added to H makes one of its endpoints connected to its complement.*

Proof. By Theorem 2.1 the satisfiability of $\mathcal{M}$ is violated exactly when a literal uv is connected to its complement vu. Because of the mirror symmetry noted in Section 2.3, this happens exactly when the connected component B of uv is connected to its mirror, which especially connects every vertex in B to its complement. As an added edge only changes the connectivity of the at most two components it connects, the lemma follows. ◀

Altogether, this allows for an incremental algorithm with the following runtime bounds. Note that we need not know in advance how big G will be—it is just used to state the running time.

▶ **Theorem 3.2.** *For a proper level graph G with initially n vertices undergoing vertex and edge insertions, let N denote the maximum number of vertices G will contain. Incremental level planarity testing can be performed in $O(N)$ time per vertex insertion and $O(N\alpha(N))$ time per edge insertion after $O(n^2)$ preprocessing time.*

Proof. As preprocessing, the simplified implication graph H is constructed from G and a Union-Find structure $\mathcal{U}$ is statically built on H. This can be done in $O(n^2)$ time.

Whenever an edge is added to E_j in G, up to $|E_j| = O(N)$ edges are added to H, requiring up to as many Merge operations in $\mathcal{U}$. By Lemma 3.1, if and only if two mirror connected components are merged, level planarity of G is violated, and in that case the algorithm reports non-level planarity for this and all further updates. If that does not occur, G is still level planar. This checking and merging takes $O(N\alpha(N))$ time. If a vertex is added to G at level V_j, then $|V_j|$ new vertices are added to H with no edges, which takes $O(|V_j|) = O(N)$ time. ◀

As in the static case, bounding the level sizes in this proof again gives better runtimes.

▶ **Corollary 3.3.** *Given a proper level graph G with initially n vertices undergoing vertex and edge insertions, if the size of a level never exceeds n^w for some $0 \leq w \leq 1$, incremental level planarity testing can answered in $O(n^w)$ time per vertex insertion and $O(n^w\alpha(n))$ time per edge insertion after $O(n^{1+w})$ preprocessing time.*

4 Decremental Level Planarity

In the decremental case, edges are deleted from G and we want to report which deletion makes the graph level planar. We use the same approach as above and focus on the connectivity of the implication graph H, for which we now use the fully-dynamic connectivity structure by Holm, de Lichtenberg, and Thorup [16]. An edge removal from G removes $O(n)$ edges from H and we want to detect once every literal is disconnected from its complement.

▶ **Theorem 4.1.** *For a proper level graph G with n vertices undergoing $O(n)$ edge deletions, decremental level planarity can be answered in amortized $O(n \log^2 n)$ per deletion.*

Proof. As preprocessing, the implication graph H is constructed from G and the fully-dynamic connectivity structure $\mathcal{C}$ of H is built in $O(n \log^2 n)$ time [16] . For each pair of complement literals in V', query $\mathcal{C}$ whether they are connected and, if so, add the pair to a list $\mathcal{L}$ of certificates that G is not level planar. When an edge is removed from E_i in G, there are $O(|E_i|) = O(n)$ edges in H that need to be removed. After these removals, some of the pairs in $\mathcal{L}$ might have become separated. We iteratively query the connectivity of each pair in $\mathcal{L}$ via $\mathcal{C}$. If a pair is no longer connected, it is removed from $\mathcal{L}$, otherwise it is a certificate that G is still not level planar, and the checking stops. If all pairs are separated and $\mathcal{L}$ becomes empty then it is reported that G is level planar. Due to Theorem 2.1, as long as any pair of complement literals is connected, that is as long as $\mathcal{L}$ is non-empty, $\mathcal{M}$ is not satisfiable and G is not level planar, showing correctness of our approach. Each update to $\mathcal{C}$ takes $O(\log^2 n)$ time, while a connectivity query takes $O(\log n)$ [16]. Deleting d edges from G thus takes $O(dn \log^2 n)$ time. Checking the $O(n^2)$ entries of $\mathcal{L}$ in total takes $O((n^2 + d) \log n)$ time: each certificate is removed at most once while a certificate not being removed only happens once per deletion. The runtime follows by amortizing over $O(n)$ deletions. ◀

References

1 Bengt Aspvall, Michael F. Plass, and Robert Endre Tarjan. A linear-time algorithm for testing the truth of certain quantified boolean formulas. *Information Processing Letters*, 8(3):121–123, 1979. doi:10.1016/0020-0190(79)90002-4.

2 Guido Brückner. *Planarity Variants for Directed Graphs*. PhD thesis, Karlsruhe Institute of Technology, Germany, 2021. URL: https://nbn-resolving.org/urn:nbn:de:101:1-20210804050229888868936.

3 Guido Brückner, Ignaz Rutter, and Peter Stumpf. Level planarity: Transitivity vs. even crossings, 2018. arXiv:1808.09931.

4 Markus Chimani, Carsten Gutwenger, Michael Jünger, Gunnar W. Klau, Karsten Klein, and Petra Mutzel. The Open Graph Drawing Framework (OGDF). In Roberto Tamassia, editor, *Handbook on graph drawing and visualization*, pages 543–569. Chapman and Hall/CRC, 2013. URL: https://cs.brown.edu/people/rtamassi/gdhandbook/chapters/ogdf.pdf.

5 Giuseppe Di Battista and Enrico Nardelli. Hierarchies and planarity theory. *IEEE Transactions on Systems, Man, and Cybernetics*, 18(6):1035–1046, 1988. doi:10.1109/21.23105.

6 Giuseppe Di Battista and Roberto Tamassia. On-line maintenance of triconnected components with SPQR-trees. *Algorithmica*, 15(4):302–318, April 1996. doi:10.1007/BF01961541.

7 Giuseppe Di Battista and Roberto Tamassia. On-Line Planarity Testing. *SIAM Journal on Computing*, 25(5):956–997, October 1996. doi:10.1137/S0097539794280736.

8 David Eppstein, Zvi Galil, Giuseppe F. Italiano, and Thomas H. Spencer. Separator Based Sparsification: I. Planarity Testing and Minimum Spanning Trees. *Journal of Computer and System Sciences*, 52(1):3–27, February 1996. doi:10.1006/jcss.1996.0002.

9 David Eppstein, Zvi Galil, Giuseppe F. Italiano, and Thomas H. Spencer. Separator-Based Sparsification II: Edge and Vertex Connectivity. *SIAM Journal on Computing*, 28(1):341–381, January 1998. doi:10.1137/S0097539794269072.

10 Simon D. Fink. Level planarity test by Harrigan & Healy – LevelPlan2SAT.ipynb. URL: https://github.com/N-Coder/HarriganHealy/blob/main/LevelPlan2SAT.ipynb, swh:1:cnt:30e0b55f197c73a097acf2dc749abd762049a75f, 2024.

11 Simon D. Fink, Matthias Pfretzschner, Ignaz Rutter, and Peter Stumpf. Level planarity is more difficult than we thought, 2024. arXiv:2409.01727.

12 Radoslav Fulek, Michael J. Pelsmajer, Marcus Schaefer, and Daniel Štefankovič. Hanani–Tutte, Monotone Drawings, and Level-Planarity. In János Pach, editor, *Thirty Essays on Geometric Graph Theory*, pages 263–287. Springer, New York, NY, 2013. doi:10.1007/978-1-4614-0110-0_14.

13 Kathrin Hanauer, Monika Henzinger, and Christian Schulz. Recent Advances in Fully Dynamic Graph Algorithms – A Quick Reference Guide. *ACM J. Exp. Algorithmics*, 27, December 2022. doi:10.1145/3555806.

14 Martin Harrigan and Patrick Healy. Practical level planarity testing and layout with embedding constraints. In Seok-Hee Hong, Takao Nishizeki, and Wu Quan, editors, *15th07*, volume 4875 of *Lecture Notes in Computer Science*, pages 62–68. Springer, 2007. doi:10.1007/978-3-540-77537-9_9.

15 Patrick Healy and Ago Kuusik. Algorithms for multi-level graph planarity testing and layout. *Theoretical Computer Science*, 320(2):331–344, June 2004. doi:10.1016/j.tcs.2004.02.033.

16 Jacob Holm, Kristian de Lichtenberg, and Mikkel Thorup. Poly-logarithmic deterministic fully-dynamic algorithms for connectivity, minimum spanning tree, 2-edge, and biconnectivity. *J. ACM*, 48(4):723–760, July 2001. doi:10.1145/502090.502095.

17 Jacob Holm, Giuseppe F. Italiano, Adam Karczmarz, Jakub Lacki, and Eva Rotenberg. Decremental spqr-trees for planar graphs. *CoRR*, abs/1806.10772, 2018. arXiv:1806.10772.

18 Jacob Holm and Eva Rotenberg. Dynamic Planar Embeddings of Dynamic Graphs. *Theory of Computing Systems*, 61(4):1054–1083, November 2017. doi:10.1007/s00224-017-9768-7.

19 Jacob Holm and Eva Rotenberg. Fully-dynamic planarity testing in polylogarithmic time. *CoRR*, abs/1911.03449, 2019. URL: http://arxiv.org/abs/1911.03449, arXiv:1911.03449.

20 Giuseppe F. Italiano. Fully Dynamic Planarity Testing. In *Encyclopedia of Algorithms*, pages 806–808. Springer, New York, NY, 2016. doi:10.1007/978-1-4939-2864-4_157.

21 Giuseppe F. Italiano, Johannes A. La Poutré, and Monika H. Rauch. Fully dynamic planarity testing in planar embedded graphs. In *1st Annual European Symposium on Algorithms (ESA 1993)*, pages 212–223. Springer, 1993. doi:10.1007/3-540-57273-2_57.

22 Michael Jünger and Sebastian Leipert. Level planar embedding in linear time. *Journal of Graph Algorithms and Applications*, 6(1):67–113, 2002. doi:10.7155/jgaa.00045.

23 Michael Jünger, Sebastian Leipert, and Petra Mutzel. Level Planarity Testing in Linear Time. In Gerhard Goos, Juris Hartmanis, Jan Van Leeuwen, and Sue H. Whitesides, editors, *Graph Drawing*, volume 1547, pages 224–237. Springer Berlin Heidelberg, 1998. doi:10.1007/3-540-37623-2_17.

24 Bert Randerath, Ewald Speckenmeyer, Endre Boros, Peter Hammer, Alex Kogan, Kazuhisa Makino, Bruno Simeone, and Ondrej Cepek. A satisfiability formulation of problems on level graphs. *Electronic Notes in Discrete Mathematics*, 9:269–277, 2001. LICS 2001 Workshop on Theory and Applications of Satisfiability Testing (SAT 2001). doi:10.1016/S1571-0653(04)00327-0.

25 Aimal Rextin and Patrick Healy. Dynamic Upward Planarity Testing of Single Source Embedded Digraphs. *The Computer Journal*, 60(1):45–59, January 2017. doi:10.1093/comjnl/bxw064.

26 Roberto Tamassia. On-line Planar Graph Embedding. *Journal of Algorithms*, 21(2):201–239, September 1996. doi:10.1006/jagm.1996.0044.

27 Robert Endre Tarjan. Efficiency of a good but not linear set union algorithm. *J. ACM*, 22(2):215–225, April 1975. doi:10.1145/321879.321884.

28 Ivor van der Hoog, Irene Parada, and Eva Rotenberg. Dynamic embeddings of dynamic single-source upward planar graphs. *CoRR*, abs/2209.14094, 2022. arXiv:2209.14094, doi:10.48550/ARXIV.2209.14094.

Partitioning a Tile Arrangement for Optimal Construction by a Team of Simple Robots

Sándor P. Fekete[*1], Jonas Friemel[*2], Prahlad Narasimhan Kasthurirangan[3], Ramin Kosfeld[*4], Christian Scheffer[*2], and Arne Schmidt[4]

1 **Department of Computer Science, TU Braunschweig and L3S, Germany**
s.fekete@tu-bs.de
2 **Department of Electrical Engineering and Computer Science, Bochum University of Applied Sciences, Germany**
{jonas.friemel, christian.scheffer}@hs-bochum.de
3 **Applied Mathematics and Statistics Department at SUNY, Stony Brook University, New York, USA**
prahladnarasim.kasthurirangan@stonybrook.edu
4 **Department of Computer Science, TU Braunschweig, Germany**
{kosfeld, aschmidt}@ibr.cs.tu-bs.de

Abstract

We investigate the problem of optimally partitioning a given polyomino-shaped area into a disjoint set of connected subareas, such that a team of simple robots can build it in shortest possible time. Each subarea must contain its own depot from a given set, which supplies the robots with tile-sized building material. A robot can carry one tile at a time from its supply depot to a new position, while moving on existing tiles of its subconfiguration. As a result, the cost for constructing a subarea corresponds to the sum of distances from its respective depot to all of its tile sites and the total cost is the cost of the most expensive subarea. We provide hardness results for two variants of the resulting clustering problem. For given depots, it is NP-hard to decide if tiles can be assigned to the depots with a maximum cost of 3 even for rectangular instances. If the depot locations can be chosen, the problem is already hard for a maximum cost of 2. For lower costs, both variants are in P.

1 Introduction

How can a team of simple active robots [17] build and modify structures consisting of many basic components? Such questions play an important role, both in fundamental theory and in a spectrum of practical settings, with sizes ranging from very large [12] to very small dimensions [25], both of which are hard to access for direct human manipulation, e.g., in extraterrestrial space [4] or in microscopic environments [2].

As illustrated in Figure 1, it is possible to address such challenges by employing simple autonomous agents with minimal capabilities: The shown robot can move on the tile arrangement, remove individual tiles and physically relocate them to a new position by walking on the remaining configuration, which needs to remain connected at all times.

Previous work [3, 14] has described how to handle these challenges with a single active robot. Here we take an important step towards more scalable methods: How can we split the full task between multiple robots to achieve shorter overall construction times? To this end, we assume that building material (consisting of a supply of identical, pixel-shaped tiles)

* Supported by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – 530918134.

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

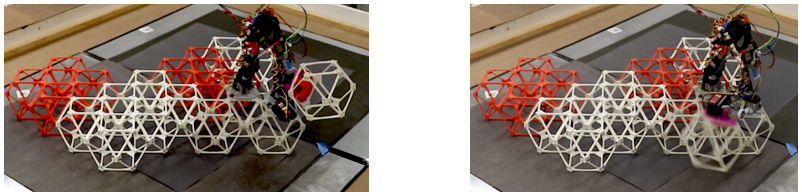


Figure 1 A simple BILL-E robot (left) carrying a light-weight tile while moving on an existing configuration and (right) placing an individual voxel for extending it. Figures by Yannuzzi [23].

is provided from a set of given depots. To ensure that robots do not collide, we partition the region into disjoint subregions, each containing a depot. Robots can pick up the necessary tiles to carry them to and place them at their destinations, while walking over previously constructed parts of their respective subregions. As a consequence, the resulting construction cost for each subregion is the sum of distances from the respective depot and the total construction cost of the system is the maximum construction cost for any subregion.

Our contributions. We investigate the problem of optimally subdividing a given polyomino-shaped area into a disjoint set of connected regions with centers, such that the maximum sum of geodesic distances of tiles assigned to a center is minimized. The cost function corresponds to the time it takes a team of simple robots to construct the shape, but the underlying clustering problem is of independent interest. We provide the following results.

(1) Deciding whether tiles in a polyomino P can be assigned to given cluster centers is NP-hard with a maximum cost of 3 per center, even if P is a rectangle (Theorem 4.2), but solvable in polynomial time if the cost is at most 2 (Theorem 4.4).
(2) If the locations of the cluster centers can be freely chosen, the problem is NP-hard for a cost of 2 (Theorem 5.1), but solvable in polynomial time for a cost of 1 (Theorem 5.2).

Due to limited space, we only provide proof sketches to theorems marked with ($\star$).

2 Preliminaries

For a set $P \subset \mathbb{Z}^2$ of N integer grid points in the plane, the graph G_P is the induced grid graph, in which two vertices $p_1, p_2 \in P$ are connected by an edge exactly if their L_1-distance is 1. The distance $d_P : P^2 \to \mathbb{N}$ is the length of a shortest path between two points $p_1, p_2 \in P$ in G_P. We call P *connected* if and only if G_P is connected. If P is connected, we consider its *polyomino* by replacing each point $p \in P$ with a unit square centered at p, which we refer to as *tile*. Throughout the paper, if the context is clear, we refer to P as the polyomino, and to G_P as the dual graph of the polyomino.

Let P be a polyomino and $C \subseteq P$ a set of *centers*. Consider an assignment function $\mathcal{A} : P \to C$ of all tiles in P to tiles in C, where $\mathcal{A}(c) = c$ for all $c \in C$. We denote the set of all tiles assigned to some $c \in C$ by $\mathcal{A}^{-1}(c)$ and for $t \in \mathcal{A}^{-1}(c)$, we say that c *covers* t. If $\mathcal{A}^{-1}(c)$ is connected for all $c \in C$, we call $\mathcal{A}$ a *C-clustering* in P. In such a C-clustering, we define the *weight* of a center $c \in C$ as the sum of distances in the subgraph $G_{\mathcal{A}^{-1}(c)}$ of G_P induced by $\mathcal{A}^{-1}(c)$ from every tile in $\mathcal{A}^{-1}(c)$ to c, i.e., $w(c) = \sum_{t \in \mathcal{A}^{-1}(c)} d_{\mathcal{A}^{-1}(c)}(t, c)$, and denote it by $w(c)$. The weight $w(\mathcal{A})$ of $\mathcal{A}$ is $\max_{c \in C} w(c)$.

In this paper, we consider the following two problems.

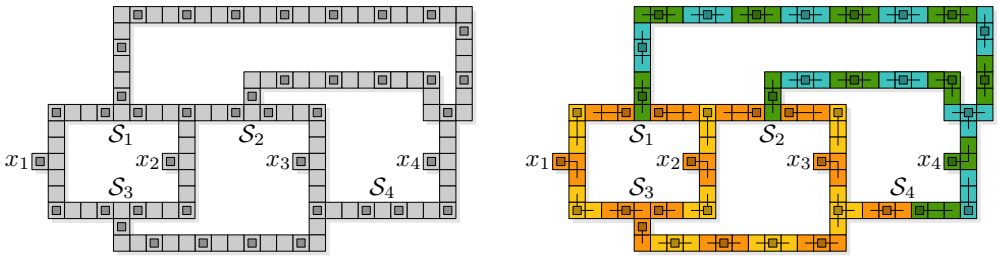


Figure 2 (a) A PCAP construction for the 3-SAT instance $\mathcal{S} = \bigwedge_{i=1}^{4} \mathcal{S}_i = (x_1 \vee x_2 \vee x_4) \wedge (x_2 \vee x_3 \vee x_4) \wedge (\overline{x_1} \vee \overline{x_2} \vee \overline{x_3}) \wedge (\overline{x_3} \vee \overline{x_4})$ in the proof of Lemma 4.1. Tiles are the bigger squares and cluster centers are marked with smaller squares. **(b)** Tile assignments are visualized with lines from the centers and alternating colors (green-cyan for positive and orange-yellow for negative variables). $\mathcal{S}$ is satisfied with $(x_1, x_2, x_3, x_4) = (\texttt{false}, \texttt{false}, \texttt{false}, \texttt{true})$.

Polyomino Cluster Assignment Problem (PCAP)
Given: The dual graph G_P of a polyomino P, a set $C \subseteq P$, and a number $\sigma \in \mathbb{N}$.
Question: Is there a C-clustering of weight at most σ?

Polyomino Clustering Problem (PCP)
Given: The dual graph G_P of a polyomino P and numbers $k, \sigma \in \mathbb{N}$.
Question: Is there a set $C \subseteq P$ with $|C| \leq k$ that yields a C-clustering of weight at most σ?

Clearly, both PCP and PCAP are in NP as we can compute the weight of a C-clustering in time polynomial in the size of G_P and verify whether $|C| \leq k$ and $w(C) \leq \sigma$.

3 Related Work

Clustering is a fundamental problem in computer science; see, for example, [8, 13, 19, 20]. While work on classic variants such as k-mean and k-median spans decades [5, 7, 16, 21], our problems of interest are in the intersection of two relatively new clustering variants: connected k-clustering [10, 11, 24] and minimum load k-facility location [1]. In the language of [1] and [11], the variants we study in this work can be stated as Connected Minimum Load Facility Location with Disjoint Clusters on Grid Graphs. Carlsson and Devulapalli study a continuous variant of our problem for different metrics [6].

4 Polyomino Cluster Assignment Problem

▶ **Lemma 4.1.** PCAP *is* NP*-hard for* $\sigma = 3$.

Proof. We reduce from Planar Monotone 3-SAT [9], i.e., planar 3-SAT where each clause consists of either only positive or only negative literals. Let $\mathcal{S}$ be such a 3-SAT instance with clauses $\mathcal{S}_i$. We show how to construct a polyomino P and a set $C \subseteq P$ that has a C-clustering of weight at most 3 if and only if $\mathcal{S}$ is satisfiable.

By [18], we can embed the planar graph for $\mathcal{S}$ such that all vertices lie at integer positions and all edges are either vertical or horizontal with blowup polynomial in $|\mathcal{S}|$. Based on this embedding, we build a polyomino by placing the gadgets for variables and clauses depicted in Figure 3 accordingly. All tiles marked with a square in Figure 3 are centers, i.e., elements of C. It remains to interconnect the constructed gadgets by wires where exactly every third tile is

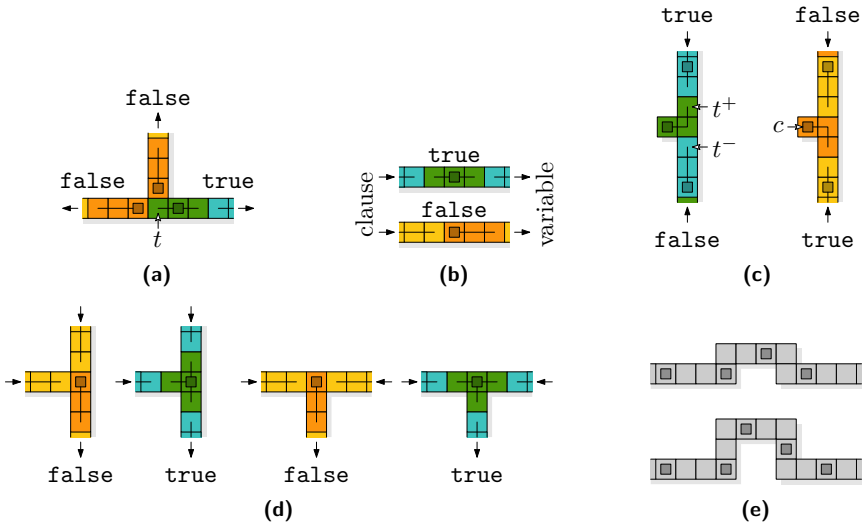


Figure 3 Gadgets: **(a)** positive clause gadget, **(b)** wire gadget from a positive clause, **(c)** variable gadgets, **(d)** splitter gadgets from positive clauses in both possible configurations, and **(e)** offset gadgets. Most gadgets are shown twice to indicate valid assignments depending on variable assignment. Gadgets from negative clauses can be constructed analogously with `true` and `false` flipped.

a center, see Figure 3(b). This construction is always possible (with enough spacing between the vertices) because we can introduce detours on straight wires to achieve every possible offset modulo 3 at the end of a wire, see Figure 3(e). If a literal occurs in multiple clauses, we place splitter gadgets along the wires, see Figure 3(d). For a full example, see Figure 2(a).

It remains to show that there exists a C-clustering of P with weight at most 3 exactly if there exists a satisfying assignment A to $\mathcal{S}$. We consider both directions separately. Given a satisfying assignment A to $\mathcal{S}$, we construct a C-clustering of weight 3 by assigning tiles as shown for every gadget in Figure 3; a complete instance is shown in Figure 2(b). Note that since there is at least one satisfying literal in every clause, at least one center in every clause gadget has enough *remaining capacity* to cover the gadget's t-tile (Figure 3(a)), where we define the remaining capacity of a center $c \in C$ as $\sigma - w(c)$.

Given a C-clustering with weight at most 3, we create a satisfying assignment A to $\mathcal{S}$ as follows. For every variable gadget, we check if tile t^+ in Figure 3(c) is assigned to the gadget's center and, exactly in this case, set the corresponding variable to `true`. We show that this assignment implies that every clause in $\mathcal{S}$ is satisfied.

By construction, the path from a clause gadget to its variable gadget follows the strict pattern of length-2 subpaths that are separated by single center tiles. We denote these subpaths as *segments*. Since segments are only connected to two centers, both of their tiles must be assigned to one of them. A segment is *mixed*, if the tile farthest away from the clause is assigned to the farther center. For example, the segments in the top wire in Figure 3(b) are mixed while those in the bottom are not. We observe that being mixed propagates to all further segments in the direction of the variable gadget: In the subsequent segment, the closer center already covers a tile of the previous segment, and therefore not all tiles of the current segment can be assigned to it, as this would exceed the center's remaining capacity. Therefore, the tile farthest away must again be assigned to the next center.

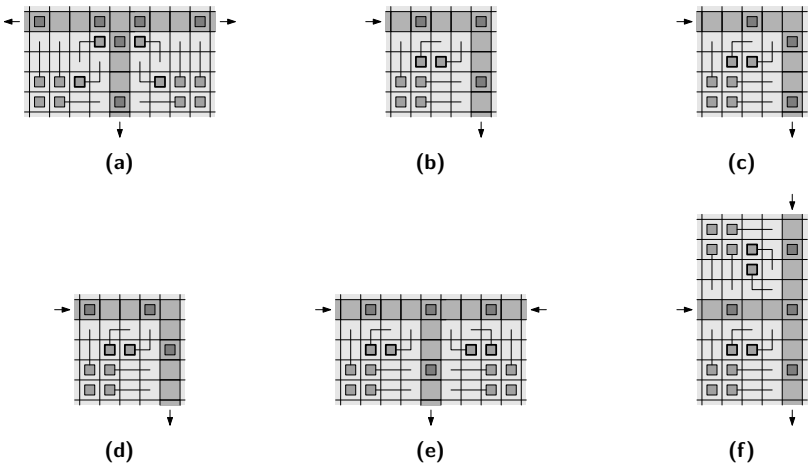


Figure 4 In convex corners, covering all newly-added tiles requires additional centers: **(a)** clause gadgets, **(b)**–**(d)** all cases of convex wire corners, and **(e)**–**(f)** splitters, up to rotation and reflection. For each setting, a valid assignment is indicated.

Now, consider a clause $\mathcal{S}_i$ of $\mathcal{S}$. In the corresponding clause gadget, the middle tile (labeled t) is assigned to one of the adjacent centers, which therefore does not have sufficient remaining capacity to be assigned all tiles on the following segment. Therefore, it and all further segments up to the respective variable gadget are mixed. Arriving in the variable gadget, there are two cases. If the corresponding literal was positive (1), the wire connects to the variable gadget from the top in Figure 3(c). Since the top center does not have sufficient remaining capacity to cover the t^+-tile, the only possibility is that t^+ gets assigned the left center (labeled c). Consequently, $\mathcal{S}_i$ is satisfied in A, because we set the corresponding variable to `true`. Analogously, if the corresponding literal was negative (2), the tile labeled t^- is assigned to the left center, and the remaining capacity does not suffice to cover t^+ as well. Thus, we set the corresponding variable to `false` and $\mathcal{S}_i$ is satisfied in A. ◀

By appropriately stretching the embedding and filling the holes in the construction, we can show that the problem remains NP-hard for $\sigma = 3$ if P is a rectangle, see Figure 5.

▶ **Theorem 4.2** ($\star$)**.** PCAP *is* NP*-hard for* $\sigma = 3$ *even if* P *is a rectangle.*

Proof sketch. We stretch the polyomino P constructed in Lemma 4.1 and fill all empty positions in the bounding box around P with tiles. We then place centers at every tile with a distance of at least three from P. While these centers are too far away to influence the gadgets and cover a majority of the newly added tiles, they cannot cover all tiles in convex corners formed by gadgets. Figure 4 shows the positions of additional centers in all such corners, paired with valid assignments to cover all tiles. See Figure 5 for a full construction. As these centers are close enough to P to influence its gadgets, we analyze all possible corners to ensure that the mixed property is still propagated from clause to variable gadgets. ◀

Consequently, the smallest feasible σ for a PCAP instance cannot be approximated with a factor better than $4/3$ unless P = NP, i.e., there is no PTAS for this variant.

The NP-hardness of PCAP in Lemma 4.1 also extends to triangle numbers. We define the triangle number $T(i)$ of $i \in \mathbb{N}$ as the sum of all natural numbers up to i, i.e., $T(i) := \sum_{k=1}^{i} k$.

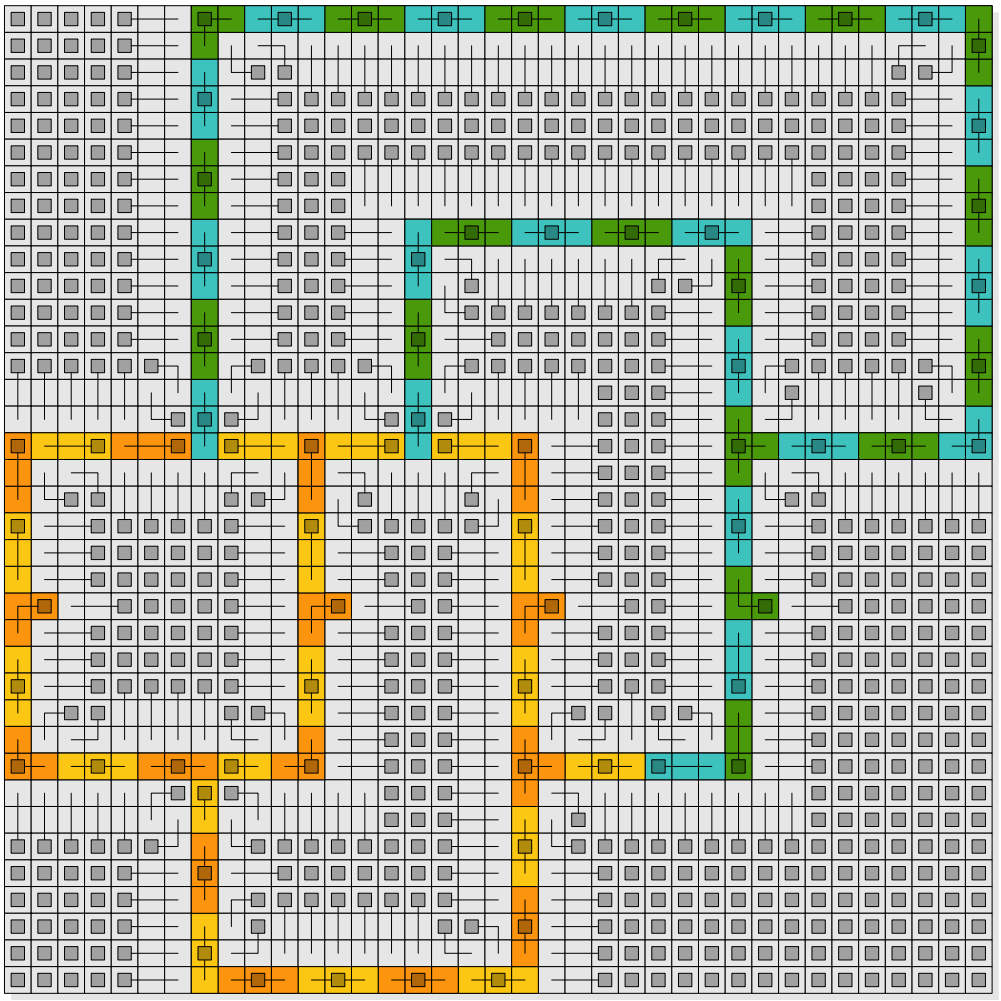


Figure 5 A rectangular PCAP instance with tile assignments for the same 3-SAT instance as in Figure 2. The colored tiles represent the embedded construction from Figure 2 (stretched), empty positions are filled with gray tiles. In the hole corners, the centers are placed as shown in Figure 4.

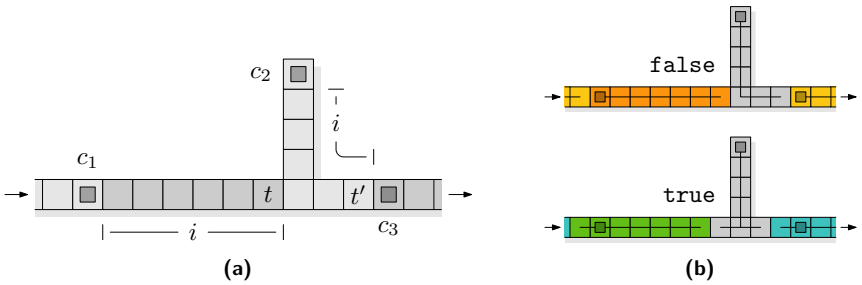


Figure 6 (a) The generalized offset gadget for $\sigma = T(i)$ is built from two paths of length i. The offset can be adapted by varying the number of tiles before the right path turns upward. **(b)** Valid assignments depend on the SAT variable assignments (shown for positive clauses).

▶ **Theorem 4.3** ($\star$). PCAP *is* NP*-hard for* $\sigma = T(i)$, $i \geq 2$.

Proof sketch. We construct a polyomino as described in Lemma 4.1, but stretch the segments to a length of i (instead of 2). As a center requires a remaining capacity of exactly $T(i)$ to cover all tiles of an adjacent segment, it cannot cover a single other tile from a different segment. Therefore, the property of being mixed is still propagated from clause gadgets to variable gadgets. The only gadget that needs to be adjusted is the offset gadget. To this end, we introduce a generalized offset gadget, see Figure 6. This gadget can be shown to equally propagate the mixed property along wires while introducing arbitrary offsets. ◀

We conjecture PCAP to be NP-hard for any constant $\sigma \geq 3$, but our proof technique cannot easily be extended beyond triangle numbers.

On the other hand, PCAP can be decided in polynomial time for $\sigma \leq 2$ with matchings.

▶ **Theorem 4.4.** PCAP *is decidable in polynomial time in the size of* G_P *for* $\sigma \leq 2$.

Proof. Let $\sigma \in \{1, 2\}$. In any valid solution, at most σ tiles can be assigned to a center $c \in C$ and they have to be adjacent to c. Otherwise, $w(c) \geq 3$. Therefore, we can restate the problem as a matching problem: For each center $c \in C$, we create up to two nodes $v_{c,1}, v_{c,\sigma}$, and for each other tile $t \in P \setminus C$, we create a node v_t. For $i \in \{1, \sigma\}$, we connect $v_{c,i}$ and v_t if the tiles of c and t are adjacent, yielding a bipartite graph G. If there is a matching in G into $V_{P\setminus C} = \{v_t \mid t \in P \setminus C\}$, i.e., any vertex in $V_{P\setminus C}$ is matched, then the PCAP instance is a yes-instance. As G has at most $2|P|$ nodes and $16|P|$ edges, any polynomial-time algorithm deciding whether there is a matching into $V_{P\setminus C}$ (e.g., Hopcroft-Karp [15]) decides PCAP. ◀

5 Polyomino Clustering Problem

▶ **Theorem 5.1.** PCP *is* NP*-hard for* $\sigma = 2$ *even if* G_P *has no vertex of degree four.*

Proof. It is NP-hard to partition a grid graph $G = (V, E)$ with maximum degree 3 into P_3 subgraphs (i.e., paths with three vertices) [22]. Because every connected grid graph is the dual graph of a polyomino, NP-hardness for PCP immediately follows by setting $k = |V|/3$ (if $|V|$ is not divisible by 3, G cannot be partitioned into P_3 subgraphs) and $\sigma = 2$ (the center will necessarily be the middle vertex of every P_3). ◀

As before, it follows that there is no PTAS for minimizing σ in PCP instances (with fixed k) and we cannot achieve an approximation factor better than $3/2$ unless P = NP.

▶ **Theorem 5.2.** PCP *is decidable in polynomial time in the size of* G_P *for* $\sigma = 1$.

Proof. Note that for each center c we have $|\mathcal{A}^{-1}(c)| \leq 2$, i.e., c itself and up to one other tile. Thus, we are searching for a maximum matching in G_P. Let M be a maximum matching in G_P, which can be found in time $O(|P|^{3/2})$ with the algorithm of Hopcroft-Karp [15]. Then, the PCP instance is a yes-instance if and only if $k \geq |M| + (|P| - 2|M|) = |P| - |M|$. ◀

6 Future Work

There are several directions for future work. On the theoretical side, most interesting is *any* constant-factor approximation algorithm; our results already indicate that a PTAS would imply $\mathsf{P} = \mathsf{NP}$. In our NP-hardness constructions, the number k of cluster centers is linear in the size of the configuration. In practical applications, k may be much smaller. As both PCAP and PCP bear resemblance to NP-hard partitioning problems even if k is constant, polynomial-time algorithms for arbitrary polyominoes seem unlikely, but the problems may be more tractable on restricted instance classes.

On the practical side, most relevant are exact methods that can solve instances of interesting size to provable optimality within reasonable time.

References

1. Sara Ahmadian, Babak Behsaz, Zachary Friggstad, Amin Jorati, Mohammad R. Salavatipour, and Chaitanya Swamy. Approximation algorithms for minimum-load k-facility location. *ACM Transactions on Algorithms*, 14(2):16:1–16:29, 2018. doi:10.1145/3173047.
2. Aaron T. Becker, Sándor P. Fekete, Phillip Keldenich, Dominik Krupke, Christian Rieck, Christian Scheffer, and Arne Schmidt. Tilt assembly: Algorithms for micro-factories that build objects with uniform external forces. *Algorithmica*, 82(2):165–187, 2020. doi:10.1007/S00453-018-0483-9.
3. Aaron T. Becker, Sándor P. Fekete, Jonas Friemel, Ramin Kosfeld, Peter Kramer, Harm Kube, Christian Rieck, Christian Scheffer, and Arne Schmidt. Efficient reconfiguration of tile arrangements by a single active robot. In *Canadian Conference on Computational Geometry (CCCG)*, pages 322–330, 2025. URL: https://cccg-wads-2025.eecs.yorku.ca/cccg-papers/7A3.pdf.
4. Mohamed Khalil Ben-Larbi, Kattia Flores Pozo, Tom Haylok, Mirue Choi, Benjamin Grzesik, Andreas Haas, Dominik Krupke, Harald Konstanski, Volker Schaus, Sándor P. Fekete, Christian Schurig, and Enrico Stoll. Towards the automated operations of large distributed satellite systems. Part 1: Review and paradigm shifts. *Advances in Space Research*, 67(11):3598–3619, 2021. doi:10.1016/j.asr.2020.08.009.
5. Pavel Berkhin. A survey of clustering data mining techniques. In *Grouping Multidimensional Data: Recent Advances in Clustering*, pages 25–71. Springer, 2006. doi:10.1007/3-540-28349-8_2.
6. John Gunnar Carlsson and Raghuveer Devulapalli. Dividing a territory among several facilities. *INFORMS Journal on Computing*, 25(4):730–742, 2013. doi:10.1287/ijoc.1120.0536.
7. Victor Chepoi, Tristan Fevat, Emmanuel Godard, and Yann Vaxès. A self-stabilizing algorithm for the median problem in partial rectangular grids and their relatives. *Algorithmica*, 62(1):146–168, 2012. doi:10.1007/s00453-010-9447-4.
8. Mark de Berg, Leyla Biabani, Morteza Monemizadeh, and Leonidas Theocharous. Clustering in polygonal domains. In *International Symposium on Algorithms and Computation (ISAAC)*, pages 23:1–23:15, 2023. doi:10.4230/LIPIcs.ISAAC.2023.23.

9 Mark de Berg and Amirali Khosravi. Optimal binary space partitions for segments in the plane. *International Journal of Computational Geometry & Applications*, 22(03):187–205, 2012. doi:10.1142/S0218195912500045.

10 Lukas Drexler, Jan Eube, Kelin Luo, Dorian Reineccius, Heiko Röglin, Melanie Schmidt, and Julian Wargalla. Connected k-center and k-diameter clustering. *Algorithmica*, 86(11):3425–3464, 2024. doi:10.1007/s00453-024-01266-9.

11 Jan Eube, Kelin Luo, Dorian Reineccius, Heiko Röglin, and Melanie Schmidt. Connected k-median with disjoint and non-disjoint clusters. In *European Symposium on Algorithms (ESA)*, pages 63:1–63:14, 2025. doi:10.4230/LIPIcs.ESA.2025.63.

12 Sándor P. Fekete. Coordinating swarms of objects at extreme dimensions. In *International Workshop on Combinatorial Algorithms (IWOCA)*, pages 3–13, 2020. doi:10.1007/978-3-030-48966-3_1.

13 Andreas Emil Feldmann and Luca Foschini. Balanced partitions of trees and applications. *Algorithmica*, 71(2):354–376, 2015. doi:10.1007/s00453-013-9802-3.

14 Javier Garcia, Jonas Friemel, Ramin Kosfeld, Michael Yannuzzi, Peter Kramer, Christian Rieck, Christian Scheffer, Arne Schmidt, Harm Kube, Dan Biediger, Sándor P. Fekete, and Aaron T. Becker. Moving matter: Using a single, simple robot to reconfigure a connected set of building blocks. In *International Conference on Automation Science and Engineering (CASE)*, pages 2260–2266, 2025. doi:10.1109/CASE58245.2025.11163739.

15 John E. Hopcroft and Richard M. Karp. An $n^{5/2}$ algorithm for maximum matchings in bipartite graphs. *SIAM Journal on Computing*, 2(4):225–231, 1973. doi:10.1137/0202019.

16 Anil K. Jain and Richard C. Dubes. *Algorithms for clustering data*. Prentice-Hall, Inc., USA, 1988.

17 Benjamin Jenett, Amira Abdel-Rahman, Kenneth Cheung, and Neil Gershenfeld. Material–robot system for assembly of discrete cellular structures. *IEEE Robotics and Automation Letters*, 4(4):4019–4026, 2019. doi:10.1109/LRA.2019.2930486.

18 Donald E. Knuth and Arvind Raghunathan. The problem of compatible representatives. *SIAM Journal on Discrete Mathematics*, 5(3):422–427, 1992. doi:10.1137/0405033.

19 Eunjin Oh, Sang Won Bae, and Hee-Kap Ahn. Computing a geodesic two-center of points in a simple polygon. *Computational Geometry*, 82:45–59, 2019. doi:10.1016/j.comgeo.2019.05.001.

20 Eunjin Oh, Jean-Lou De Carufel, and Hee-Kap Ahn. The geodesic 2-center problem in a simple polygon. *Computational Geometry*, 74:21–37, 2018. doi:10.1016/j.comgeo.2018.02.008.

21 Josh Reese. Solution methods for the p-median problem: An annotated bibliography. *Networks*, 48(3):125–142, 2006. doi:10.1002/net.20128.

22 René van Bevern, Robert Bredereck, Laurent Bulteau, Jiehua Chen, Vincent Froese, Rolf Niedermeier, and Gerhard J. Woeginger. Partitioning perfect graphs into stars. *Journal of Graph Theory*, 85(2):297–335, 2017. doi:10.1002/jgt.22062.

23 Michael Yannuzzi. Personal communication: Figure provided by e-mail, March 2026.

24 William Chung-Kung Yen and Chien-Tsai Chen. The p-center problem with connectivity constraint. *Applied Mathematical Sciences*, 1(27):1311–1324, 2007. URL: https://www.m-hikari.com/ams/ams-password-2007/ams-password25-28-2007/yenAMS25-28-2007.pdf.

25 Yangsheng Zhu, Mingjing Qi, Zhiwei Liu, Jianmei Huang, Dawei Huang, Xiaojun Yan, and Liwei Lin. A 5-mm untethered crawling robot via self-excited electrostatic vibration. *IEEE Transactions on Robotics*, 38(2):719–730, 2022. doi:10.1109/TRO.2021.3088053.

Flipping Non-crossing Spanning Trees is NP-Hard*

Håvard Bakke Bjerkevik[1], Joseph Dorfer[2], Linda Kleist[3], Torsten Ueckerdt[4], and Birgit Vogtenhuber[5]

1 **University at Albany**
`hbjerkevik@albany.edu`
2 **Graz University of Technology**
`joseph.dorfer@tugraz.at`
3 **Universität Hamburg**
`linda.kleist@uni-hamburg.de`
4 **Karlsruhe Institute of Technology**
`torsten.ueckerdt@kit.edu`
5 **Graz University of Technology**
`birgit.vogtenhuber@tugraz.at`

Abstract

A flip sequence between two plane spanning trees on a point set P is a sequence of flips (that is, single edge exchanges) such that each intermediate structure is again a plane spanning tree on P. We prove that finding a shortest flip sequence between two non-crossing spanning trees is NP-hard, already on point sets in convex position. The result still holds in the case when we restrict the flips to be compatible flips (where the removed and added edge do not cross) or rotations (where the removed and added edge share a vertex). Our proof relies on a tool called *conflict graphs* for pairs of non-crossing spanning trees, which was first used by Bjerkevik, Kleist, Ueckerdt, and Vogtenhuber [SODA 2025] for the study of flip sequences between such trees.

1 Introduction

Let P denote a set of n points in the plane in general position, that is, no three points of P are collinear. A *non-crossing spanning tree on* P is a spanning tree that has P as its vertex set and whose edges are pairwise non-crossing straight-line segments. A *flip* in a non-crossing spanning tree T is the exchange of an edge e_1 of T by a different edge e_2 such that the resulting graph $T' = (T - e_1) \cup e_2$ is again a non-crossing spanning tree; see Figure 1 for an example. The *flip graph* $\mathcal{F}(P)$ has a vertex for each non-crossing spanning tree on P and an edge between two trees whenever they can be transformed into each other by a single flip. For two trees T and T' on P, we denote their *flip distance*, the length of a shortest path between T and T' in $\mathcal{F}(P)$, by $\mathrm{dist}(T, T')$. The flip graph $\mathcal{F}(P)$ is connected for any n-point set P in general position, and its radius is bounded from above by $n - 2$ (for $n \geq 2$), implying that its diameter $\mathrm{diam}(\mathcal{F}(P))$ is bounded from above by $2n - 4$ [5].

It further holds that $\mathrm{diam}(\mathcal{F}(P)) \geq \lfloor 3/2 \cdot n \rfloor - 5$ [12]. Since the appearance of those two benchmark results, the flip graph $\mathcal{F}(P)$ of non-crossing geometric spanning trees on point sets P has been subject of intensive study, yielding several results for the special case when P is in convex position. The upper bound was improved to $2n - O(\log(n))$ [1] and $2n - \sqrt{n}$ [9]. The leading constant of 2 was broken, when an upper bound of $1.95n$ was provided [8].

* This research was funded in part by the Austrian Science Fund (FWF) 10.55776/DOC183 and by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – 520708409

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

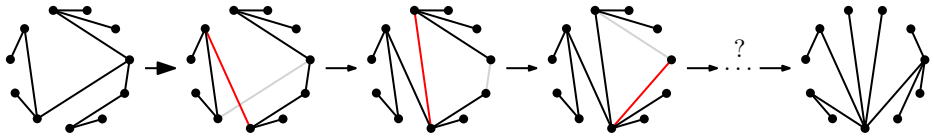


Figure 1 Reconfiguration of non-crossing spanning trees on convex point sets by edge exchange. The appearing edge is highlighted in red and the disappearing edge in gray.

Very recently, the currently best known bounds of $^{14}/_{9} \cdot n + O(1) \leq \operatorname{diam}(\mathcal{F}(P)) \leq {}^{15}/_{9} \cdot n - O(1)$ for the diameter of the flip graph on non-crossing spanning trees on point sets in convex position have been shown [7]. The crucial tool for obtaining these bounds were *conflict graphs* for pairs of non-crossing spanning trees.

We show a further application of conflict graphs: Proving NP-hardness of computing the shortest flip sequence between two non-crossing spanning trees, even for convex point sets.

▸ **Theorem 1.1.** *The decision problem of whether there exists a flip sequence of length at most d between two non-crossing spanning trees is* NP*-complete.*

Interestingly, this result still holds for more constrained flip operations, concretely, *compatible* flips (where the removed and the added edge do not cross) and *rotations* (where the removed and the added edge share an endpoint). In Figure 1, while the first flip is just an edge exchange, the second flip is a compatible flip, and the third flip is a rotation.

▸ **Theorem 1.2.** *The decision problem of whether there exists a compatible flip sequence / a rotation sequence of length at most d between two non-crossing spanning trees is* NP*-complete.*

We put Theorem 1.2 into relation with the *happy edge property*, which states that there always exists a shortest flip sequence that does not remove edges that lie in both the initial and the target configuration. For point sets in general position it is NP-hard to compute the flip distance between two triangulations [15, 16], perfect matchings [6] and odd matchings [2]. In all three settings, the happy edge property does not hold. For perfect matchings in convex point sets the happy edge property holds and the flip distance is linear-time computable [13]. These results support the belief that the happy edge property is tied to the complexity of graph reconfiguration problems: If the happy edge property holds, then the problem is supposedly easy to solve. If not, the problem may be hard to solve. This pattern was broken when a linear-time algorithm for the flip distance of two plane spanning paths on a convex point set was provided, despite the absence of the happy edge property [3]. We break the pattern the other way: The happy edge property holds for compatible flips in non-crossing spanning trees on convex point sets [4], yet flip distances are NP-hard to compute.

Further, our NP-hardness result for non-crossing spanning trees marks the, to the best of our knowledge, first NP-hardness result for a graph reconfiguration problem on a convex point set, as surveyed in Table 1. (The result on triangulations was obtained after we submitted our work to EuroCG and the preprint [11] has recently been uploaded to arXiv.)

2 Fundamental concepts: Edge pairs, conflict graphs, and blowups

We introduce some concepts that were applied or developed in [7].

Linear Representation. A *linear representation* of a non-crossing tree T on a convex point set is constructed by cutting open the circular ordering of points, unfolding it into a horizontal

Table 1 Overview of complexity results. Results obtained in this work are marked by [*].

setting	flip type	convex position	general position
triangulations	edge exchange	NP-hard [11]	NP-hard [15, 16]
non-crossing spanning trees	edge exchange	NP-hard [*]	NP-hard [*]
non-crossing spanning trees	compatible	NP-hard [*]	NP-hard [*]
non-crossing spanning trees	rotation	NP-hard [*]	NP-hard [*]
non-crossing spanning trees	slide	open	open
non-crossing spanning paths	edge exchange	in P [3]	open
non-crossing perfect matchings	edge exchange	in P [14]	NP-hard [6]
non-crossing odd matchings	edge exchange	open	NP-hard [2]

line segment, called spine, see also Figure 2. Edges in T are illustrated by semi-circles above the spine connecting their endpoints. Given two trees (for example the initial tree and the target tree of a flip sequence), in the same linear representation, it is convenient to represent the edges of one tree above the spine and the edges of the other tree below the spine. By labeling the vertices along the linear representation representation by $p_1, \dots, p_n$, we get a notion of the *length* of an edge p_ip_j defined by $|i-j|$. We say a point p_k is *covered* by an edge p_ip_j with $i < j$ if $i \le k \le j$. An edge *covers* another edge if it covers both of its endpoints.

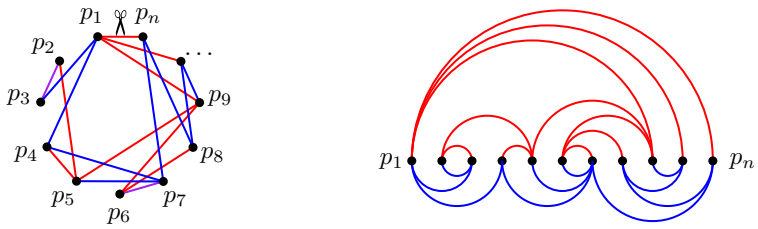


Figure 2 Figure reproduced from [7]. Two non-crossing trees T, T' on a circularly labeled point set in convex position and its linear representation with T above and T' below the spine.

Gaps and Edge Pairing. A *gap* in a linear representation is a set $g_i = \{p_i, p_{i+1}\}$ of two consecutive points on the spine. For every tree T, there is a natural *gap-edge bijection* between gaps and edges of T: Each gap $g_i = \{p_i, p_{i+1}\}$ is assigned to the shortest edge e_i that covers p_i and p_{i+1}. This assignment is indeed a bijection by Lemma 3.1 in [7]. For an example, consider Figure 3. Based on the assigned gap, edges of T are partitioned into three groups. In particular, an edge is *short* if it shares both vertices with its gap, *near* if it shares one vertex with its gap, and *long* if shares no vertex with its gap.

For two trees T and T', we obtain an *edge-edge bijection* by pairing the edge $e_i \in T$ with the edge $e'_i \in T'$ that is assigned to the same gap $g_i = \{p_i, p_{i+1}\}$. The authors of [7] explain how to handle nine types of edge pairs. For our results it is sufficient to concentrate on pairs (e_i, e'_i) with two distinct near edges, so-called *near-near pairs*; see Fig. 3 for an illustration.

Conflict Graphs. *Conflict graphs* for pairs of non-crossing spanning trees were applied as a tool to find a largest set of near-near pairs (e_i, e'_i) where the initial edge $e_i \in T$ can be flipped directly to the target edge $e'_i \in T'$. The conflict graph H has a vertex for each (gap associated to a) near-near pair and a directed edge from a pair (e_i, e'_i) to (e_j, e'_j) if the edge e_i needs to be removed before the flip that removes e_j and adds e'_j can be made. There exist

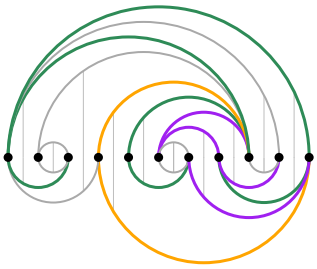


Figure 3 Illustration of the gap-edge and edge-edge bijection. Near-near pairs are colored.

three types of conflicts. Specifically, there is a directed edge $\overrightarrow{g_i g_j}$ in $E(H)$ from g_i to g_j if **type 1** e_i crosses e'_j, see Figure 4**(a)**; **type 2** e'_j covers e_i and e_i covers g_j, see Figure 4**(b)**; or **type 3** e_i covers e'_j and e'_j covers g_i, see Figure 4**(c)**.

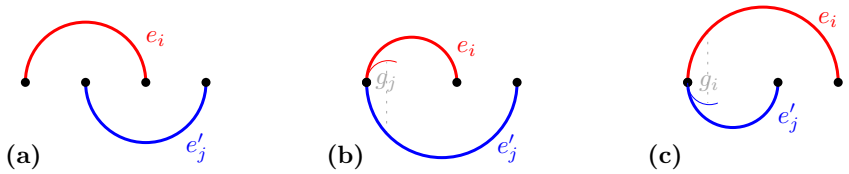


Figure 4 Figure reproduced from [7]. Examples of conflicts: **(a)** type 1, **(b)** type 2, **(c)** type 3.

We slightly abuse terminology and view the vertices of H interchangeably as gaps and as near-near pairs. Let $\mathrm{ac}(H)$ denote the largest number of vertices in H which induce an acyclic subgraph. The authors of [7] derive the following relations.

▸ **Theorem 2.1** ([7, Theorem 2.1]). *Let T, T' be two non-crossing trees on linearly ordered points $p_1, \ldots, p_n$ with corresponding conflict graph $H = H(T, T')$.*

(i) If $V(H)$ is non-empty, then $\mathrm{dist}(T, T') \le \max\left\{\frac{3}{2}, 2 - \frac{\mathrm{ac}(H)}{|V(H)|}\right\}(n-1)$. *If $V(H)$ is empty, then* $\mathrm{dist}(T, T') \le \frac{3}{2}(n-1)$.

(ii) If $V(H)$ is non-empty, then there is a constant c depending only on T and T' such that for all $\bar{n} \ge 1$, we have $\mathrm{diam}(\mathcal{F}_{\bar{n}}) \ge \left(2 - \frac{\mathrm{ac}(H)}{|V(H)|}\right)\bar{n} - c$.

Blowups. Part (ii) of Theorem 2.1 is obtained by considering *blowups*. For trees T, T' and integer $\beta > 0$, the β-blowup of (T, T') is a pair $(\beta \cdot T, \beta \cdot T')$ of trees obtained by the following construction. For every gap g that corresponds to a near-near pair (e, e'), insert a set $V(e) = V(e')$ of β additional points in the gap g. In $\beta \cdot T$ add an edge from each $v \in V(e)$ to the endpoint of e that is not adjacent to g. Proceed similarly in $\beta \cdot T'$ for e'. For large values of β, the flip distance between two blowups of non-crossing spanning trees βT and $\beta T'$ is determined almost exclusively by the cardinality of a largest acyclic subset of the conflict graph of the original trees T, T' and the value β. A lower bound on the flip distance between βT and $\beta T'$ is given in Lemma 2.2.

▸ **Lemma 2.2** ([7, Lemma 23, arXiv-Version]). *The flip distance from $\beta \cdot T$ to $\beta \cdot T'$ is at least $(\beta - 2n)(2|V(H)| - \mathrm{ac}(H))$, where $H = H(T, T')$.*

3 NP-Completeness results

In this section, we show Theorem 1.1, which we do via an intermediate problem: We sketch the proof that it is NP-complete to decide whether a directed graph H has an acyclic subset of size at least k, even if H is the conflict graph of two non-crossing trees with a given linear representation (full proof in the full version of the paper).

▸ **Proposition 3.1.** *Given two non-crossing trees T, T' with a linear representation and corresponding conflict graph $H = H(T, T')$, as well as an integer k, it is* NP*-complete to decide whether* $\mathrm{ac}(H) \geq k$.

Sketch of the proof. We reduce from the NP-complete problem Planar Max-2SAT [10] with the variable-clause incidence graph G_ϕ drawn with variables on a horizontal line and clauses either above or below the line. For a fixed 2SAT formula ϕ and an integer t, we construct T, T' with a linear representation and an integer k (that depends on ϕ and t) such that the following are equivalent:

- The conflict graph $H = H(T, T')$ admits an acyclic subset of size at least k.
- The 2SAT-formula ϕ admits a truth assignment that satisfies at least t clauses.

To construct T and T', we model the variables and clauses in ϕ with small gadgets.

Let $x \in X$ be a variable and $d = \deg(x)$ be its degree in G_ϕ, i.e., x occurs in exactly d clauses. The variable-gadget for x consists of $8d + 1$ pairs of edges, of which $6d$ are near-near pairs and hence their corresponding $6d$ gaps are vertices in the conflict graph $H = H(T, T')$. The gadget and the conflict graph are illustrated in Figure 5. The $6d$ gaps form a path P of double conflicts, i.e., every acyclic subset is an independent set in P. We partition P into two independent sets $t(x)$ and $f(x)$ by alternating the gaps in $t(x)$ and $f(x)$ along P. Later we ensure that every acyclic subset S of H with $|S| \geq k$ either contains all of $t(x)$ or all of $f(x)$, which encodes the two possible truth assignments True and False for variable x.

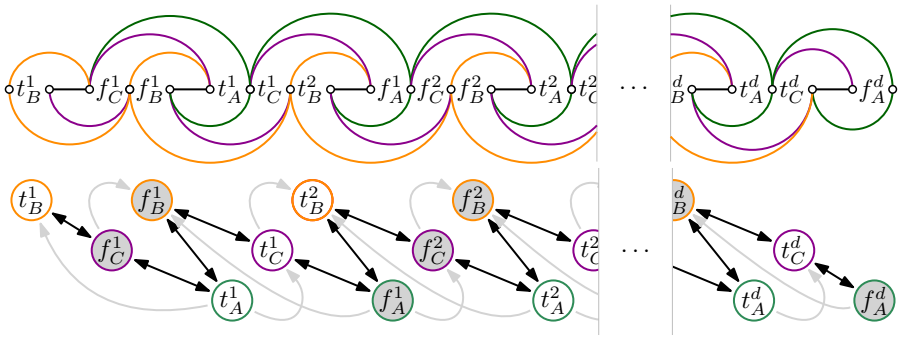


Figure 5 Illustration of a variable-gadget and the graph of its mixed conflicts.

We place each variable-gadget at the position of the corresponding variable in the drawing of G_ϕ in such a way that the rightmost point of any gadget coincides with the leftmost point of the next gadget (immediately to the right). Secondly, for each clause C with incident variables x and y we introduce a new gap g_x in the variable-gadget of x and a new gap g_y in the variable-gadget of y. For each $z \in \{x, y\}$, the edge pair for g_z has an edge of length 3 in the variable-gadget of z and an edge connecting the gadgets of x and y. See Figures 6 and 7 for an illustration. To ensure that different clause-gadgets do not cross, long edges are put in the halfplane that contains the clause C in the aligned drawing of G_ϕ. Moreover, the

position of g_z on the path P of length $6\deg(z)$ in the variable-gadget of z is chosen with respect to the order of incident edges at z in the drawing of G_ϕ.

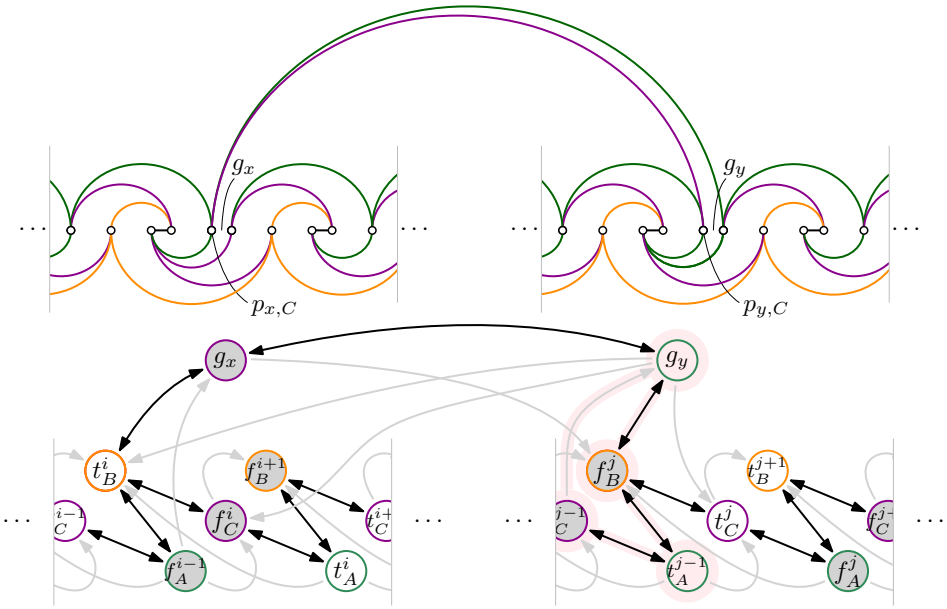


Figure 6 A clause-gadget for a clause above the horizontal line and its conflict graph.

The gap g_z and its corresponding edge pair are chosen such that there is a double conflict with a gap in $t(z)$ if z appears negated in C, and a double conflict with a gap in $f(z)$ if z appears non-negated in C. This ensures that an acyclic set S contains the gap g_z only if the truth assignment of variable z satisfies the clause C. A double conflict between the two gaps g_x, g_y of the clause-gadget ensures that at most one of g_x, g_y is in any acyclic set S. Given a truth assignment, we choose all of $t(x)$ when x is True, $f(x)$ when x is False, and one gap g_z in each satisfied clause. This gives an acyclic subset S of $H(T, T')$ with the desired size, which we discuss in more detail in a full version of the paper.

This way, every truth assignment satisfying t clauses of ϕ corresponds to an acyclic set $S \subseteq V(H)$ containing exactly half of all variable-gaps and exactly t of all clause-gaps. Finally, for the other direction, we need to ensure that for every variable x we have $t(x) \subseteq S$ or $f(x) \subseteq S$, as long as S is large enough. To this end, let $m = |\mathcal{C}|$. For each variable-gadget we apply an m-blowup to the first and last gap and a $2m$-blowup to all other variable-gaps. We apply no blowup to the clause-gaps, see Figure 8. Setting $k = m(12m + 6 - |X|) + t$ concludes our reduction from Planar Max-2SAT and thus the proof of Proposition 3.1. ◂

In order to finally prove Theorem 1.1, we now reduce the problem of finding large acyclic subsets in conflict graphs to the problem of finding short flip sequences in β-blowups.

▸ **Theorem 3.2.** *Let T and T' be two non-crossing spanning trees on n vertices with a linear representation, and k be an integer. Let $\beta = 4n^2 + 2n$. Then the following are equivalent.*

- *The conflict graph $H = H(T, T')$ has an acyclic subset of size at least k.*
- *The flip distance between $\beta \cdot T$ and $\beta \cdot T'$ is at most $(\beta + 1)(2|V(H)| - k) + 2(n - |V(H)|)$.*

Proof. For convenience, let us denote $|H| = |V(H)|$. Assume H does not contain an acyclic

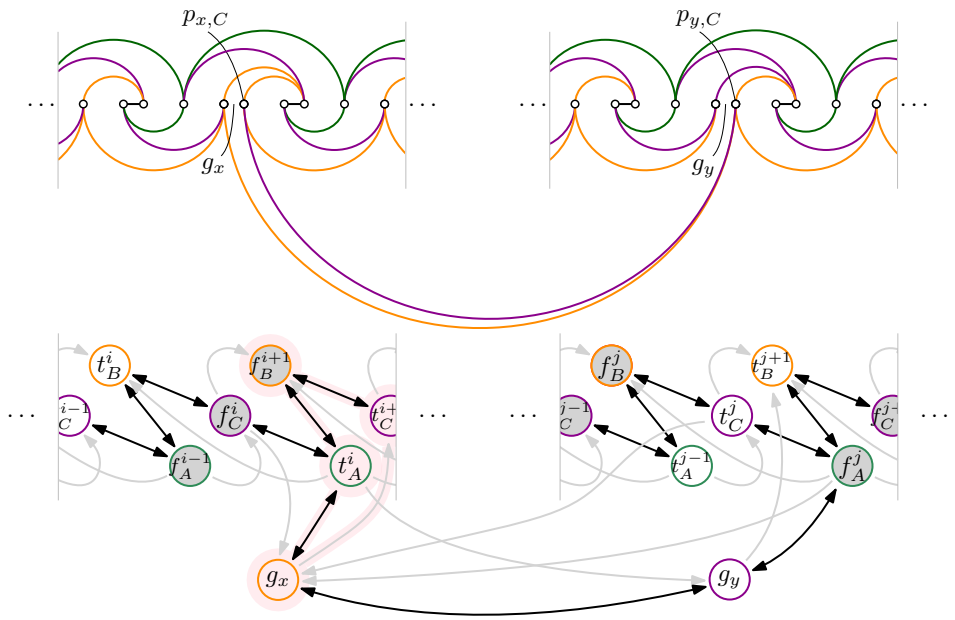


Figure 7 A clause-gadget for a clause below the horizontal line and its conflict graph.

set of size k, i.e., $\mathrm{ac}(H) < k$. Then by Lemma 2.2

$$\begin{aligned}\mathrm{dist}(\beta\cdot T, \beta\cdot T') &\geq (\beta-2n)(2|H|-\mathrm{ac}(H)) \geq (\beta+1-(2n+1))(2|H|-k+1)\\ &= (\beta+1)(2|H|-k)+(\beta+1)-(2n+1)(2|H|-k+1)\\ &> (\beta+1)(2|H|-k)+2(n-|H|).\end{aligned}$$

For the last inequality observe that $|H| < n$ and hence $(2n+1)(2|H|-k+1) \leq (2n+1)(2n-1) \leq 4n^2$, which together with $\beta = 4n^2+2n > 4n^2+2(n-|H|)$ gives the desired lower bound.

Conversely, assume $H = H(T,T')$ has an acyclic subset of size k, i.e., $\mathrm{ac}(H) \geq k$. Then, the conflict graph $H^\beta = H(\beta\cdot T, \beta\cdot T')$ of the blowup has an acyclic set of size $k\cdot(\beta+1)$, i.e., $\mathrm{ac}(H^\beta) \geq k(\beta+1)$. For the trees T, T' we have n vertices and $n-1$ gaps, of which $|H|$ are affected by the blowup. Thus, each of $\beta\cdot T$ and $\beta\cdot T'$ has $n+\beta|H|$ vertices. Moreover, $|V(H^\beta)| =$

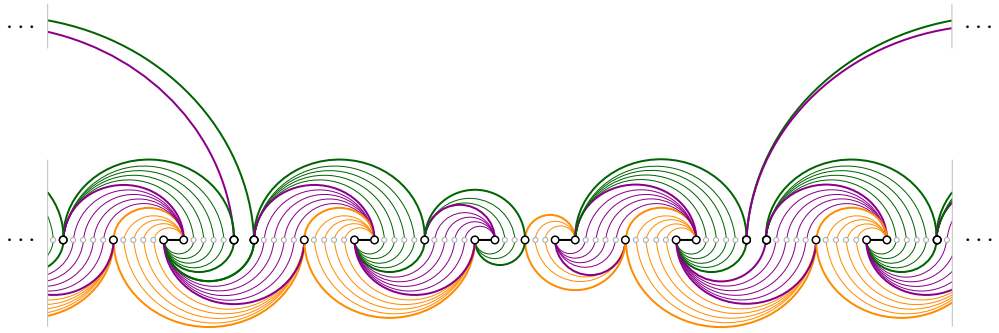


Figure 8 Applying an m-blowup (here $m = 2$) to the first and last gap of each variable-gadget and $2m$-blowups to all other gaps in variable-gadgets.

$(\beta+1)|H|$. By applying Theorem 2.1(i) the proof follows from the following calculation.

$$\begin{aligned}\operatorname{dist}(\beta\cdot T,\beta\cdot T') &\le \left(2-\frac{\operatorname{ac}(H^\beta)}{|V(H^\beta)|}\right)(n+\beta|H|-1)\\ &< \left(2-\frac{k(\beta+1)}{(\beta+1)|H|}\right)(n+\beta|H|) = \frac{2|H|-k}{|H|}(n+\beta|H|)\\ &= (\beta+1)(2|H|-k)+\frac{2|H|-k}{|H|}(n-|H|) < (\beta+1)(2|H|-k)+2(n-|H|)\end{aligned}$$ ◂

In the full version of the paper we show how to adapt the proof of Theorem 3.2 for compatible flips and rotations, which then proves Theorem 1.2.

References

1 Oswin Aichholzer, Brad Ballinger, Therese Biedl, Mirela Damian, Erik D. Demaine, Matias Korman, Anna Lubiw, Jayson Lynch, Josef Tkadlec, and Yushi Uno. Reconfiguration of non-crossing spanning trees, 2022. doi:10.48550/arXiv.2206.03879.

2 Oswin Aichholzer, Sofia Brenner, Joseph Dorfer, Hung P. Hoang, Daniel Perz, Christian Rieck, and Francesco Verciani. Flipping odd matchings in geometric and combinatorial settings. In *Symposium on Graph Drawing and Network Visualization (GD)*, pages 12:1–12:18, 2025. doi:10.4230/LIPIcs.GD.2025.12.

3 Oswin Aichholzer and Joseph Dorfer. A linear time algorithm for the flip distance between plane spanning paths in convex point sets. *Computing in Geometry and Topology*, 5(3):2–1, 2026.

4 Oswin Aichholzer, Joseph Dorfer, and Birgit Vogtenhuber. Constrained flips in plane spanning trees. In *Symposium on Graph Drawing and Network Visualization (GD)*, pages 5:1–5:18, 2025. doi:10.4230/LIPIcs.GD.2025.5.

5 David Avis and Komei Fukuda. Reverse search for enumeration. *Discrete Applied Mathematics*, 65(1-3):21–46, 1996. doi:10.1016/0166-218X(95)00026-N.

6 Carla Binucci, Fabrizio Montecchiani, Daniel Perz, and Alessandra Tappini. Flipping matchings is hard. *Arxiv preprint*, 2025. doi:10.48550/ARXIV.2503.02842.

7 Håvard Bakke Bjerkevik, Linda Kleist, Torsten Ueckerdt, and Birgit Vogtenhuber. Flipping non-crossing spanning trees. *Proceedings of the 2025 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 2313–2325. arXiv:2410.23809, doi:10.1137/1.9781611978322.77.

8 Nicolas Bousquet, Lucas de Meyer, Théo Pierron, and Alexandra Wesolek. Reconfiguration of plane trees in convex geometric graphs. In *International Symposium on Computational Geometry (SoCG 2024)*, volume 293 of *LIPIcs*, pages 22:1–22:17, 2024. arXiv:2310.18518, doi:10.4230/LIPIcs.SoCG.2024.22.

9 Nicolas Bousquet, Valentin Gledel, Jonathan Narboni, and Théo Pierron. A note on the flip distance between non-crossing spanning trees. *Computing in Geometry and Topology*, 2(1):8:1–8:7, 2023. arXiv:2303.07710, doi:10.57717/cgt.v2i1.36.

10 Kevin Buchin, Valentin Polishchuk, Leonid Sedov, and Roman Voronov. Geometric secluded paths and planar satisfiability. In *36th International Symposium on Computational Geometry (SoCG 2020)*, pages 24–1. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2020.

11 Joseph Dorfer. Flip distance of triangulations of convex polygons/rotation distance of binary trees is NP-complete. *arXiv preprint arXiv:2602.22874*, 2026.

12 Carmen Hernando, Ferran Hurtado, Alberto Márquez, Mercè Mora, and Marc Noy. Geometric tree graphs of points in convex position. *Discrete Applied Mathematics*, 93(1):51–66, 1999. doi:10.1016/S0166-218X(99)00006-2.

13 Carmen Hernando, Ferran Hurtado, and Marc Noy. Graphs of non-crossing perfect matchings. *Graphs and Combinatorics*, 18(3):517–532, 2002. doi:10.1007/S003730200038.

14 M. Carmen Hernando, Ferran Hurtado, and Marc Noy. Graphs of non-crossing perfect matchings. *Graphs and Combinatorics*, 18(3):517–532, 2002. doi:10.1007/S003730200038.

15 Anna Lubiw and Vinayak Pathak. Flip distance between two triangulations of a point set is NP-complete. *Computational Geometry: Theory & Applications*, 49:17–23, 2015. doi:10.1016/J.COMGEO.2014.11.001.

16 Alexander Pilz. Flip distance between triangulations of a planar point set is APX-hard. *Computational Geometry*, 47(5):589–604, 2014. doi:10.1016/j.comgeo.2014.01.001.

The Chromatic Dispersive Art Gallery Problem in Polyominoes*

Anna Brötzner[1], Kien C. Huynh[2], Christiane Schmidt[2], and Frederick Stock[3]

1 **Faculty of Technology and Society, Malmö University, Sweden**
anna.brotzner@mau.se
2 **Department of Science and Technology, Linköping University, Sweden**
{chi.kien.huynh, christiane.schmidt}@liu.se
3 **Miner School of Computer and Information Sciences, University of Massachusetts Lowell, USA**
frederick_stock@student.uml.edu

Abstract

The chromatic dispersive art gallery problem (CDAGP) asks for a guard cover in which guards of the same color have a distance of at least d. We consider the CDAGP for two colors in polyominoes. We provide an algorithm to compute a cover for polyominoes with and without holes when $d = 3$, and show that the problem is NP-complete in polyominoes with holes for $d = 5$.

1 Introduction

In the classical Art Gallery Problem (AGP) [1, 2, 4, 9, 11], we ask for the minimum number of (stationary) guards that are needed to monitor a given polygon P. Various other objectives have been studied for the AGP, for example, in the dispersive and the chromatic variants, we are not interested in the number of guards, but in guard sets that abide by a restriction on the relationship between individual guards. In the Dispersive Art Gallery Problem (DAGP) introduced by Rieck and Scheffer [14], we aim to maximize the geodesic distance between any pair of guards—or, in the decision version, for a given distance d, we aim to find a guard set with pairwise minimum geodesic distance at least d (see also [8, 12]). Assigning a color to each guard such that no two guards of the same color have overlapping visibility polygons yields a colored guard cover. In the Chromatic AGP (CAGP), introduced by Erickson and LaValle [5, 7, 6], we ask for a colored guard cover with the minimum number of colors.

Recently, Iwamoto and Takahashi [10] introduced a problem taking both aspects into account: the Chromatic Dispersive Art Gallery Problem (CDAGP). The CDAGP asks for a guard cover in which guards of the same color are placed as far away from each other as possible. That is, in contrast to the CAGP, the visibility polygons of guards of one color may overlap in the CDAGP. Instead, we enforce that guards of the same color have a high pairwise minimum geodesic distance (in the decision version, again of at least a bound d), while guards of different colors may be located arbitrarily close to each other. Iwamoto and Takahashi considered the problem in polyominoes under r-visiblity (two points p and q in a polygon P are r-visible, if the axis-aligned rectangle spanned by p and q is fully

* A. B. and C. S. were supported by grant 2021-03810 (Illuminate: provably good algorithms for guarding problems) from the Swedish Research Council (Vetenskapsrådet). F. S. was supported by NSF grant CCF-2348067.

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

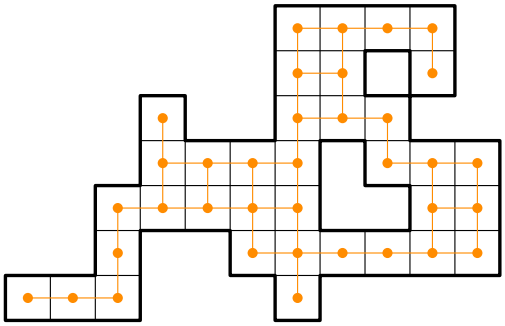


Figure 1 A polyomino induced by a grid graph G (orange).

contained in P) placing guards in the center of polyomino pixels. In fact, any cover in the r-visibility model is also a cover in the *all-or-nothing* model introduced by Biedl et al. [3], in which any pixel p is visible to a point guard g only if the guard sees all of p. Iwamoto and Takahashi showed that the CDAGP for two colors in polyominoes with holes is NP-complete for $d = 6$—we refer to this bichromatic version of the CDAGP as the 2-CDAGP. In this paper, we improve this to $d = 5$, and propose an algorithm to place guards for $d = 3$.

2 Preliminaries

A *polyomino* is a shape formed by joining multiple unit squares along their edges. For a given polyomino P, we call each square comprising P a *pixel*. Let G be a connected finite induced subgraph of the grid graph. We consider polyominoes that are duals of G, i.e. each vertex of G represents a pixel and pixels have a common edge if their vertices share an edge in G. Moreover, two pixels may touch each other at a corner without having a common edge. See Figure 1 for examples. G partitions the plane into faces, and every face of G that is bounded by a cycle of length > 4 corresponds to a hole in the polyomino. We say that a polyomino is *simple* if it has no holes.

3 An Algorithm for Distance $d = 3$

We present a worst-case optimal algorithm to cover simple polyominoes and polyominoes with holes.

▶ **Theorem 3.1.** *2-CDAGP in polyominoes can be solved in polynomial time for $d = 3$.*

The algorithm has two steps:

1. We first consider a fixed set of guards G_Φ that is a subset of a particular guard pattern in $\mathbb{Z}^2$ and satisfies the distance lower bound.
2. Let $P_{\overline{\Phi}}$ be the subset of pixels of P not seen by guards in G_Φ. We claim that for each connected component $S \subset P_{\overline{\Phi}}$, we can find a set of guards to guard S that also satisfies $d = 3$.

Let $(x_g, y_g) \in \mathbb{Z}^2$ be the integer coordinates of a guard. Consider the infinite set of guards Φ such that $x_g + y_g \equiv 0 \pmod 3$, y_g is even for red guards and y_g is odd for blue guards, note we assume that P lies in the first quadrant (i.e. all x and y coordinates are positive). Notice that Φ satisfies the minimum distance requirement of $d = 3$ for both colors. $G_\Phi = \Phi \cap P$ will be the set of guards used in Step (1). See Figure 2 for Φ and G_Φ. The set $P_{\overline{\Phi}} \subseteq P$ that

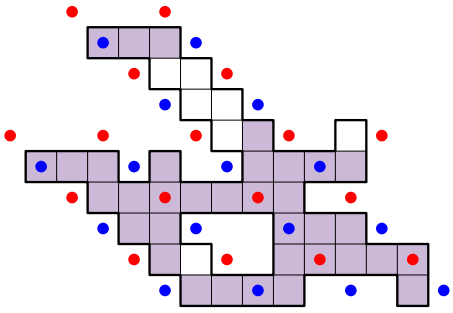


Figure 2 The set of guards G_Φ in step (1) of our algorithm is a subset of the infinite set $\Phi \subset \mathbb{Z}^2$. The subset of P that guards in G_Φ collectively see is shaded in purple. The white portions in the polyomino are not seen by guards in G_Φ; each such component is a subset of some staircase σ_c.

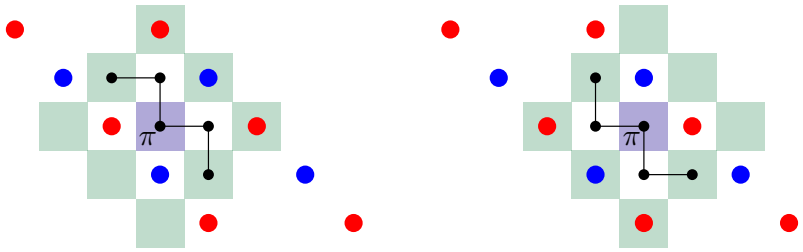


Figure 3 A pixel $\pi \notin \Phi$ (purple) and the set of pixels Π of distance 2 to it (green).

G_Φ does not see consists of a set of connected components, each of which is a subset of a "staircase" of infinite area. Formally, a staircase σ_c is a set of connected pixels where the coordinates (x, y) of each pixel satisfy $3c + 1 \leq x + y \leq 3c + 2$ for some integer constant c. Let S be a connected component of $P_{\overline{\Phi}}$. Since S can only be a subset of $\mathbb{Z}^2 \setminus \Phi$ (which is a set of staircases), and S is connected, it must be a subset of some σ_c.

Let π, π' be two pixels in the polyomino, we use $d(\pi, \pi')$ to denote the geodesic distance between them. Similarly, $d(S, S') = \min_{\pi \in S, \pi' \in S'}(d(\pi, \pi'))$ is the geodesic distance between two connected components in $P_{\overline{\Phi}}$. Then, the following holds:

▶ **Lemma 3.2.** $\forall S, S' \in P_{\overline{\Phi}}, S \neq S' : d(S, S') \geq 3$.

Proof. Consider any pixel $\pi = (x, y) \in S \subseteq P_\Phi \subseteq P \setminus \Phi$. Next, we will only consider pixels of distance 2 to π in $\mathbb{Z}^2$, and prove that they cannot belong to a connected component different from S. We use Π to denote the set of such pixels. Figure 3 shows π and Π. For any pixel $\pi' \in \Pi \setminus \{(x + 1, y - 1), (x - 1, y + 1)\}$, the shortest path from π to π' must cross one of the guards in Φ and π' is seen by that guard. Therefore, we only need to consider $\pi' \in \{(x + 1, y - 1), (x - 1, y + 1)\}$. We assume $(x + y) \equiv 1 \pmod 3$ for our argument, i.e., the left illustration in Figure 3 (the case $(x+y) \equiv 2 \pmod 3$ is similar). If $\pi' = (x+1, y-1)$, the only available shortest path from π to π' must be $[\pi, (x + 1, y), \pi']$ (since $(x, y - 1)$ is occupied by a guard). For π and π' to belong to two different components, $(x + 1, y)$ must be seen by a guard in Φ. But it is not possible to see $(x + 1, y)$ without also seeing π or π' because there is no obstacle between π (or π') and $(x + 1, y)$. Therefore, π and π' cannot both be unseen pixels in different components. The case when $\pi' = (x - 1, y + 1)$ can be argued similarly. ◀

Theorem 3.2 essentially allows us to guard each connected component independently from the other. Next, we define $d(S, G_\Phi) = \min_{p \in S, g \in G_\Phi} d(p, g)$ to be the distance between S and the guard set G_Φ. Note that $d(S, G_\Phi) \geq 2$. Otherwise, the guard in G_Φ closest to S must see at least one more pixel of S. Let $|S|$ be the number of pixels of S and $g \in G_\Phi$ be the guard such that $d(S, g) = 2$, we claim that (proof omitted due to lack of space):

▶ **Lemma 3.3.** *If $d(S, G_\Phi) = 2$, then $|S| = 1$ and g is unique.*

In this case, we can guard S by placing a guard g' of a different color from g at the only pixel of S. For any S where $|S| \geq 2$, we reuse the pattern in Φ, shifted one pixel to the left. More specifically, we place a guard at every pixel (x, y) of S such that $(x + y) \equiv 2 \pmod 3$ and the guard is red if y is even and blue if y is odd. This will guard all of S. Due to Lemma 3.2 and Lemma 3.3, the choice of guards in S is independent of other connected components, as well as guards in G_Φ. Reapplying this process to all unseen components, P will be guarded.

4 NP-Hardness of 2-CDAGP with Distance $d = 5$

Iwamoto and Takahashi [10] prove that 2-CDAGP where guards of the same color need to keep a distance of at least $d = 6$ is NP-complete. They provide a reduction from Clause-Linked Planar 3SAT, a variation of Planar 3SAT where the clause vertices $C_1, \ldots, C_m$ in the corresponding graph are connected with edges (C_j, C_{j+1}) for $1 \leq j \leq m - 1$, whose NP-completeness follows from [13].

We use the same reduction from Clause-Linked Planar 3SAT as in [10], and modify the gadgets such that in some parts of the polyomino it is necessary to place guards of the same color with distance 5 to each other. Let Ψ be an instance of Clause-Linked Planar 3SAT with n variables and m clauses. We construct a polyomino P_C with n variable gadgets and m clause gadgets.

▶ **Lemma 4.1.** *A 3SAT-instance Ψ is satisfiable if and only if there is a set of guards of two colors with r-visibility for polyomino P_C such that every two guards with the same color are placed at a distance of at least 5.*

We use the same techniques as [10], so, our proof is nearly identical to that of Lemma 1 from their paper. We instead present redesigned gadgets, which let us reduce the dispersion distance by one.

All our gadgets are functionally the same as those in [10]. For the brevity we will omit a proof of Lemma 4.1.

Variable gadgets. To encode the input variables $x_1, x_2, \ldots, x_n$ of Ψ, we add a variable gadget for each x_i, see Figure 4(a). The gadget consists of a horizontal corridor with notches attached on its top and bottom. The distance between consecutive notches at the top, and consecutive notches at the bottom, is 5. Therefore, to see the notches, guards must be placed inside the notches. Moreover, the distance between top and bottom notches enforces that top notches are guarded by exclusively one color and bottom notches by the other. To cover the corridor one guard must be placed inside it. An initial notch in the left fixes this guards' position right below the notch; this guard also determines the truth value assigned to x_i, where a blue guard represents true and a red guard represents false. The guards in the top notches take the same color as the initial guard.

Wire gadgets. Wire gadgets form the connections between gadgets, transferring the true or false values of variables. A wire gadget is similar to a variable gadget, excluding the initial notch. It consists of a corridor with notches of distance 5 on its top and bottom

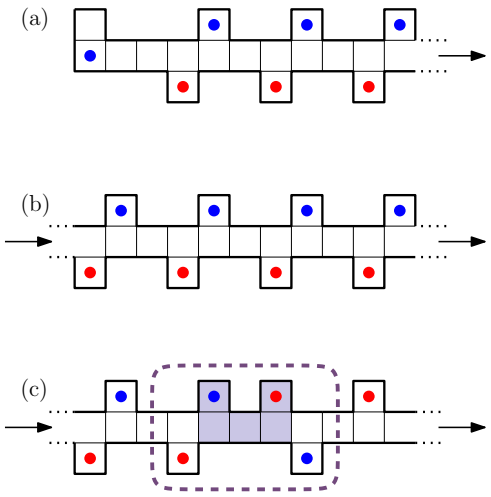


Figure 4 (a) A variable gadget, (b) a wire gadget, and (c) a NOT gadget.

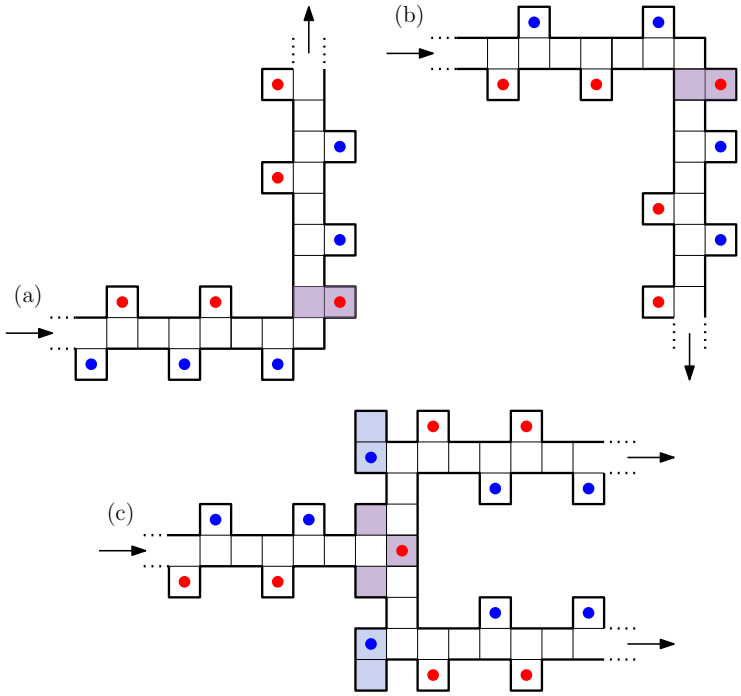


Figure 5 (a)-(b) Curve gadgets. (c) A branching gadget. The sets of highlighted pixels are alternative placements for guards to see the corresponding notches.

edges, see Figure 4(b). The gadget is guarded analogously to the variable gadget. Note, in Figure 4(b), notches of opposite colors are placed in pairs with adjacent x-coordinates. We could instead place a pair with a gap that is one unit larger, which would shift the notch pattern. When constructing our final polyomino this allows us to place gadgets without having to respect this pattern.

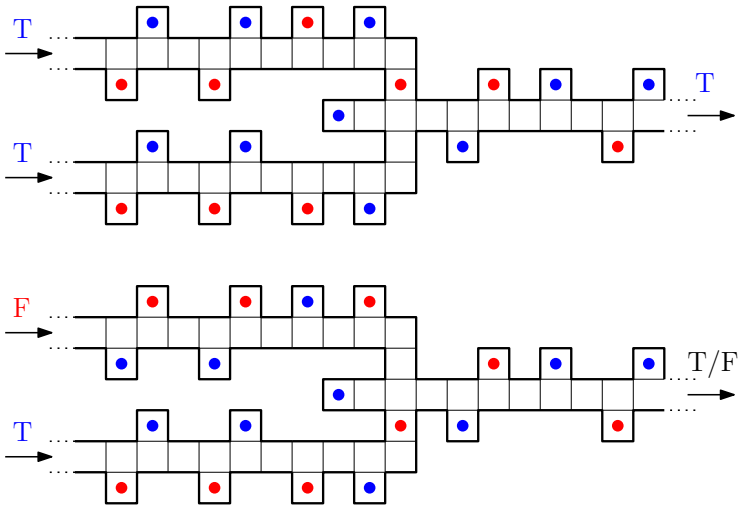


Figure 6 An OR gadget. Equal input assignments forces the same output assignment (T/T → T, F/F→ F) (top); different input assignments allow for both true and false output (bottom).

NOT gadgets. The NOT gadget takes a wire and performs a small modification to shift the parity, see Figure 4(c). Two notches with distance 4 are added on the top; they can only be guarded by placing two guards with different colors inverting the corridor's color scheme.

Curve gadgets. The curve gadget is a universal tool that changes the orientation of an input variable in the polyomino, see Figure 5(a) and (b). It connects two perpendicular wire gadgets with each other. We use two different versions of curve gadgets, depending on the parity of the shifted input notches, turning it either away from the last notch on the input wire (Figure 5(a)), or towards it (Figure 5(b)). To guard the first notch after the bend, one guards needs to be placed either inside the notch or in the adjacent pixel. The latter placement ensures that also the corridor is seen.

Branching gadgets. The branching gadget takes an input variable x and routes it to two outputs which are negations of the original value, see Figure 5(c). However, this negation is readily undone by placing a NOT gadget in each output (not depicted). The gadget consists of two opposite curve gadgets overlaid. One guard needs to be placed in the center to see the two center notches; the valid placements are marked in purple in Figure 5(c). However, if it were placed in one of the left purple pixels, the vertical corridor would remain unguarded.

OR gadgets. An OR gadget takes two inputs, and yields one output, see Figure 6. If both inputs have the same value (both true or both false), then guards can only be assigned such that the output corresponds to the opposite value. As with the branching gadget, this negation can be undone by adding a NOT gadget to the output wire. If the input guards encode different values, then both true and false output is possible.

Clause gadgets. Our clause gadgets (Figure 7(a)) are identical to those of [10], see Figure 9 in [10]. For a clause of three variables $\{x_1, x_2, x_3\}$ we construct three OR gadgets, one $x_1 \vee x_2 = \rho$ and two OR gadgets $\rho \vee x_3$. The clause gadget then has two outputs, one for each $\rho \vee x_3$. The gadget requires two outputs to form the clause cycle for the reduction from Planar Clause Linked 3-SAT.

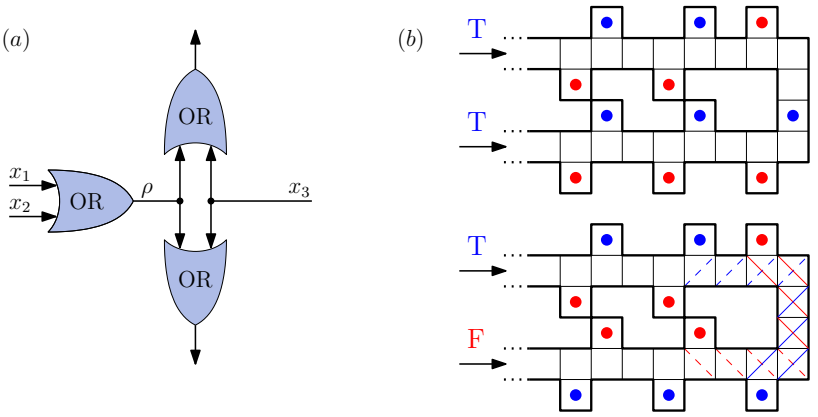


Figure 7 (a) A clause gadget. (b) An XNOR gadget. The tunnel can be guarded if the two inputs have the same truth value (top), but not if the two inputs have a different truth assignment (bottom), as the guards in the rightmost and second to rightmost notches prohibit a guard to be placed in the tunnel. The crossed out pixels indicate a violation to the distance requirement.

XNOR gadgets. The XNOR gadget is used to combine two clauses C_j and C_{j+1}. It consists of two input wires and one tunnel of four pixels connecting the inputs, see Figure 7(b). If both input wires are assigned the same truth value, a guard can be placed in the tunnel and the gadget is guarded. If the two input wires have different truth values, no color of guard may be placed in the tunnel and some pixels of the XNOR are unguarded.

Constructing the polyomino. Similar to [10], we put these gadgets together to form the final polyomino. Given an embedding of a Planar Clause Linked 3-SAT instance Ψ, we replace each arc with wire and branch gadgets, each variable node with a variable gadget, and each clause with a clause gadget. The gadgets are linked using wire gadgets and curve gadgets. We also add a variable x_0 and one "dummy" clause $C_0 = x_0 \vee x_0 \vee x_0$, so C_0 is true if x_0 is set to true (in particular, the choice of color for x_0 determines the color encoding for true and false). Finally, following the clause cycle in Ψ, we place an XNOR gadget between the outputs of two adjacent clause gadgets. By connecting clauses with XNORs we force each clause to have the same value, true or false. Therefore, with the XNOR linkages, if the polyomino can be guarded, each clause will be assigned true.

▶ **Theorem 4.2.** *2-CDAGP is NP-complete in polyominoes with holes for $d = 5$.*

5 Closing the Gap: Distance $d = 4$?

Chromatic Dispersive Art Gallery is NP-complete for $d \geq 5$, and polynomial time solvable for $d \leq 3$. The remaining question is then: what about $d = 4$? As of yet we have been unable to adapt our $d = 5$ OR gadgets. In an OR gadget, the two input wires need to be near each other so they "interact" (produce OR gate logic), and for $d = 4$ they must be so close that our $d = 5$ OR gadgets cannot be adapted in any obvious (or less than obvious) manner. Also, there is no immediate way to extend our algorithm to $d = 4$, since it is not possible to place a guard in every pixel along a diagonal while maintaining an intracolor distance of 4. Additionally, while every polyomino is guardable for $d \leq 3$, there are polyominoes that are impossible to guard for $d \geq 4$, see for example Figure 8.

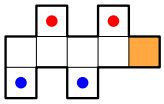


Figure 8 An infeasible instance for distance $d = 4$. After placing one guard in each notch, the orange pixel on the right cannot be guarded.

Acknowledgments. We thank the anonymous reviewers for their careful consideration and sharp eyes. Their comments improved our paper and alerted us to error with our construction.

References

1 Mikkel Abrahamsen, Anna Adamaszek, and Tillmann Miltzow. Irrational guards are sometimes needed. In *Symposium on Computational Geometry (SoCG)*, pages 3:1–3:15, 2017. doi:10.4230/LIPIcs.SoCG.2017.3.

2 Mikkel Abrahamsen, Anna Adamaszek, and Tillmann Miltzow. The art gallery problem is $\exists\mathbb{R}$-complete. *Journal of the ACM*, 69(1):4:1–4:70, 2022. doi:10.1145/3486220.

3 Therese C. Biedl, Mohammad T. Irfan, Justin Iwerks, Joondong Kim, and Joseph S. B. Mitchell. Guarding polyominoes. In *Symposium on Computational Geometry (SoCG)*, pages 387–396, 2011. doi:10.1145/1998196.1998261.

4 Vasek Chvátal. A combinatorial theorem in plane geometry. *Journal of Combinatorial Theory*, 18(1):39–41, 1975. doi:10.1016/0095-8956(75)90061-1.

5 Lawrence H. Erickson and Steven M. LaValle. A chromatic art gallery problem. Technical report, University of Illinois at Urbana-Champaign, 2010.

6 Lawrence H. Erickson and Steven M. LaValle. How many landmark colors are needed to avoid confusion in a polygon? In *2011 IEEE International Conference on Robotics and Automation*, pages 2302–2307, 2011. doi:10.1109/ICRA.2011.5979838.

7 Lawrence H. Erickson and Steven M. LaValle. An art gallery approach to ensuring that landmarks are distinguishable. In *Robotics: Science and Systems VII*. The MIT Press, 06 2012. doi:10.7551/mitpress/9481.003.0016.

8 Sándor Fekete, Joseph S.B. Mitchell, Christian Rieck, Christian Scheffer, and Christiane Schmidt. Dispersive vertex guarding for simple and non-simple polygons. In *36th Canadian Conference on Computational Geometry (CCCG 2024)*, pages 33–40, 2024. URL: https://cccg.ca/proceedings/2024/CCCG_2024_proceedings.pdf.

9 Steve Fisk. A short proof of Chvátal's watchman theorem. *Journal of Combinatorial Theory*, 24(3):374, 1978. doi:10.1016/0095-8956(78)90059-X.

10 Chuzo Iwamoto and Shun Takahashi. Computational complexity of the chromatic dispersive art gallery problem. *IEICE Transactions on Information and Systems*, advpub, 2025. doi:10.1587/transinf.2025FCP0004.

11 D. T. Lee and Arthur K. Lin. Computational complexity of art gallery problems. *IEEE Transactions on Information Theory*, 32(2):276–282, 1986. doi:10.1109/TIT.1986.1057165.

12 Fekete Sándor P., Kobbe Kai, Krupke Dominik, Mitchell Joseph S. B., Rieck Christian, and Scheffer Christian. Guarding Offices with Maximum Dispersion. In Paweł Gawrychowski, Filip Mazowiecki, and Michał Skrzypczak, editors, *50th International Symposium on Mathematical Foundations of Computer Science (MFCS 2025)*, volume 345 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 46:1–46:17, Dagstuhl, Germany, 2025. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.MFCS.2025.46.

13 Alexander Pilz. Planar 3-sat with a clause/variable cycle. *Discret. Math. Theor. Comput. Sci.*, 21(3), 2019. doi:10.23638/DMTCS-21-3-18.

14 Christian Rieck and Christian Scheffer. The dispersive art gallery problem. *Computational Geometry: Theory and Applications*, 117:102054, 2024. doi:10.1016/j.comgeo.2023.102054.

Online Packing of Orthogonal Polygons

Tim Gerlach[1], Benjamin Hennies[2], and Linda Kleist[3]

1 **Universität Hamburg, Germany**
`tim.gerlach@uni-hamburg.de`
`https://orcid.org/0009-0004-6294-9235`
2 **Technische Universität Braunschweig, Germany**
`benjaminhennies@posteo.de`
3 **Universität Hamburg, Germany**
`linda.kleist@uni-hamburg.de`
`https://orcid.org/0000-0002-3786-916X`

Abstract

Given recent insights that convex polygons do not allow for constant competitive online algorithms for diverse variants under translation, we study orthogonal polygons, in particular of small complexity. For translational packings of orthogonal 6-gons, we show that the competitive ratio of any online algorithm that aims to pack the items into a minimal number of unit bins is in $\Omega(n/\log n)$, where n denotes the number of objects. In contrast, we present constant competitive algorithms when the orthogonal 6-gons are symmetric or small. For (orthogonally convex) orthogonal 8-gons and thus for general orthogonal polygons, we prove that the trivial n-competitive algorithm, which places each item in its own bin, is best-possible, i.e., every online algorithm has an asymptotic competitive ratio of at least n. We also discuss the consequences of our insights for other packings variants, e.g. strip packing, perimeter packing, and min area packing.

1 Introduction

Packing problems not only present us with constant challenges in everyday life, but also find applications in manufacturing industries, logistics, and scheduling. While rectangular and box-shaped objects dominate the classic discourse of theoretic investigations, a fascinating frontier lies in the packing of more complex objects, e.g., convex polygons. The problem of packing convex polygons is particularly interesting when restricting the allowed motions to translations, as allowing for rotations reduces the problem to mere rectangle packing. Across the numerous variants – be it strip packing, bin packing, or perimeter packing – the overall picture is that constant factor approximations exist for translational packings of convex polygons [5, 6, 1, 16]. However, the landscape changes when we step into the online realm where items arrive sequentially one after the other and decisions have to be made irrevocably without knowing the future items: In this setting no online algorithm for translational packings of convex polygons has a constant competitive ratio, as recently shown by Aamand, Abrahamsen, Beretta, and Kleist [1]. This is in stark contrast to the case of rectangles, for which constant competitive algorithms exist for various variants.

In this work, we study the online packing problem of *orthogonally convex orthogonal polygons*, i.e., the intersection with any axis-parallel line has at most one connected component and all sides are axis-parallel. The simplest yet unresolved setting involves orhogonally convex orthogonal 6-gons, which we call *L-shapes*. An L-shape is the union of a rectangle of width ℓ_x and height w_y with a rectangle of width w_x and height ℓ_y, with $w_x \leq \ell_x$ and $w_y \leq \ell_y$, such that their lower left corners coincide. Figure 1(a) presents an example. An L-shape is *small* if $\ell_x, \ell_y \leq 1/2$, *large* if $\ell_x, \ell_y \geq 1/2$, and *symmetric* if $\ell_x = \ell_y$ and $w_x = w_y$.

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

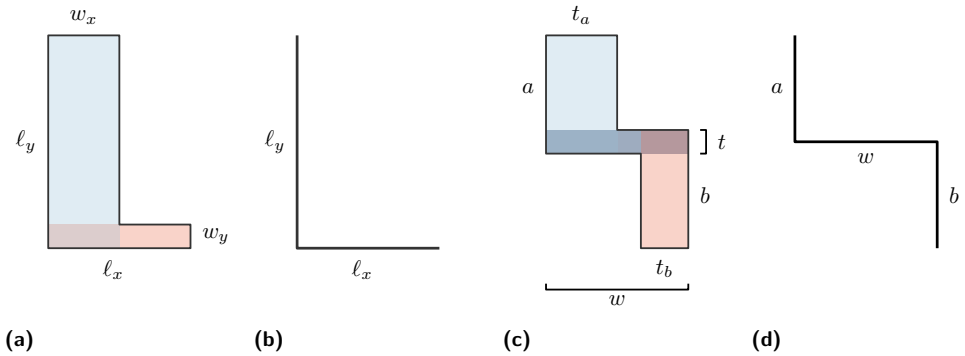


Figure 1 Parameters of an (a) L-shape, (b) L-skeleton, (c) Z-shape, (d) Z-skeleton.

A *Z-shape* is the union of three rectangles, see also Figure 1(c): A base rectangle of height t and width w, and two arm rectangles; one arm (above) of width t_a and height a, whose lower left corner coincides with the lower left corner of the base, and one arm (below) of width t_b and height b such that its upper right corner coincides with the upper right corner of the base, where $t_a, t_b \leq w$. L-skeletons and Z-skeletons are the corresponding shapes with zero thickness, i.e., $w_x = w_y = 0$ and $t_a = t_b = t = 0$, respectively, see Figures 1(b) and 1(d).

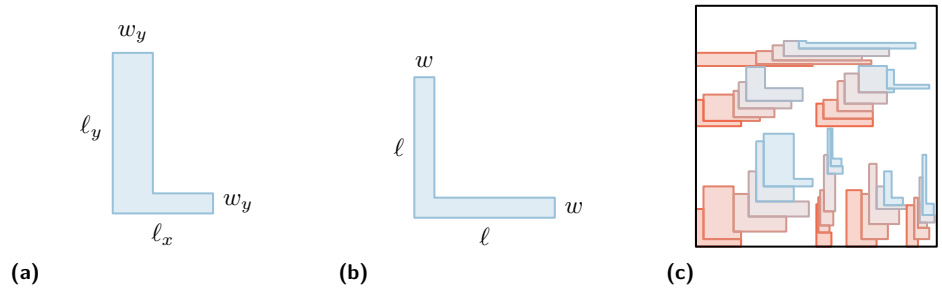


Figure 2 (a) An L-shape, (b) a symmetric L-shape, and (c) a bin packing of L-shapes.

Packing Problems The diverse applications motivate a multitude of interesting packing problems. Among them are bin packing, strip packing, perimeter packing and min area packing for various types of objects and allowed motions. Given a set (or stream) of items and an unbounded supply of identical unit bins, the goal of *bin packing* is to pack the items into as few bins as possible. Many variants of these packing problems are known to be computationally difficult. While NP-hardness is usually known for a long time [4, 11, 14, 17], some variants have more recently even been proved to be complete for the existential theory of the reals by Abrahamsen, Miltzow, and Seiferth [3].

Adding to this difficulty, we might not have complete information about the objects in advance, e.g., they may appear *online* over time. Here, the input is a stream $\sigma_1, \ldots, \sigma_n$ of objects, and we need to place each object σ_i irrevocably before the next object σ_{i+1} is revealed. As usual, we say an online algorithm has *(absolute) competitive ratio* $f(|I|)$ if, for all instances I, it holds that $\mathcal{A}(I) \leq f(|I|) \cdot \mathrm{OPT}(I)$, where $|I|$ denotes the size of I. Similarly, $\mathcal{A}$ has *asymptotic competitive ratio* $f(|I|)$ if there exists a constant $\beta > 0$ such that $\mathcal{A}(I) \leq f(|I|) \cdot \mathrm{OPT}(I) + \beta$ for all instances I.

1.1 Our contributions

We now present our contributions.

Bin Packing of 6-gons For bin packing of orthogonal polygons, it is natural to consider the variants where the objects can only be translated, and where the objects may be rotated by multiples of 90 degrees. The following proposition shows that even in the latter variant we may restrict our analysis to the translational case. In the problem Ortho-BinPack, we receive a stream of orthogonal polygons $P_1, ..., P_n$ which has to be placed into a minimum number of unit square bin by translation. Ortho-BinPack$_{\text{rot}}$ denotes the variant where the polygons might be rotated by multiples of 90 degrees.

▶ **Proposition 1.** *The best (absolute/asymptotic) competitive ratios for Ortho-BinPack and Ortho-BinPack$_{rot}$ are within constant factors of each other. The analogous statements hold for the best approximation ratios of the offline variants.*

Due to Proposition 1, we focus on translations only. We start by showing a lower bound on the competitive ratio of any online algorithm for L-shapes. To this end, let L-BinPack denote the variant of Ortho-BinPack where each polygon in the stream is an L-shape.

▶ **Theorem 2.** *The asymptotic competitive ratio of any online algorithm for L-BinPack is in $\Omega(n/\log n)$.*

Note that the trivial algorithm, which packs each object into its own bin, is n-competitive.

In contrast to Theorem 2, there exist constant competitive algorithms for some interesting subclasses of L-shapes, namely when the L-shapes are either all symmetric or all small. An L-shape is described by four parameters $\ell_x, \ell_y, w_x, w_y \in [0, 1]$. It is *small* if $\ell_x, \ell_y \leq 1/2$, *large* if $\ell_x, \ell_y \geq 1/2$, and *symmetric* if $\ell_x = \ell_y$ and $w_x = w_y$. For illustrations see Figure 2.

▶ **Theorem 3.** *There is an online algorithm for L-BinPack with a constant asymptotic competitive ratio when all L-shapes are symmetric or small. In particular, there exist algorithms that use at most*

(a) $33 \cdot OPT + 2$ *bins when all L-shapes are large and symmetric.*
(b) $8 \cdot OPT + 7$ *when all L-shapes are small.*
(c) $41 \cdot OPT + 9$ *when all L-shapes are symmetric.*

Theorem 3 directly yields algorithms with a constant absolute competitive ratio of 35, 15 and 50, respectively, as in bin packing, we have OPT ≥ 1 for any nonempty set of items.

Table 1 Bounds on the best asymptotic competitive ratios of bin packing orthogonal polygons.

Objects		Lower Bound		Upper Bound	
▯	rectangles	1.91	[9]	2.56	[12]
⌊	symmetric L-shapes	1.91	[9]	41	[Thm 3c]
⌊	orthogonal 6-gons	$n/\log n$	[Thm 2]	n	
⌊	orthogonal 8-gons	n	[Thm 5]	n	
⌊	L-frames	1.54	[7]	2	[Thm 6]
⌐	Z-frames	n	[Thm 6]	n	

The results for symmetric L-shapes are based on identifying their connection to scheduling problems. Competitive algorithms for two special cases of large and symmetric L-shapes follow directly from previous work on scheduling problems. Precisely, an e-competitive algorithm for objects of equal widths [8] and a 2-competitive algorithm for the case of equal lengths [10] can be deduced. A connection to scheduling problems is also used for the 33-competitive algorithm which builds on a constant competitive algorithm by Devanur et al. [8] for a special machine minimization problem in combination with an online coloring algorithm for intervals graphs by Kierstead and Trotter [15].

Other packing variants Our insights on online bin packing of L-shapes have consequences for other packing variants. In the problem L-PeriPack, we receive a stream of L-shapes $L_1, ..., L_n$, and the goal is to place them in the plane such that the perimeter of the bounding box is minimized. In L-AreaPack, the goal is to minimize the area of the bounding box. For a unit square bin and L-shapes with arm lengths ℓ_x, ℓ_y bounded by some constant $t < 1$, the *online critical density* is the largest value of A such that there exists an online algorithm that packs any stream with total area at most A into the bin.

▶ **Corollary 4.** *The following statements hold for L-shapes with arm lengths in* $[0, 1]$*:*

- **(i)** *The asymptotic competitive ratio of any online algorithm for L-StripPack is in* $\Omega(n/\log n)$.
- **(ii)** *There is an algorithm for L-PeriPack with an absolute competitive ratio in* $O(1)$.
- **(iii)** *There is an algorithm for L-AreaPack with an asymptotic competitive ratio of* $O(\sqrt{n})$.
- **(iv)** *The online critical packing density is positive if the arm lengths of each L-shape are bounded by* $t < 1$*. For* $t = 1$*, the online critical packing density is* 0.

The $O(\sqrt{n})$-competitive algorithm for L-AreaPack is asymptotically best-possible, as a lower bound of $\Omega(\sqrt{n})$ already for rectangles has been shown by Abrahamsen and Beretta [2].

Table 2 Overview on the known upper and lower bounds on the best competitive ratios for the various packing problems, and on the critical densities.

Packing Variant	rectangles	L-shapes	convex polygons	ortho polygons
Bin Packing	$\Theta(1)$ [13]	$\Omega\left(\frac{n}{\log n}\right)$ [Thm 2]	$\Omega\left(\sqrt{\frac{\log n}{\log\log n}}\right)$ [1]	n
Strip Packing	$\Theta(1)$ [18]	$\Omega\left(\frac{n}{\log n}\right)$ [Thm 4i]	$\Omega\left(\sqrt{\frac{\log n}{\log\log n}}\right), O(n^{0.59})$ [1]	$\Omega\left(\frac{n}{\log n}\right)$ [Thm 4i]
Perimeter Packing	$\Theta(1)$ [2]	$\Theta(1)$ [Thm 4ii]	$\Omega\left(\sqrt[4]{\frac{\log n}{\log\log n}}\right)$ [1]	
Min Area Packing	$\Theta(\sqrt{n})$ [2]	$\Theta(\sqrt{n})$ [2, Thm 4iii]	$\Omega(\sqrt{n})$ [2]	$\Omega(\sqrt{n})$ [2]
Critical Density	> 0	> 0	0	0

Orthogonal polygons of higher complexity For bin packing of orthogonal 8-gons (and thus also for Ortho-BinPack), we show that the trivial algorithm, that uses one bin for each item, achieves the best possible competitive ratio.

▶ **Theorem 5.** *There is no online algorithm for Ortho-BinPack that achieves an asymptotic competitive ratio better than* n*, even when restricting to orthogonally convex 8-gons (of equal thickness).*

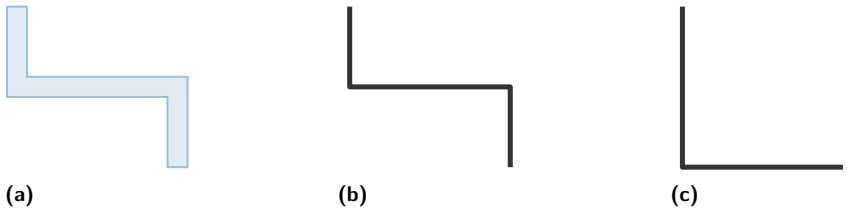


Figure 3 (a) A Z-shape (b) A Z-skeleton (c) An L-skeleton

We obtain Theorem 5 by considering Z-shapes of equal thickness; Figure 3(a) depicts an example. Interestingly, we obtain this result via first considering packings of skeletons, which can be viewed as orthogonal polygons of widths 0, see Figures 3(b) and 3(c). In the online problem L-Skel-BinPack (Z-Skel-BinPack), a stream of L-skeletons (Z-skeletons) shall be packed into a minimum number of bins. In this context, we consider a packing of skeletons as *valid* if any two objects are disjoint, with the exception that either of the two endpoints of some one skeleton may touch another; a point of a Z-skeleton is an endpoint if it is a unique extreme point, i.e., topmost, bottommost, leftmost, or rightmost. From L- to Z-skeletons, there is a quite drastic jump in complexity.

▶ **Theorem 6.** *For online bin packing of orthogonal skeletons the following holds.*

(i) *There exists an online algorithm for L-Skel-BinPack with a constant absolute competitive ratio.*

(ii) *No online algorithm for Z-Skel-BinPack has an asymptotic competitive ratio better than n.*

2 Lower bound for bin packing of L-shapes – Theorem 2

We present a proof sketch for the lower bound for L-BinPack. We proceed in two steps: We prove that any online algorithm for L-BinPack yields an online algorithm for an online sorting problem called BinSorting with the same competitive ratio, and then show that every online algorithm for BinSorting has a competitive ratio in $\Omega(n/\log n)$.

Bin sorting The online problem BinSorting$[k]$ can be described as a game between two players, Presenter and Algorithm, playing on several arrays with k slots each, in which natural numbers have to be inserted. Within each array, the numbers must be increasing. Presenter presents distinct numbers. When receiving a number, Algorithm has to irrevocably decide in which array and in which slot the number is placed, obeying the rule that the numbers of each array are increasing. Figure 4 presents an example of this game.

	2_4	16_1		
4_3	8_2			
		3_5		

Figure 4 An instance of the BinSorting with $k = 5$ for the stream $16, 8, 4, 2, 3$.

Its easy to see that the offline optimum to place n numbers in arrays with k slots is $\mathrm{OPT} = \lceil n/k \rceil$. In the online setting, we present a strategy for Presenter to keep

Algorithm from using more than a logarithmic number of slots in each array.

▶ **Lemma 7.** *The asymptotic competitive ratio of any online algorithm for* BinSorting$[k]$ *is at least* $\lceil n/\lfloor \log(k+1) \rfloor \rceil / \lceil n/k \rceil \in \Omega(k/\log k)$. *For* $k = n$, *the lower bound is* $n/\lfloor \log(n+1) \rfloor \in \Omega(n/\log n)$.

From L-BinPack to BinSorting We now show how to create instances of L-BinPack that simulate BinSorting. The construction is visualized in Figure 5. For fixed $k \in \mathbb{N}$, we consider the family $\mathcal{L}^k = \{L_i\}_{i \in \mathbb{N}}$ of L-shapes with the following parameters:

$$\ell_x(L_i) = 1/2 + 1/2k, \quad w_x(L_i) = 1/2k, \quad \ell_y(L_i) = 1 - 2^{-i-1}, \quad w_y(L_i) = 2^{-i-1}$$

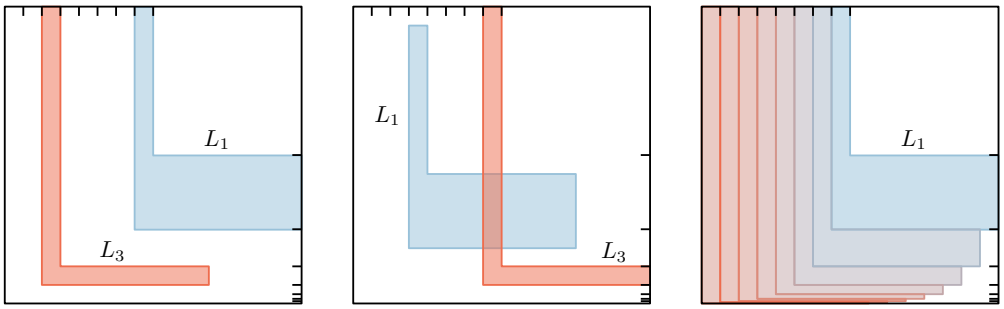


Figure 5 Illustrations for $\mathcal{L}^8$. (a) While L_1 may be placed to the right/above of L_3, (b) L_1 cannot be placed to the left/below of L_3 as $l_x(L_1)+l_x(L_3) > 1$ and $l_y(L_3)+w_y(L_1) = 1-2^{-3-1}+2^{-1-1} > 1$, respectively. (c) A packing of $L_1, \ldots, L_k \in \mathcal{L}^k$ for $k = 8$ into one bin.

The key idea is that L_i in L-BinPack mimics the natural number i for BinSorting.

▶ **Lemma 8.** *Each set* $S \subset \mathcal{L}^k$ *packs into one bin if and only if* $|S| \leq k$ *and the items are placed in descending order of their indices* i.

To prove Lemma 8, we show that k L-shapes can be packed, but even a single pair of incorrectly ordered shapes can not be placed within one bin. With this property, we can now translate solutions for L-shape online bin packing to solutions for BinSorting.

▶ **Lemma 9.** *An online-algorithm for L-BinPack yields an algorithm for* BinSorting$[k]$ *with the same (absolute/asymptotic) competitive ratio.*

Together, Lemmas 7 and 9 show the following statement, which proves Theorem 2.

▶ **Corollary 10.** *For each online algorithm* $\mathcal{A}$ *for L-BinPack and each* n, *there exists a strategy to present* n *L-shapes from* $\mathcal{L}^n$ *(no duplicates) such that they can all be packed into one bin while* $\mathcal{A}$ *uses at least* $n/\log(n+1)$ *bins.*

3 Open Problems

Many interesting problems remain for future work. Most notably, it remains to improve the bounds or even close the gaps for the problems stated in Tables 1 and 2. This includes the problem of determining the best competitive ratio for online bin packing of L-shapes, as it remains open whether it it possible to beat the trivial algorithm for L-BinPack.

References

1 Anders Aamand, Mikkel Abrahamsen, Lorenzo Beretta, and Linda Kleist. Online sorting and translational packing of convex polygons. In *Symposium on Discrete Algorithms (SODA 2023)*, pages 1806–1833. SIAM, 2023. doi:10.1137/1.9781611977554.ch69.

2 Mikkel Abrahamsen and Lorenzo Beretta. Online Packing to Minimize Area or Perimeter. In *37th International Symposium on Computational Geometry (SoCG 2021)*, pages 6:1–6:15, 2021. doi:10.4230/LIPIcs.SoCG.2021.6.

3 Mikkel Abrahamsen, Tillmann Miltzow, and Nadja Seiferth. Framework for $\exists\mathbb{R}$-completeness of two-dimensional packing problems. *TheoretiCS*, 3, 2024. doi:10.46298/theoretics.24.11.

4 Helmut Alt. Computational aspects of packing problems. *Bulletin of the EATCS*, 118, 2016. URL: http://eatcs.org/beatcs/index.php/beatcs/article/view/390.

5 Helmut Alt, Mark de Berg, and Christian Knauer. Approximating minimum-area rectangular and convex containers for packing convex polygons. *Journal of Computational Geometry*, 8(1):1–10, 2017. doi:10.20382/jocg.v8i1a1.

6 Helmut Alt, Mark de Berg, and Christian Knauer. Corrigendum to: Approximating minimum-area rectangular and convex containers for packing convex polygons. *Journal of Computational Geometry*, 11(1):653–655, 2020. doi:10.20382/jocg.v11i1a26.

7 János Balogh, József Békési, György Dósa, Leah Epstein, and Asaf Levin. A new lower bound for classic online bin packing. *Algorithmica*, 83:2047–2062, 2021. doi:10.1007/s00453-021-00818-7.

8 Nikhil Devanur, Konstantin Makarychev, Debmalya Panigrahi, and Grigory Yaroslavtsev. Online Algorithms for Machine Minimization, 2014. arXiv:1403.0486.

9 Leah Epstein. A lower bound for online rectangle packing. *Journal of Combinatorial Optimization*, 38:846–866, 2019. doi:10.1007/s10878-019-00423-z.

10 Leah Epstein. Open-end bin packing: new and old analysis approaches. *Discrete Applied Mathematics*, 321:220–239, 2022. doi:10.1016/j.dam.2022.07.003.

11 Robert Fowler, Michael Paterson, and Steven Tanimoto. Optimal packing and covering in the plane are NP-complete. *Information processing letters*, 12(3):133–137, 1981. doi:10.1016/0020-0190(81)90111-3.

12 Xin Han, Francis Y. L. Chin, Hing-Fung Ting, Guochuan Zhang, and Yong Zhang. A new upper bound 2.5545 on 2d online bin packing. *ACM Transactions on Algorithms*, 7(4):50:1–50:18, 2011. doi:10.1145/2000807.2000818.

13 Rolf Harren, Klaus Jansen, Lars Prädel, Ulrich M. Schwarz, and Rob van Stee. Two for one: Tight approximation of 2D bin packing. *International Journal of Foundations of Computer Science*, 24(8):1299–1328, 2013. doi:10.1142/S0129054113500354.

14 Rolf Harren, Klaus Jansen, Lars Prädel, and Rob Van Stee. A $(5/3+\varepsilon)$-approximation for strip packing. *Computational Geometry*, 47(2):248–267, 2014.

15 Henry A. Kierstead and William T. Trotter. An extremal problem in recursive combinatorics. *Congressus Numerantium*, 33(143-153):98, 1981.

16 Adam Kurpisz and Silvan Suter. Improved Approximations for Translational Packing of Convex Polygons. In *European Symposium on Algorithms (ESA 2023)*, volume 274 of *LIPIcs*, pages 76:1–76:14, 2023. doi:10.4230/LIPIcs.ESA.2023.76.

17 Joseph Leung, Tommy Tam, Chin Wong, Gilbert Young, and Francis Chin. Packing squares into a square. *Journal of Parallel and Distributed Computing*, 10(3):271–275, 1990. doi:10.1016/0743-7315(90)90019-L.

18 Deshi Ye, Xin Han, and Guochuan Zhang. A note on online strip packing. *Journal of Combinatorial Optimization*, 17(4):417–423, 2009. doi:10.1007/s10878-007-9125-x.

Approximating Triangle Covers of Polygons

Linda Kleist[1] **and Lena Scherzer**[1]

1 **University of Hamburg (UHH)**
`linda.kleist@uni-hamburg.de,lena.scherzer@uni-hamburg.de`

Abstract

We study the APX-hard problem of covering a polygon with a minimum number of triangles, which is known to be $\exists\mathbb{R}$-complete, even for polygons without holes and vertices in general position. Our main result is a 12-approximation for computing a minimum triangle cover in polygons without holes. For the special case of triangle partitions, we argue that a triangulation yields a 3-approximation.

1 Introduction

Representing complex objects as a union of geometric primitives such as triangles, rectangles, or convex polygons is a fundamental problem and may facilitate processing, rendering, and analysis. In this work, we study the problem of covering a polygon with triangles. Precisely, a set of triangles is a *triangle cover* of a polygon P if the union of the triangles equals P.

In the **Minimum Triangle Cover Problem (MTC)**, we are given a polygon and seek a triangle cover of P with a minimum number of triangles. Figure 1 depicts an example. We highlight that we do not restrict the vertices of the triangles, i.e., we allow Steiner points.

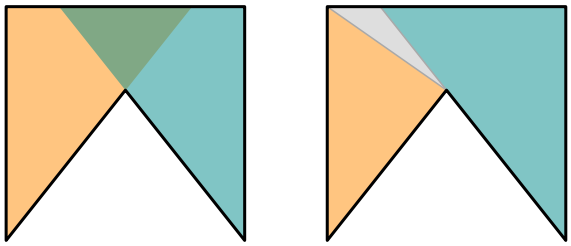


Figure 1 A polygon with a minimum triangle cover and minimum triangle partition.

Abrahamsen [1] settles the computational complexity by proving that MTC is $\exists\mathbb{R}$-complete, even for simple polygons with vertices in general position. Moreover, Christ [4] shows that MTC is APX-hard. As observed by Christ [4, p.2], if a polygon P does not have collinear sides, then a triangle can cover at most three sides of P. Hence, a triangulation of the polygon yields a simple 3-approximation. The interesting case is thus to handle polygons with collinear edges. MTC is a special case of the *Minimum Convex Cover Problem (MCC)*, where each primitive is a convex polygon. This problem has a long history; in fact the SoCG logo is from a paper by O'Rourke where he shows that MCC is decidable [12]. In modern terminology, he shows containment in $\exists\mathbb{R}$. As for triangles, $\exists\mathbb{R}$-hardness is proved by Abrahamsen [1]. For simple polygons, Browne, Kasthurirangan, Mitchell, and Polishchuk provide a 6-approximation algorithm for MCC [3]. We show how to modify their algorithm and obtain a 12-factor approximation for minimum triangle cover. While the algorithm reads similar, the analysis requires a new approach.

▶ **Theorem 1.1.** *There is a 12-approximation algorithm for computing a minimum triangle cover of simple polygons.*

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

A close inspection of the $O(\log n)$-approximation for MCC of polygons with holes by Eidenbenz and Widmayer [7] shows that the same tools can be used for MTC.

▶ **Corollary 1.2.** *There is an $O(\log n)$-approximation algorithm for computing a minimum triangle cover of polygons (possibly with holes).*

We also study the **Minimum Triangle Partition Problem (MTP)**, in which a triangle partition of a given polygon with a minimum number of triangles is sought; a *(triangle) partition* of a polygon is a special (triangle) cover where no two pieces share interior points. Figure 1 gives an example. Keil and Snoeyink [11] proved that MTP is NP-hard. We show that a triangulation is a 3-approximation for a minimum triangle partition of a polygon.

▶ **Theorem 1.3.** *There is a 3-approximation algorithm for computing a minimum triangle partition for a given polygon (possibly with holes).*

Table 1 provides an overview of the state of the art of related polygon covering problems. We highlight that for general polygons, developing constant-factor approximations for polygon coverings with triangles or convex pieces remains an interesting challenge for future research.

Table 1 An overview of the computational complexity of and approximation ratios for various polygon covering problems. GP stands for general position.

	simple polygon	polygon with holes
triangle partition	open 3-approx [Thm 1.3]	NP-hard [11] 3-approx [Thm 1.3]
triangle cover	$\exists\mathbb{R}$-complete [1], APX-hard [4] 12-approx [Thm 1.1] (GP: 3-approx [4])	$O(\log(n))$-approx [7]+[Cor. 1.2] (GP: 3-approx [4])
convex cover	$\exists\mathbb{R}$-complete [1], APX-hard [7] 6-approx [3]	$O(\log(n))$-approx [7]
(axis-aligned) rectangle cover	NP-complete [6] 2-approx [8]	NP-complete [9], APX-hard [2] $O(\sqrt{\log(n)})$-approx [10]

Organization In Section 2, we introduce the fundamental notions. We prove Theorems 1.1 and 1.3 in Sections 3 and 4, respectively. A self-contained proof of Corollary 1.2 will appear in a long version of this extended abstract.

2 Fundamental Concepts

Let P be a polygon. We say that P is *simple* if its boundary consists of a single non-crossing curve. A vertex of P is *convex* if the angle of P at v is $< \pi$, and *non-flat* if the angle differs from π. A set of points is in *general position* if no three points lie on a line.

Equivalent edges Two edges e and e' of P are *equivalent* if there exists a segment s that contains e and e', and lies within P, see Figure 2. For an edge e, $[e]$ denotes the equivalence class of e, and $\mathrm{s}([e])$ the convex hull of $[e]$.

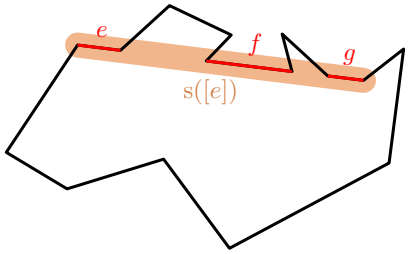


Figure 2 Three pairwise equivalent edges e, f, and g and their segment $\mathrm{s}([e])$.

Partition in weakly visible polygons A point p *sees* a point q of P if the segment between p and q lies within P. The visibility region of p is the set of all points seen by p. A polygon is *weakly visible* if it has an edge e such that each point in P is seen by some point in e.

A simple polygon P can be partitioned into weakly visible polygons by *staged illumination*, see Figure 3. The process starts from a convex vertex v of P. The first weakly visible polygon Q consists of the visibility region of v. Note that Q is separated from $P \setminus Q$ by segments, so-called *windows*. These windows are either disjoint or share common end points. In each following step, we consider a current window w and define a weakly visible polygon as the set of points in the remainder of P that see some point on w. This partition $\mathcal{Q}$ into weakly visible polygons can be computed in linear time [13].

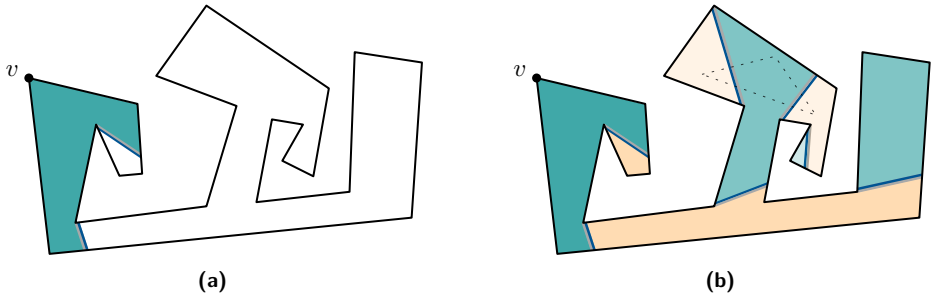


Figure 3 Illustration of stage illumination. (a) The first polygon and its windows; for each weakly visible polygon the blue side indicates the defining window, the gray sides behave as regular edges of the subpolygons. (b) A partition and a triangle that intersects three weakly visible polygons.

Browne et al. [3] establish a crucial property. By convexity, it also holds for triangles.

▶ **Lemma 2.1** ([3], Lemma 4.1). *Any convex polygon $K \subseteq P$ intersects at most three weakly visible polygons of $\mathcal{Q}$.*

3 Constant Factor Approximation for Minimum Triangle Cover

In this section, we present a proof of Theorem 1.1. We start by describing the algorithm for a simple polygon P, which is a modification of the algorithm for convex covers [3].

Firstly, we partition P into weakly visible polygons $\mathcal{Q}$ using staged illumination.

Secondly, for each weakly visible polygon Q in $\mathcal{Q}$, we construct two directed acyclic graphs, the previously used edge graph G and the new class graph $\tilde{G}$. We denote the vertices of Q by $v_1, v_2, \ldots, v_n$ such that $w = v_1 v_n$ is the defining window of Q, see also Figure 4. The path along the boundary from v_1 to v_n yields a total order $\prec$ of the vertices and edges of Q,

except w. The vertices of G correspond to the edges of Q except for w. Let e_i and e_j be two edges of Q with $e_i \prec e_j$. Then $(e_i, e_j) \in E(G)$ if and only if the convex hull of e_i and e_j lies in Q. In contrast, the class graph $\tilde{G}$ has a vertex for each equivalence class of the edges of Q, except for w. Two equivalent classes $[e_i]$ and $[e_j]$ share an edge $([e_i], [e_j])$ if $e \prec e'$ for all $e \in [e_i]$ and $e' \in [e_j]$, and the convex hull of $\mathrm{s}([e_i])$, $\mathrm{s}([e_j])$ and the intersection point x of their supporting lines is a triangle contained in P. For illustrations, consider Figure 4.

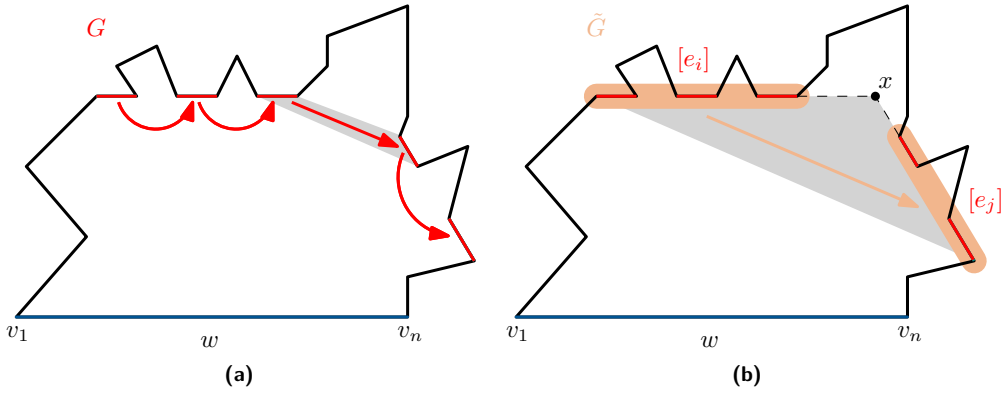


Figure 4 A weakly visible polygon with (a) a path in G (which corresponds to the path in (b)), and (b) a path in $\tilde{G}$.

Thirdly, we compute a minimum path partition $\tilde{\Pi}$ of $\tilde{G}$; a set of directed paths is a *path partition* of a graph if each vertex belongs to exactly one path. A minimum path partition of a directed acyclic graph can be computed via a standard reduction to a maximum matching in a bipartite graph [5, Problem 26-2]. Note that each path $\tilde{\pi} \in \tilde{G}$ corresponds to a path π on $\bigcup_{[e]\in\tilde{\pi}}[e]$ in G, see also Figure 4. Here we use the fact that, by definition of G and $\tilde{G}$, the edges of $[e]$ induce a path in G and an edge $([e_i], [e_j]) \in E(\tilde{G})$ implies that $(e, e') \in E(G)$ for all $e \in [e_i]$ and $e' \in [e_j]$. Hence $\Pi = \{\bigcup_{[e]\in\pi}[e] \mid \pi \in \tilde{\Pi}\}$ is a path partition of G.

Lastly, we construct a triangle cover of Q (and P). To this end, we consider a path $\tilde{\pi} \in \tilde{\Pi}$ (with corresponding path $\pi \in \Pi$) and construct a set T_π of $|\tilde{\pi}| + 1$ triangles; here $|\tilde{\pi}|$ denotes the number of vertices of $\tilde{\pi}$. Let a and b be the first and last vertex incident to an edge of π, respectively, see Figure 5. For each point $p \in Q$, let p_ℓ and p_r be the points on the window w that are visible from p and closest to v_1 and v_n, respectively. We define $p_{a,b}$ as the intersection point of segments aa_r and bb_ℓ. We add the triangles $abp_{a,b}$ and $bb_\ell b_r$ to T_π. Browne et al. [3] show that for any path of G these triangles lie in Q.

If $|\tilde{\pi}| \geq 2$, we add $|\tilde{\pi}| - 1$ more triangles to T_π. Let $\tilde{\pi} = [e_1], [e_2], \ldots, [e_k]$ and consider the convex hull C of all segments $\mathrm{s}([e_i])$ with $i \in [1, k]$, and the intersection points of the supporting lines of $\mathrm{s}([e_i])$ and $\mathrm{s}([e_{i+1}])$ for $i \in [1, k-1]$. By definition of $\tilde{G}$, two consecutive equivalence classes and their intersection point form a triangle. Hence, we can triangulate C with $k - 1$ triangles, which we add to T_π. We need to argue that these triangles lie in P. Conveniently, Browne et al. show that the convex hull C' of the edges in π is contained in Q [3, Lemma 2]. Thus the convex hull C, which additionally contains the tips of the triangles, lies in P. Lastly, we remark that $T_\Pi := \{T_\pi \mid \pi \in \Pi\}$ covers Q. This follows from the fact that the union over all paths π of a path partition of the convex hull C' of the edges of π, the triangle $abp_{a,b}$ and the triangle $bb_\ell b_r$ is a convex cover of Q [3, Lemma 3.4] and the fact that $C' \subset C$ and $abp_{a,b}, bb_\ell b_r \in T_\pi$ for all π. For later reference, we note the following.

▶ **Lemma 3.1.** *T_Π contains $|\tilde{\Pi}| + |V(\tilde{G})|$ triangles.*

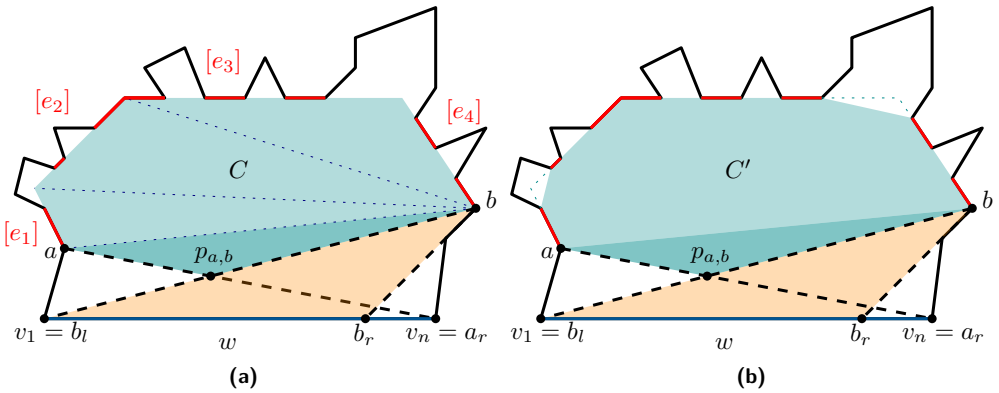


Figure 5 (a): Triangles in T_π for a path $\tilde{\pi} := ([e_1], [e_2], \ldots, [e_k]) \in \Pi_{\tilde{G}}$. (b): Two convex sets added by Browne et al. for the corresponding path π in G; they are covered by the triangles in T_π.

Proof. Each $\tilde{\pi}$ contributes $|\tilde{\pi}|+1$ triangles. Hence, we have $\sum_{\tilde{\pi}\in\tilde{\Pi}}(|\tilde{\pi}|+1) = |V(\tilde{G})|+|\tilde{\Pi}|$. ◀

To obtain a cover of P, we combine the sets of triangles of all polygons in $\mathcal{Q}$ to a set T. Because each $Q \in \mathcal{Q}$ is covered, T is a triangle cover of P. It remains to show that T is small. We note that our analysis needs new arguments: Firstly, the partition into weakly visible polygons may cut triangles of an optimal solution into convex but not triangular pieces. Secondly, for a better approximation guarantee, we partition C (including the tips) instead of C', as a result $\tilde{G}$ is not transitive and hence Dilworth's theorem cannot be applied. We instead use a charging argument. Let T^* be a minimum triangle cover of P.

▶ **Lemma 3.2.** $|T| \leq 12 \cdot |T^*|$.

Proof. For each weakly visible polygon Q of $\mathcal{Q}$, we construct a path partition $\tilde{\Pi}^*$ of its class graph $\tilde{G}$. To this end, we say an edge f of a (convex) polygon *supports* a class $[e]$ if the intersection of f and the edges in $[e]$ is not finite. For each triangle $t \in T^*$, we consider the subset t_Q that lies in Q; we call such a subset a *section*. Consider an edge f of t_Q supporting $[e]$. If no other edge of a section of a triangle in T^* supports $[e]$ then f has *full* responsibility; otherwise it has *shared* responsibility, see Figure 6.

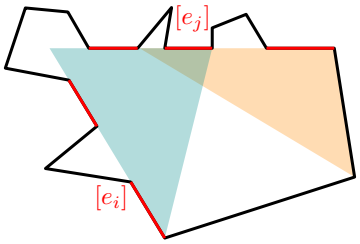


Figure 6 The cyan triangle has full responsibility for $[e_i]$ and shared responsibility for $[e_j]$.

With the help of T^*, we construct a path partition $\tilde{\Pi}^*$ of $\tilde{G}$. We consider the sides of a triangle section t_Q clockwise as a cyclic order, see Figure 7. If there is a side of t_Q that is fully contained in the window w_Q of Q, then we remove this side from the cyclic order. For each maximal sequence of consecutive sides $s_j, s_{j+1} \ldots, s_k$ of t_Q in the defined cyclic order where s_i has full responsibility for the class $[e_i]$, we add the path $[e_j], \ldots, [e_k]$ to $\tilde{\Pi}^*$.

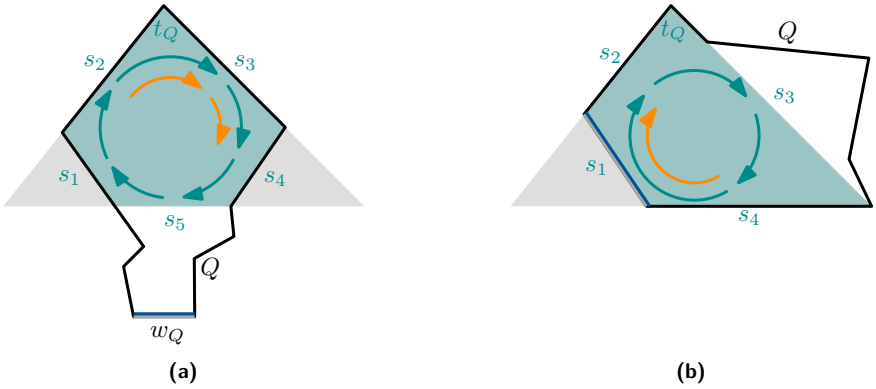


Figure 7 Two examples of a section t_Q with cyclical order (a): s_1, s_2, s_3, s_4, s_5 and (b): s_2, s_3, s_4 and maximal sequences of consecutive sides that have full responsibility (a): s_2, s_3, s_4 and (b): s_4, s_2.

Note that the edge $([e_i], [e_{i+1}])$, $i \in \{j, \dots, k-1\}$, is in $\tilde{G}$ because either $[e_i]$ and $[e_{i+1}]$ are contained in adjacent sides of t_Q or (if they are separated by w_Q) t. This guarantees that the convex hull of $s([e_i])$ and $s([e_{i+1}])$ and their intersection point is contained in P. For each not yet covered class $[e]$, we add $[e]$ to $\tilde{\Pi}^*$ (as a path of length 1); note $[e]$ is supported by at least two section edges with shared responsibility. Let $T(\tilde{\Pi}^*)$ denote the set of triangles that corresponds to the triangle cover obtained by using our algorithm on $\tilde{\Pi}^*$ instead of the minimal path cover $\tilde{\Pi}$ yielding T_Π. By Lemma 3.1 and minimality of $\tilde{\Pi}$, we have

$$|T_\Pi| = |\tilde{\Pi}| + |V(\tilde{G})| \leq |\tilde{\Pi}^*| + |V(\tilde{G})| = |T(\tilde{\Pi}^*)|.$$

It thus remains to show that the union of all $T(\tilde{\Pi}^*)$ over all $Q \in \mathcal{Q}$ contains at most $12 \cdot |T^*|$ triangles. To this end, we charge each triangle in some $T(\tilde{\Pi}^*)$ to a triangle in T^* and show that each triangle in T^* obtains charge of at most 12.

Consider a section t_Q of a triangle $t \in T^*$. Let $c \in \{3, 4, 5\}$ denotes its complexity, i.e., the number of sides of t_Q. For each path of length ℓ in $\tilde{\Pi}^*$ from full responsibilities of sides of t_Q, we charge all $\ell + 1$ triangles to t_Q. For a path of length 1 in $\tilde{\Pi}^*$ from a shared responsibility for some $[e]$, we charge one triangle to t_Q; because $[e]$ is supported by another side, the (two) constructed triangles for the path are accounted for. Let e be the number of edges that do not support any equivalence class of t_Q and Δ_w be 1 if a side of t_Q is contained in the window w_Q and 0 otherwise. For the charge $\mathrm{ch}(t_Q)$ of t_Q, we show

$$\mathrm{ch}(t_Q) \leq c - e + \lfloor (c - \Delta_w)/2 \rfloor. \tag{1}$$

We denote the number of shared responsibility edges by s and the lengths of full responsibility paths by $f_1, \dots, f_p$ with $f := \sum_i f_i$. Then the section t_Q obtains a charge of s from the shared responsibility edges and $\sum_i (f_i + 1) = f + p$ from the full responsibility paths. This yields a total charge $\mathrm{ch}(t_Q) \leq s + f + p$. Using $c - e = s + f$, the charge is at most $c - e + p$. Lastly, p is maximized if every second edge in the cyclic order has full responsibility and every other edge has shared responsibility (or does not support an equivalence class in Q). Hence, $p \leq \lfloor (c - \Delta_w)/2 \rfloor$. This completes the claim.

Now, we analyse the total charge $\mathrm{ch}(t)$ of a triangle t. By Lemma 2.1, each triangle in T^* is intersected by at most two windows, i.e., its total charge comes from at most tree sections. If we have at most two sections of complexities c_1 and c_2, then the total complexity is at most $c = c_1 + c_2 \leq 7$ and thus $\mathrm{ch}(t) \leq c - \lfloor c_1/2 \rfloor - \lfloor c_2/2 \rfloor \leq c - \lfloor c/2 \rfloor \leq 10$.

For the remainder, we consider the case of three sections t_A, t_B, t_C. One of them, denoted by t_C, is *central*, i.e., adjacent to the other two sections. The weakly visible polygon C is discovered before A and B during the staged illumination; otherwise the entire triangle t is visible from each window. Thus, for t_A and t_B, we have $\Delta_w = 1$ and $e \geq 1$. For t_C, we consider the edge or two edges of t_C that separate t_C from the window w_C of C. These edges do not support any equivalence class of C; otherwise the boundary of C intersects the sight line from w_C to the windows in t_C, see also Figure 8. Thus, $e \geq 1$ for t_C.

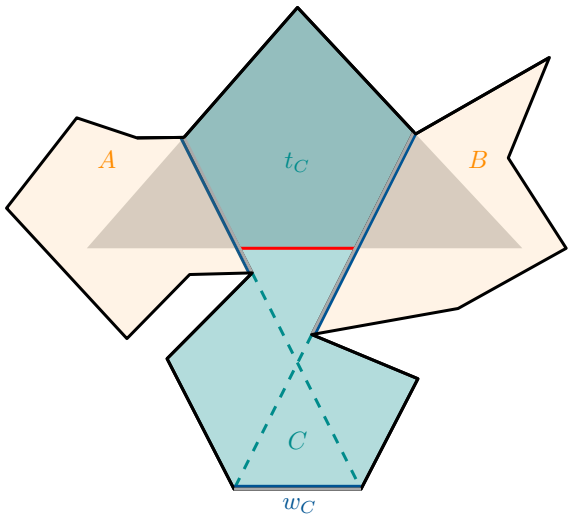


Figure 8 The red edge separates t_C from the window w_C of C. It can be seen that the red edge can not touch the boundary of C anywhere besides at the two endpoints.

Figure 9 illustrates the two interesting cases how two windows can intersect t. (It is straight-forward to check that similar arguments apply for the case that two windows share an endpoint or intersect in a vertex of t and only yield a smaller charge.) In the first case, sections t_A, t_B, t_C have complexity 3, 4, 4, respectively. By Inequality (1), $\Delta_w = 1$ for A, B, and $e \geq 1$ for A, B, C, we have $\mathrm{ch}(t) \leq \mathrm{ch}(t_A)+\mathrm{ch}(t_B)+\mathrm{ch}(t_C) \leq 3+4+5 = 12$. In the second case, sections t_A, t_B, t_C have complexity 3, 3, 5, respectively. By Inequality (1), $\Delta_w = 1$ for A, B, and $e \geq 1$ for A, B, C, we have $\mathrm{ch}(t) \leq \mathrm{ch}(t_A) + \mathrm{ch}(t_B) + \mathrm{ch}(t_C) \leq 3+3+6 = 12$. ◀

Lastly, we note that the running time is in $O(n^{2+o(1)})$.

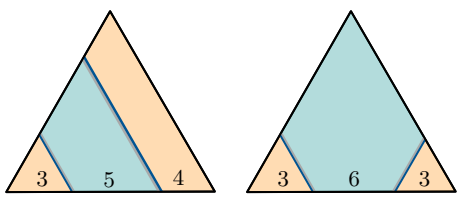


Figure 9 Two variants how a triangle can be split by at most two windows into sections of maximal complexity. The number in each triangle section denotes the maximum received charge.

4 Constant Factor Approximation for Minimum Triangle Partition

In this section, we prove Theorem 1.3. Let P be a polygon with n non-flat vertices. We compute a triangulation T of P with $n-2$ triangles and show that any triangle partition of P has at least $n/3$ triangles. Hence, T is 3-approximation of a minimum triangle partition.

▶ **Lemma 4.1.** *Any triangle partition T' of a polygon P with n non-flat vertices has at least $n/3$ triangles.*

Proof. Let v be a non-flat vertex of P. We show that T' contains a triangle with vertex v. To this end, consider a triangle t of T' incident to v. Either v constitutes a vertex of t or is in the interior of an edge of t. In the latter case, t covers an angle of π at v. As v is non-flat, there exists a remaining angle $\alpha \in (0, \pi)$ at v. Any such angle is covered by a collection of triangles with v as a vertex, see also Figure 10. ◀

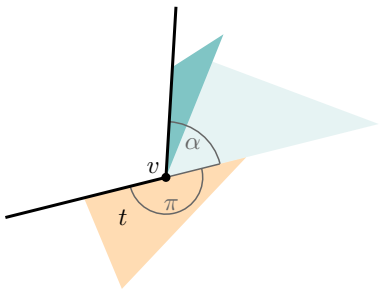


Figure 10 In every triangle partition, each non-flat vertex v is a vertex of some triangle.

References

1 Mikkel Abrahamsen. Covering polygons is even harder. In *Symposium on Foundations of Computer Science (FOCS)*, pages 375–386, 2022. doi:10.1109/FOCS52979.2021.00045.

2 Piotr Berman and Bhaskar Dasgupta. Complexities of efficient solutions of rectilinear polygon cover problems. *Algorithmica*, 17, 10 1996. doi:10.1007/BF02523677.

3 Reilly Browne, Prahlad Narasimham Kasthurirangan, Joseph S. B. Mitchell, and Valentin Polishchuk. Constant-factor approximation algorithms for convex cover and hidden set in a simple polygon. In *Symposium on Foundations of Computer Science (FOCS)*, pages 1357–1365, 2023. doi:10.1109/FOCS57990.2023.00083.

4 Tobias Christ. Beyond triangulation: Covering polygons with triangles. In *Algorithms and Data Structures*, pages 231–242. Springer, 2011. doi:10.1007/978-3-642-22300-6_20.

5 Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms, Third Edition*. The MIT Press, 3rd edition, 2009.

6 Joseph C. Culberson and Robert A. Reckhow. Covering polygons is hard. In *Symposium on Foundations of Computer Science (FOCS)*, pages 601–611, 1988. doi:10.1109/SFCS.1988.21976.

7 Stephan J. Eidenbenz and Peter Widmayer. An approximation algorithm for minimum convex cover with logarithmic performance guarantee. *SIAM Journal on Computing*, 32(3):654–670, 2003. doi:10.1137/S0097539702405139.

8 Deborah S. Franzblau. Performance guarantees on a sweep-line heuristic for covering rectilinear polygons with rectangles. *SIAM Journal on Discrete Mathematics*, 2(3):307–321, 1989. doi:10.1137/0402027.

9 Michael R. Garey and David S. Johnson. *Computers and Intractability; A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., USA, 1990.

10 V. S. Anil Kumar and H. Ramesh. Covering rectilinear polygons with axis-parallel rectangles. *SIAM Journal on Computing*, 32(6):1509–1541, 2003. doi:10.1137/S0097539799358835.

11 Andrzej Lingas. The power of non-rectilinear holes. In *Automata, Languages and Programming*, pages 369–383, 1982. doi:10.1007/BFb0012784.

12 Joseph O'Rourke. The complexity of computing minimum convex covers for polygons. In *Annual Allerton Conference on Communication, Control, and Computing*, pages 75–84, 1982. URL: https://www.science.smith.edu/~jorourke/Papers/MinConvexCovers_Allerton_1982.pdf.

13 Subhash Suri. On some link distance problems in a simple polygon. *IEEE Transactions on Robotics and Automation*, 6(1):108–113, 1990. doi:10.1109/70.88124.

A Framework for Dimension Reduction for Curves

Matthijs Ebbens[*1], **Jie Lu**[2], **and Alexander Munteanu**[*3]

1 **University of Cologne, Germany**
ymebbens@gmail.com
2 **University of Cologne, Germany**
jie.lu.duesseldorf@gmail.com
3 **TU Dortmund, Germany**
alexander.munteanu@tu-dortmund.de

Abstract

We revisit random projections for reducing the dimension of high-dimensional polygonal curves. Drawing from the toolbox of randomized linear algebra, we give a considerably simplified proof of the known $O(\varepsilon^{-2}\log(nm))$ bound on the target dimension of a random projection that preserves the continuous Fréchet distance of polygonal curves up to a factor $(1\pm\varepsilon)$. Our proof is based on the concept of sparse oblivious subspace embeddings. While previous techniques were limited to the case of the Fréchet distance, our techniques are fairly general and extend to all possible distance measures that involve the maximum, a sum or an integral over Euclidean distances between pairs of points on both input curves. We define a generalized dissimilarity measure for curves that includes several popular measures such as Fréchet, q-DTW, Hausdorff, etc. as special cases and show that the same dimension reduction technique works for this generalized dissimilarity measure. Finally, we apply the same framework for dimension reduction to piecewise linear surfaces, after extending the distance measure suitably to such surfaces.

1 Introduction

In this paper, we study dimension reduction for polygonal curves and build up a fairly general framework, which is widely independent of the dissimilarity measure that is used to compare them. Thus, in contrast to metric embeddings, our framework aims to embed *the curves* into lower dimensional space, not a specific dissimilarity measure. We first motivate our work from a broader perspective before we return to polygonal curves and surfaces.

Dimension reduction is one of the most fundamental tasks in computational geometry and has many applications. For instance, algorithms for a computational task may have a dependence on the dimension of their input objects, such as points, shapes, or polygonal curves. Reducing the dimension of these objects, say from d to some $t \ll d$, allows to apply the same algorithm on the reduced data, but with the dependence on d replaced by the much smaller value t. This comes at the cost of an approximation error which is typically bounded within a factor of $(1\pm\varepsilon)$. Other areas where dimension reduction plays a crucial role are the construction of coresets [4, 1], compressed sensing [8, 7, 5] and learning problems [6, 9, 2].

As a first step towards opening these avenues for problems on polygonal curves and surfaces, we propose a fairly general framework for embedding these objects in lower dimensional space, while preserving their 'distance' as measured by popular dissimilarity measures such as the Fréchet metric, dynamic time warping (DTW) and Hausdorff distance.

* Funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – Project Numbers 459420781 (ME) and 535889065 (AM).

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

1.1 Results

We present our main result, which is a general framework for dimension reduction of parameterized piecewise γ-dimensional linear surfaces. Notably, this includes polygonal curves as special case for $\gamma = 1$. For the notation, and more details, we refer to Sections 2 and 4.

▶ **Theorem 1.1.** *Let $\sigma_1, \ldots, \sigma_n : [0,1]^\gamma \to \mathbb{R}^d$ be n piecewise linear surfaces with m pieces. Let $\mathcal{T}$ be a set of admissible traversals between surfaces and let μ be a measure on $[0,1]^\gamma$. There exists a random linear mapping $f : \mathbb{R}^d \to \mathbb{R}^t$ with $t \in O(\varepsilon^{-2}(\gamma + \log(mn/\delta)))$ such that, with probability at least $1-\delta$, it holds for all $1 \le i < j \le n$ that*

$$(1-\varepsilon)\, d(\sigma_i, \sigma_j) \le d(f(\sigma_i), f(\sigma_j)) \le (1+\varepsilon)\, d(\sigma_i, \sigma_j),$$

where the distance d between surfaces σ_i and σ_j is defined as

$$d(\sigma_i, \sigma_j) = \inf_{(\alpha,\beta)\in\mathcal{T}} \left(\int_{[0,1]^\gamma} \|\sigma_i(\alpha(r)) - \sigma_j(\beta(r))\|_2^q \, d\mu(r) \right)^{1/q}.$$

The dissimilarity measure is highly general and includes the most popular continuous dissimilarity measures: Fréchet (for $q \to \infty$), DTW (for $q = 1$), and its variants such as weak Fréchet, q-DTW for $q \in [0, \infty)$, etc. as special cases. While Hausdorff distance is not subsumed by the generalized dissimilarity measure, an extension to Hausdorff distance is immediate from our results.

We note that discrete variants are also included by choosing the measure μ of integration to be a counting measure. While discrete and continuous variants of admissible traversals for dissimilarity measures between polygonal curves have been defined in the literature, no notion of discrete traversals is known for surfaces. In particular, there is no natural ordering of the vertices. We thus propose a set-valued generalization of discrete traversals to piecewise linear surfaces and prove that our generalization is consistent with the standard definition in the special case of polygonal curves.

2 Preliminaries

Notation We set $[n] := \{1, \ldots, n\}$. For a vector $v \in \mathbb{R}^d$, and $p \in (0, \infty)$, we denote its ℓ_p-norm by $\|v\|_p = \left(\sum_{i=1}^d |v_i|^p\right)^{1/p}$. We remark that the cases $0 < p < 1$ only define quasi-norms. Other special cases are $\|v\|_\infty := \lim_{p\to\infty} \|v\|_p = \max_{i\in[d]} |v_i|$, and $\|v\|_0 = |\{i \in [d] \mid v_i \neq 0\}|$. We denote by $\Psi_k := \{x \in \mathbb{R}^d \mid \|x\|_0 \le k\}$ the set of k-sparse vectors in $\mathbb{R}^d$.

Polygonal curves and surfaces A *curve* in Euclidean space $\mathbb{R}^d$, is a continuous function $\sigma : [0,1] \to \mathbb{R}^d$. A *polygonal curve* is a curve such that there exist a finite number $m \in \mathbb{N}$ of values $0 = s_1 \le s_2 \le \ldots \le s_m = 1$, with $v_i = \sigma(s_i)$ which we call *vertices*. We say that σ has *complexity* m. Further, for each $i \in \{1, \ldots, m-1\}$ the two consecutive vertices $\sigma(s_i)$ and $\sigma(s_{i+1})$ are connected by an affine straight-line segment, i.e., it holds that

$$\sigma(s_i + \varsigma) = \left(1 - \frac{\varsigma}{s_{i+1} - s_i}\right) \cdot \sigma(s_i) + \frac{\varsigma}{s_{i+1} - s_i} \cdot \sigma(s_{i+1}),$$

for all $\varsigma \in [0, s_{i+1} - s_i]$. For simplicity of notation, we can fully characterize a polygonal curve σ by defining the sequence of its vertices $\sigma = (v_1, \ldots, v_m)$.

Analogously, a piecewise linear *surface* in $\mathbb{R}^d$ is a continuous function $\sigma : [0,1]^\gamma \to \mathbb{R}^d$ whose image consist of m linear pieces which are polygons/polyhedrons bounded by a total of N vertices $v_i \in \mathbb{R}^d$, such that every piece lies in a γ-dimensional affine subspace of $\mathbb{R}^d$.

Traversals and dissimilarity measures We work with discrete and continuous 'transformations'. One way to define them is the notion of *traversals*. Let $\sigma = (u_1, \ldots, u_{m'})$ and $\tau = (v_1, \ldots, v_{m''})$ be two curves of complexities m' and m'' respectively. Traversals are usually defined by inductively constructed sequences of point pairs (p, q) such that $p \in \sigma$ and $q \in \tau$. In particular, any such sequence T satisfies the properties that T starts with (u_1, v_1) and ends with $(u_{m'}, v_{m''})$ and that the successor of (u_i, v_j) in T is exactly one of $(u_{i+1}, v_j), (u_i, v_{j+1})$, or (u_{i+1}, v_{j+1}) for all $i < m', j < m''$. In the full version, we provide an equivalent definition using unordered sets instead of sequences, which will prove useful for our generalization to surfaces.

Traversals can be extended to the continuous case by specifying pairs of 'reparameterizations', which are functions $\alpha, \beta \colon [0,1]^\gamma \to [0,1]^\gamma$. Then we let the induced traversal be the set $T = \{(\sigma(\alpha(r)), \tau(\beta(r))) \mid r \in [0,1]^\gamma\}$. Sometimes it is appropriate to impose further restrictions on α, β such as injectivity, surjectivity, and/or monotonicity, as required, and denote $\mathcal{T} \subseteq \{(\alpha, \beta) \mid \alpha, \beta \colon [0,1]^\gamma \to [0,1]^\gamma\}$ the corresponding subset of *admissible* traversals.

The distance between two given curves σ, τ is defined as

$$d(\sigma, \tau) \coloneqq \inf_{(\alpha,\beta)\in\mathcal{T}} \left(\int_{[0,1]^\gamma} \|\sigma(\alpha(r)) - \tau(\beta(r))\|_2^q \, d\mu(r) \right)^{1/q},$$

where $\mathcal{T}$ is the set of admissible traversals and μ is a measure on $[0,1]^\gamma$.

Subspace embeddings The notion of ε-subspace embeddings [13] is central to our work. The following version is due to [14] and is slightly modified to fit our notation and setting.

▶ **Theorem 2.1** (Theorem 2.3 of [14]). *Let $A \in \mathbb{R}^{d\times k}$. There exists a distribution over random matrices $S \in \mathbb{R}^{t\times d}$ with $t \in O(\varepsilon^{-2}(k + \log(1/\delta)))$ such that it holds with probability $1 - \delta$ that*

$$\forall x \in \mathbb{R}^d : (1-\varepsilon)\|Ax\|_2 \le \|SAx\|_2 \le (1+\varepsilon)\|Ax\|_2.$$

We will exploit ε-subspace embeddings for *collections* of k-dimensional subspaces. To this end we use the following theorem of [10] slightly modified to fit our notation and setting.

▶ **Theorem 2.2** (Theorem 7 of [10]). *Let $A \in \mathbb{R}^{d\times N}$. There exists a distribution over random matrices $S \in \mathbb{R}^{t\times d}$ with $t \in O(\varepsilon^{-2} k \log(N/(\delta k)))$ such that it holds with probability $1 - \delta$ that*

$$\forall x \in \Psi_k : (1-\varepsilon)\|Ax\|_2 \le \|SAx\|_2 \le (1+\varepsilon)\|Ax\|_2.$$

The idea is that there are $\binom{N}{k} \le (eN/k)^k$ choices of k non-zero coordinates out of N. Every choice corresponds to a linear subspace of dimension at most k. By a union bound over all choices, Theorem 2.1 applies with target dimension $t \in O(\varepsilon^{-2} k \log(N/(\delta k)))$ and yields $(1\pm\varepsilon)$ distortion for the Euclidean norm of all vectors in the collection of k-sparse subspaces.

3 Fréchet distance between curves

In this section, we will prove the following proposition, which can be seen as the restriction of Theorem 1.1 to the continuous Fréchet distance. This result is already known [11, 12]. However, our approach is different, in particular considerably simpler and shorter.

We have decided to present this result in detail since it captures the main technical insights in a simple setting. At the same time, it serves as a warm-up for proving Theorem 1.1 in full generality. The high-level idea of this extension will be treated in Section 4, while a detailed proof is deferred to the full version.

▶ **Proposition 3.1.** *Let $\sigma_1, \ldots, \sigma_n : [0,1] \to \mathbb{R}^d$ be n piecewise linear curves with m vertices. There exists a random linear mapping $f : \mathbb{R}^d \to \mathbb{R}^t$ with $t \in O(\varepsilon^{-2} \log(mn/\delta))$ such that, with probability at least $1-\delta$ it holds for all $1 \le i < j \le n$ that*

$$(1-\varepsilon)\, d_{\mathrm{cF}}(\sigma_i, \sigma_j) \le d_{\mathrm{cF}}(f(\sigma_i), f(\sigma_j)) \le (1+\varepsilon)\, d_{\mathrm{cF}}(\sigma_i, \sigma_j),$$

where d_{cF} denotes the continuous Fréchet distance between curves.

Proof. The proof consists of two parts: first, we show that there exists a random linear mapping $S : \mathbb{R}^d \to \mathbb{R}^t$ with $t \in O(\varepsilon^{-2} \log(mn/\delta))$ such that

$$(1-\varepsilon)\|p-q\|_2 \le \|S(p) - S(q)\|_2 \le (1+\varepsilon)\|p-q\|_2$$

for every pair of points $p \in \sigma_i, q \in \sigma_j$ on every pair of curves σ_i, σ_j. Then, we show that this leads to the desired approximation for the continuous Fréchet distance between σ_i and σ_j.

Consider any pair of curves σ_i, σ_j and any pair of points $p \in \sigma_i$ and $q \in \sigma_j$. Denote the vertices of σ_i and σ_j by $p_1, \ldots, p_m$ and $q_1, \ldots, q_m$, respectively. Observe that p and q can be expressed as convex combinations of their neighboring vertices, i.e., there exist $\lambda_p, \lambda_q \in [0,1]$ and indices $h, \ell \in [m-1]$ such that $p = (1-\lambda_p)p_h + \lambda_p p_{h+1}$ and $q = (1-\lambda_q)q_\ell + \lambda_q q_{\ell+1}$. Then, $p - q = (1-\lambda_p)p_h + \lambda_p p_{h+1} - (1-\lambda_q)q_\ell - \lambda_q q_{\ell+1}$ is a linear combination of 4 vertices. Hence, to preserve $\|p-q\|_2$ up to a factor $(1 \pm \varepsilon)$ for all $p \in \sigma_i, q \in \sigma_j$, it suffices to preserve the norm of all linear combinations of 4 vertices of σ_i and σ_j. Let $A \in \mathbb{R}^{d \times mn}$ be the matrix whose columns comprise all vertices of all curves $\sigma_1, \ldots, \sigma_n$. Then, any linear combination of 4 vertices of σ_i and σ_j can be written as Ax for some 4-sparse vector $x \in \mathbb{R}^{mn}$. By setting $k = 4$ and applying Theorem 2.2 with $N = nm$, there exists a random linear mapping $S : \mathbb{R}^d \to \mathbb{R}^t$ with $t \in O(\varepsilon^{-2} \log(mn/\delta))$ such that

$$(1-\varepsilon)\|Ax\|_2 \le \|SAx\|_2 \le (1+\varepsilon)\|Ax\|_2$$

for all 4-sparse x. It follows that

$$(1-\varepsilon)\|p-q\|_2 \le \|S(p) - S(q)\|_2 \le (1+\varepsilon)\|p-q\|_2$$

for every pair of points $p \in \sigma_i, q \in \sigma_j$ on every pair of curves σ_i, σ_j.

Now, to show that S preserves the continuous Fréchet distance between any given pair of curves σ_i and σ_j, let (α^*, β^*) be reparameterizations realizing $d_{\mathrm{cF}}(\sigma_i, \sigma_j)$. Let $r' \in \arg\max_{r \in [0,1]} \|S(\sigma_i(\alpha^*(r))) - S(\sigma_j(\beta^*(r)))\|_2$. Then,

$$\begin{aligned}
d_{\mathrm{cF}}(S(\sigma_i), S(\sigma_j)) &= \inf_{(\alpha,\beta)} \max_{r \in [0,1]} \|S(\sigma_i(\alpha(r))) - S(\sigma_j(\beta(r)))\|_2 \\
&\le \max_{r \in [0,1]} \|S(\sigma_i(\alpha^*(r))) - S(\sigma_j(\beta^*(r)))\|_2 \\
&= \|S(\sigma_i(\alpha^*(r'))) - S(\sigma_j(\beta^*(r')))\|_2 \\
&\le (1+\varepsilon)\, \|\sigma_i(\alpha^*(r')) - \sigma_j(\beta^*(r'))\|_2 \\
&\le (1+\varepsilon) \max_{r \in [0,1]} \|\sigma_i(\alpha^*(r)) - \sigma_j(\beta^*(r))\|_2 \\
&= (1+\varepsilon) \inf_{(\alpha,\beta)} \max_{r \in [0,1]} \|\sigma_i(\alpha(r)) - \sigma_j(\beta(r))\|_2 \\
&= (1+\varepsilon)\, d_{\mathrm{cF}}(\sigma_i, \sigma_j)\,,
\end{aligned}$$

which proves the dilation bound. The contraction bound can be proved in a similar way. ◀

4 Distance measures between piecewise linear surfaces

4.1 Dimension reduction

The proof of Theorem 1.1 is very similar to the special case of the Fréchet distance treated in Section 3. The argument works as follows.

Consider two piecewise linear surfaces. Let p, q be points on some linear piece of either of the two surfaces. By Carathéodory's theorem there exists a convex combination parameterized by $\lambda \in \mathbb{R}^{\gamma+1}$ such that $p = \sum_{i=1}^{\gamma+1} \lambda_i p_i$, where p_i are vertices of a simplex on the linear piece. The same construction holds for q, so the difference vector $v = p - q$ can be expressed as a linear combination of at most $k = 2(\gamma + 1)$ vertices. To preserve the norms of all such linear combinations, we can again apply the k-sparse subspace embedding of [10] with target embedding dimension $t \in O(\gamma \log(nm/(\delta\gamma)))$.

However, to achieve our improved target dimension bound of $t \in O(\varepsilon^{-2}(\gamma + \log(nm/\delta)))$, it is not sufficient to apply the sparse subspace embedding given in Theorem 2.2. Instead, we exploit the geometric structure of the problem to avoid considering simplices whose vertices span the same γ-dimensional subspace multiple times and other overcounting issues that did not matter in the case of polygonal curves, where $\gamma \in O(1)$. The final result then follows by applying Theorem 2.1 with a union bound over the reduced set of subspaces.

Our generalized dissimilarity measure again serves as a general framework for the surface variants of continuous q-DTW, including the limiting cases $q = 1$ (continuous DTW), and $q \to \infty$ (Fréchet). Hausdorff distance can also be handled in a similar way as indicated for curves. Continuous reparameterizations between two surfaces can be handled using homeomorphisms, which require slightly more care. Our proof will appear in the full version.

4.2 Discrete traversals for piecewise linear surfaces

In this subsection, we provide a notion of discrete traversals for surfaces and show that this definition is consistent with discrete traversals for curves. It thus suits the requirements for generalizing the discrete Fréchet distance, and variants of discrete DTW to surfaces.

In contrast with curves, for which there are definitions and algorithms for both the continuous and the discrete Fréchet distance, for surfaces only a continuous Fréchet distance has been defined in the literature [3]. Given two piecewise linear surfaces $f, g : \mathbb{R}^\gamma \to \mathbb{R}^d$, the continuous Fréchet distance between f and g is defined as

$$\inf_{h:\mathbb{R}^\gamma \to \mathbb{R}^\gamma} \sup_{x \in \mathbb{R}^\gamma} \|f(x) - g(h(x))\|_2,$$

where the supremum is taken over all orientable homeomorphisms h from $\mathbb{R}^\gamma$ to itself. This is consistent with the previous definition using reparameterizations by setting h such that $h \circ \alpha = \beta$. For simplicity of notation, we will treat $h \colon \mathbb{R}^d \to \mathbb{R}^d$ as a mapping between the ambient spaces of f and g instead of their parameter spaces.

Recall that in the definition of discrete Fréchet and DTW distances for curves, a minimum is taken over the set of discrete traversals. Hence, a definition for discrete Fréchet and DTW distance between surfaces could be obtained by defining a suitable notion of discrete traversals between surfaces. However, unlike curves, the vertices on a piecewise linear surface do not have a natural ordering that can be used to define a discrete traversal as a sequence. Instead, we define discrete traversals starting from an orientable homeomorphism.

We assume that each surface comes with a fixed 'starting' point, say u^* for f and v^* for g and restrict ourselves to homeomorphisms h satisfying $h(u^*) = v^*$. Let $u_1, \ldots, u_{m'} \in \mathbb{R}^d$ and $v_1, \ldots, v_{m''} \in \mathbb{R}^d$ be the sets of vertices of the surfaces f and g, respectively. A discrete

traversal T_h of the surfaces is a set that consists of pairs of vertices, induced by h in the following way. For each vertex u_i of f, consider the Voronoi cell $V(u_i)$ of u_i within f. Similarly, consider the Voronoi cells $V(v_j)$ for the vertices v_j of g. We add the pair (u_i, v_j) to the traversal T_h (without duplicates) if and only if $h(u_i) \in V(v_j)$ or $h^{-1}(v_j) \in V(u_i)$, where ties lying on common boundaries of two or more distinct Voronoi cells are broken in lexicographic order. A traversal constructed in this way is consistent with the standard definition of a discrete traversal in the special case of polygonal curves. Our proof will appear in the full version.

References

1 Pankaj K Agarwal, Sariel Har-Peled, and Kasturi R Varadarajan. Geometric approximation via coresets. *Combinatorial and computational geometry*, 52(1):1–30, 2005.

2 Meysam Alishahi and Jeff M. Phillips. No dimensional sampling coresets for classification. In *41st International Conference on Machine Learning, (ICML)*, pages 1008–1049, 2024.

3 Helmut Alt and Maike Buchin. Can we compute the similarity between surfaces? *Discrete & Computational Geometry*, 43(1):78–99, 2010. doi:10.1007/S00454-009-9152-8.

4 Mihai Bădoiu, Sariel Har-Peled, and Piotr Indyk. Approximate clustering via core-sets. In *Proc. of the 34th ACM Symposium on Theory of Computing (STOC)*, pages 250–257, 2002.

5 Richard Baraniuk, Mark Davenport, Ronald DeVore, and Michael Wakin. A simple proof of the restricted isometry property for random matrices. *Constructive Approximation*, 28(3):253–263, 2008.

6 Maria Sofia Bucarelli, Matilde Fjeldsø Larsen, Chris Schwiegelshohn, and Mads Toftrup. On generalization bounds for projective clustering. In *Advances in Neural Information Processing Systems 36 (NeurIPS)*, 2023.

7 Emmanuel J Candes, Justin K Romberg, and Terence Tao. Stable signal recovery from incomplete and inaccurate measurements. *Communications on Pure and Applied Mathematics*, 59(8):1207–1223, 2006.

8 David L. Donoho. Compressed sensing. *IEEE Transactions on Information Theory*, 52(4):1289–1306, 2006.

9 Amer Krivošija, Alexander Munteanu, André Nusser, and Chris Schwiegelshohn. Improved learning via k-DTW: A novel dissimilarity measure for curves. In *42nd International Conference on Machine Learning (ICML)*, pages 31696–31726, 2025. URL: https://proceedings.mlr.press/v267/krivosija25a.html.

10 Tung Mai, Alexander Munteanu, Cameron Musco, Anup Rao, Chris Schwiegelshohn, and David P. Woodruff. Optimal sketching bounds for sparse linear regression. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 11288–11316, 2023. URL: https://proceedings.mlr.press/v206/mai23a.html.

11 Ioannis Psarros and Dennis Rohde. Random projections for curves in high dimensions. In *39th International Symposium on Computational Geometry, (SoCG)*, pages 53:1–53:15, 2023. doi:10.4230/LIPICS.SOCG.2023.53.

12 Ioannis Psarros and Dennis Rohde. Random projections for curves in high dimensions. *Discrete & Computational Geometry*, 74(2):374–398, 2025. doi:10.1007/S00454-024-00710-5.

13 Tamás Sarlós. Improved approximation algorithms for large matrices via random projections. In *47th Annual IEEE Symposium on Foundations of Computer Science, (FOCS)*, pages 143–152, 2006. doi:10.1109/FOCS.2006.37.

14 David P. Woodruff. Sketching as a tool for numerical linear algebra. *Foundations and Trends in Theoretical Computer Science*, 10(1-2):1–157, 2014. doi:10.1561/0400000060.

Improved Approximation of Two Watchmen's Routes in Simple Polygons*

Anna Brötzner[1], Bengt J. Nilsson[1], and Christiane Schmidt[2]

1 **Faculty of Technology and Society, Malmö University, Sweden**
{anna.brotzner, bengt.nilsson.TS}@mau.se
2 **Department of Science and Technology, Linköping University, Sweden**
christiane.schmidt@liu.se

Abstract

We study the minmax watchman route problem for two watchmen. We present an approximation algorithm to see a discrete set of m points in the interior of P with approximation ratio $2+\varepsilon$ taking $O(m^5 n)$ time. We generalize the algorithm to see all of the interior of P in $O(n^6)$ time with approximation ratio $2+\pi/2 \approx 3.570$, improving on the previously known best algorithm that has an approximation ratio of ≈ 6.922 and runtime $O(n^2)$ [5].

1 Introduction

The problem of finding a shortest watchman route goes back to Chin and Ntafos, who provided algorithms for rectilinear polygons [2]. For two watchmen, the *two-watchman route* problem, Mitchell and Wynters [4] showed that computing routes with the objective of minimizing the length of the longer route (minmax) is NP-hard. Nilsson and Packer [5] recently provided approximation algorithms for computing minmax two-watchman routes with an approximation ratio of ≈ 5.969 for the floating case taking $O(n^8)$ time, and ≈ 6.922 for the case of two given starting points, not necessarily on the boundary, taking $O(n^2)$ time.

We consider a variant of the two-watchman route problem, where we are given m point shaped treasures in the simple polygon that need to be guarded [1, 3]. We want to compute the shortest two routes V_s and $V_{s'}$, starting at two points s and s', respectively, such that every point $p_i \in \mathcal{S}$ is seen from at least one point on one of the routes. For the minmax-measure, a simple modification of the construction by Mitchell and Wynters [4] shows the NP-hardness of this problem in simple polygons.

2 Preliminaries

Let P be a simple polygon with n vertices, and let $\mathcal{S} = \{p_1, \ldots, p_m\}$ be a set of m points in P, the *treasures*. We assume that no three points among the polygon vertices and $\mathcal{S}$ are collinear. Furthermore, let s and s' be the two boundary starting points for the watchman routes in P. A point $p \in P$ is *visible* to a point $q \in P$ if the segment pq lies in P. For a point $p \in P$, let $VP(p)$ be the *visibility polygon* of p, the set of points in P that are visible to p. An edge of $VP(p)$ that is not contained in an edge of P is called a *window* of p.

Without loss of generality, we may assume that $s, s' \notin VP(p_i)$, for any $p_i \in \mathcal{S}$, as those points are already seen from the starting points. Let C_s^i be the window that separates the subpolygon of $P \setminus VP(p_i)$ containing s from $VP(p_i)$. Now, observe that a watchman who

* All authors were supported by grant 2021-03810 (Illuminate: provably good algorithms for guarding problems) from the Swedish Research Council (Vetenskapsrådet).

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

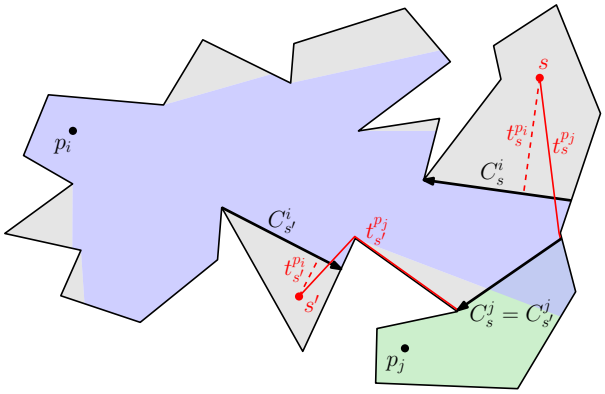


Figure 1 Illustration of the visibility polygon of a point and the windows (black arrows). The windows point towards their corresponding top points. For point p_j, $C_s^j = C_{s'}^j$, hence $p_j \in \mathcal{S}_1$, while for p_i, $C_s^i \neq C_{s'}^i$, thus $p_i \in \mathcal{S}_2$.

starts walking at s sees point p_i as soon as it visits C_s^i. We assign a direction to each window C_s^i so that $VP(p_i)$ lies locally to the left of C_s^i if the polygon is rotated so that the window is directed upwards, see also Figure 1. We denote the lower endpoint of the window the *bottom point* of the window and the upper endpoint the *top point*. We define $C_{s'}^i$ for s' and assign a direction to it analogously.

We study the problem of computing two routes such that each of the treasures can be seen from at least one of the routes. We denote the length of a route W by $\|W\|$. We then extend this problem and require the whole polygon to be seen.

Let W_s and $W_{s'}$ be the shortest routes that sees all the points in $\mathcal{S}$ and pass through s and s', respectively. If s (or s') lies on the boundary of P, we can use a modification of the algorithm by Tan and Jiang [6] to compute W_s ($W_{s'}$) in $O(m^2 n)$ time. If s (s') does not lie on the boundary, we need to modify the algorithm to obtain which window is visited first along a clockwise traversal of the tour starting at s (s'). By trying all possible windows, we can compute the optimal route in $O(m^3 n)$ time.

We denote the minmax shortest pair of routes that together see all the points in $\mathcal{S}$ by $(T_s, T_{s'})$, where T_s and $T_{s'}$ pass through s and s', respectively. We can w.l.o.g. assume that each of T_s and $T_{s'}$ is as short as possible. Furthermore, we define $OPT \stackrel{\text{def}}{=} \max\{\|T_s\|, \|T_{s'}\|\}$.

For each $p_i \in \mathcal{S}$, let $t_s^{p_i}$ be the shortest path from starting point s to $VP(p_i)$, i.e., to the window C_s^i; define $t_{s'}^{p_i}$ analogously for s'. Let $t_{\min}^{p_i} = \arg\min\{\|t_s^{p_i}\|, \|t_{s'}^{p_i}\|\}$ be the shorter of the two paths and let $t_{\max}^{p_i} = \arg\max\{\|t_s^{p_i}\|, \|t_{s'}^{p_i}\|\}$ be the longer of the two paths.

Finding an (approximate) minmax optimal pair of routes requires correctly partitioning the points to be seen between the two routes. Once the partitioning is obtained, the routes can be computed in $O(m^3 n)$ time as previously described.

For brevity, the proofs of statements marked by ($\star$) are omitted in this extended abstract.

3 The Algorithm for Seeing a Discrete Point Set

We describe an algorithm to compute two watchman routes to see a set of treasures $\mathcal{S}$ in a polygon P. This is also the basis for the algorithm for seeing the whole polygon that we present in Section 4.

First, compute a shortest watchman route W_s that sees all points in $\mathcal{S}$ and starts at s,

and a shortest watchman route $W_{s'}$ that sees all points in $\mathcal{S}$ and starts at s'. If the two routes T_s and $T_{s'}$ of an optimal two-watchman route intersect, then either of W_s and $W_{s'}$, together with a route of length 0 at the respective other starting point, gives a 2-approximation. This is because we can trace the subpaths of the routes between their intersection points to obtain a route that sees all the points in $\mathcal{S}$. Thus, $\max\{\|W_s\|, \|W_{s'}\|\} \leq \|T_s\| + \|T_{s'}\| \leq 2 \cdot \max\{\|T_s\|, \|T_{s'}\|\}$. All the remaining steps are under the assumption that the routes T_s and $T_{s'}$ do not intersect.

We partition the points in $\mathcal{S}$ into two sets, depending on where s and s' lie with respect to $VP(p_i)$. Let $\mathcal{S}_1$ be the set of points in $\mathcal{S}$ for which $C_s^i = C_{s'}^i$, and let $\mathcal{S}_2$ be the remaining set of points in $\mathcal{S}$; see also Figure 1 for an illustration. To compute the two watchman routes, we consider $\mathcal{S}_1$ and $\mathcal{S}_2$ separately.

3.1 Handling the Points in $\mathcal{S}_1$

Let $t_s^{p_i}$ and $t_{s'}^{p_i}$ be the shortest paths from s and s', respectively, to $VP(p_i)$. The endpoints of $t_s^{p_i}$ and $t_{s'}^{p_i}$ different from s and s' lie on windows C_s^i and $C_{s'}^i$, respectively. If $C_s^i = C_{s'}^i$, then p_i belongs to the set $\mathcal{S}_1$ and we denote the window C_i.

Consider the set $P \setminus VP(p_i)$ of disjoint subpolygons of P. Let P_i be the subpolygon that contains s and s', for each point $p_i \in \mathcal{S}_1$. If $P_i \subset P_j$, we say that p_j *dominates* p_i, $i \neq j$, since any route through s or s' that visits $VP(p_j)$ must also visit $VP(p_i)$. We call any point in $\mathcal{S}_1$ *dominating* if it is not dominated by any other point in $\mathcal{S}_1$ and assume henceforth that $\mathcal{S}_1$ contains only dominating points.

Let T_s^1 and $T_{s'}^1$ be two min-max length routes that together see all points in $\mathcal{S}_1$. Assume that the windows $C_0, C_1, \ldots, C_{|\mathcal{S}_1|-1}$ in $\mathcal{S}_1$ are ordered as their bottom points appear along the boundary in clockwise order. We can prove: ($\star$) If T_s^1 and $T_{s'}^1$ do not intersect, there exist at most two indices i and j such that T_s^1 visits $C_i, C_{i+1}, \ldots, C_{j-1}$ and $T_{s'}^1$ visits $C_j, C_{j+1}, \ldots, C_{i-1}$.

There are $O(m^2)$ index pairs, and computing the shortest route takes $O(m^3 n)$ for each of these pairs, as described in Section 2. Thus, the time complexity is bounded by $O(m^5 n)$. We state this result as a theorem.

▶ **Theorem 3.1.** *The shortest min-max pair of routes in P that see the points in $\mathcal{S}_1$ can be computed in $O(m^5 n)$ time under the assumption that they do not intersect.*

3.2 Handling the Points in $\mathcal{S}_2$

Let C_s^i and $C_{s'}^i$ be the windows of $VP(p_i)$ that $t_s^{p_i}$ and $t_{s'}^{p_i}$ reach, respectively. If $C_s^i \neq C_{s'}^i$, then p_i belongs to the set $\mathcal{S}_2$. The visibility polygon $VP(p_i)$ must intersect the shortest path $SP(s, s')$ since the starting points lie in different components of $P \setminus VP(p_i)$. Our objective in this section is to show how to compute two routes V_s^ε and $V_{s'}^\varepsilon$ of almost optimal minmax length such that they together see all the points in $\mathcal{S}_2$.

First, we introduce a concept that we will use extensively in the following. Let L be some length $L \leq \|SP(s, s')\|$. If p_L is the point at distance L from s on $SP(s, s')$, we can trace the geodesic quarter circle Q_U of radius L upwards from p_L (this is the connected locus of points having shortest paths in P of length L to s), ending the trace early if it encounters the polygon boundary. Similarly, we can trace the geodesic quarter circle Q_D of radius L downward from p_L; see Figure 2(a). Define $H_s^L = Q_U \cup Q_D$ as the *geodesic half circle* of radius L from s. The region $F_s^L = \{SP(s, p) \mid p \in H_s^L\}$ we call the *funnel* from s of radius L. We symmetrically define the geodesic half circle $H_{s'}^L$ and the funnel $F_{s'}^L$ from s' of radius L.

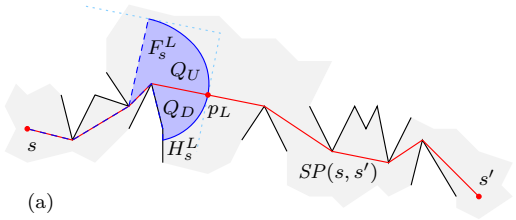


Figure 2 Illustrating the definition of a funnel.

Recall that $t_{\min}^{p_i} = \arg\min\{\|t_s^{p_i}\|, \|t_{s'}^{p_i}\|\}$ for every point $p_i \in \mathcal{S}$, $B = \max_{p_i \in \mathcal{S}_2}\{\|t_{\min}^{p_i}\|\}$, and let F_s^B and $F_{s'}^B$ be the funnels of radius B from s and s', respectively.

▶ **Lemma 3.2** (⋆). For any point $p_i \in \mathcal{S}_2$, either $VP(p_i) \cap F_s^B \neq \emptyset$ or $VP(p_i) \cap F_{s'}^B \neq \emptyset$.

Next, we compute an initial solution for which we can bound the minmax length. Let $\mathcal{S}_s^{\text{std}}$ and $\mathcal{S}_{s'}^{\text{std}}$ be a partitioning of the points in $\mathcal{S}_2$: If $\|t_s^{p_i}\| \leq \|t_{s'}^{p_i}\|$, then $p_i \in \mathcal{S}_2$ belongs to $\mathcal{S}_s^{\text{std}}$, otherwise p_i belongs to $\mathcal{S}_{s'}^{\text{std}}$. We call this the *standard partitioning* of the points in $\mathcal{S}_2$. Each visibility polygon $VP(p_i)$, $\forall p_i \in \mathcal{S}_2$, is at distance at most B from one of s or s'.

Let V_s^{std} and $V_{s'}^{\text{std}}$ be the two shortest routes that see the points in $\mathcal{S}_s^{\text{std}}$ and $\mathcal{S}_{s'}^{\text{std}}$, respectively, that we can compute with the modified method of Tan and Jiang [6].

▶ **Lemma 3.3.** *The length of the two routes V_s^{std} and $V_{s'}^{\text{std}}$ is bounded by*

$$\max\left\{\|V_s^{\text{std}}\|, \|V_{s'}^{\text{std}}\|\right\} \leq \left(1 + \frac{\pi}{2}\right) OPT.$$

Proof. The perimeter of the funnel F_s^B consists of two paths of length B and a sequence of connecting circular arcs with radius $\leq B$; the sum of their lengths is bounded by πB. The perimeter route intersects all the visibility polygons of the points in $\mathcal{S}_s^{\text{std}}$. Since V_s^{std} is the shortest route that intersects these visibility polygons, $\|V_s^{\text{std}}\| \leq (2+\pi)B$. The length of the perimeter of $F_{s'}^B$ is bounded in the same way and since $OPT \geq 2B$, the lemma follows. ◀

Next, we improve on Lemma 3.3 and obtain a $(1+\varepsilon)$-approximation, for any $\varepsilon > 0$. We consider ε to be a constant, sufficiently small, and define a large integer $k = \lceil O(1)/\varepsilon \rceil$.

For any point $p_i \in \mathcal{S}_2$ where $\|t_{\max}^{p_i}\| > 3B$, we can immediately assume that p_i must be seen by the route from the starting point for which the geodesic is shorter. Otherwise, $OPT > 6B$, as the optimal route must at least be as long as the geodesic $t_{\max}^{p_i}$, followed to $VP(p_i)$ and back. In this case, the standard partitioning together with the optimal routes for $\mathcal{S}_1$ gives at worst a 2-approximation to our problem, since $\max\left\{\|V_s^{\text{std}}\|, \|V_{s'}^{\text{std}}\|\right\} \leq (2+\pi)B < 6B \leq OPT$. Let $\mathcal{S}_2^{\text{L}}$ be the subset of the points in $\mathcal{S}_2$ for which $\|t_{\max}^{p_i}\| > 3B$ and assume they are partitioned according to the starting point of their shorter geodesic $t_{\min}^{p_i}$. It remains to find a meaningful partition of the points in $\mathcal{S}_2^{\text{S}} = \mathcal{S}_2 \setminus \mathcal{S}_2^{\text{L}}$.

Define $r \stackrel{\text{def}}{=} \min\{3B, \|SP(s,s')\|/2\}$. For the points $p_i \in \mathcal{S}_2^{\text{S}}$, we have $\|t_{\max}^{p_i}\| \leq r$ by definition, giving us that $\|t_s^{p_i}\|, \|t_{s'}^{p_i}\| \leq r$. Let F_s^r and $F_{s'}^r$ be the two funnels of radius r from s and s', respectively. As $F_s^B \subseteq F_s^r$ and $F_{s'}^B \subseteq F_{s'}^r$, Lemma 3.2 also holds for F_s^r and $F_{s'}^r$.

We make a similarity decomposition of the set of points in $\mathcal{S}_2^{\text{S}}$ into at most k^4 distinct subsets. H_s^r is the geodesic half circle of radius r centered at s; see Figure 3(a). Consider the $k+1$ consecutive equidistant points h_j, for $0 \leq j \leq k$, along H_s^r, h_0 and h_k being the two endpoints of H_s^r. The points of F_s^r between the two geodesics $SP(s, h_{j-1})$ and $SP(s, h_j)$, for each $1 \leq j \leq k$, define a *subfunnel* of F_s^r, denoted $F_s^r(j)$, lightly shaded in Figure 3(a). We can further decompose each subfunnel $F_s^r(j)$ into k cells $c_s^{j,j'}$, for $1 \leq j' \leq k$, depending on

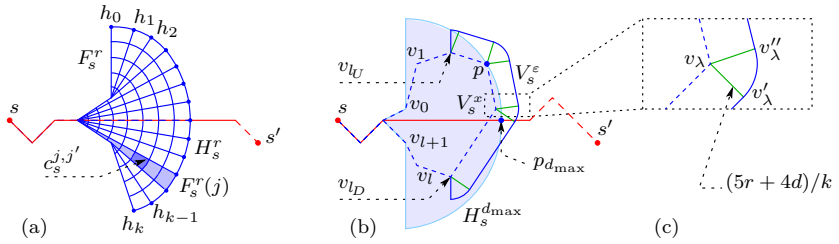


Figure 3 (a) The cells in the similarity decomposition. (b)-(c) Construction of V_s^ε from V_s^x.

the distance to s; this is more heavily shaded in Figure 3(a). The funnel F_s^r thus decomposes into k^2 cells. Similarly, we decompose the funnel $F_{s'}^r$ into k^2 cells $c_{s'}^{j,j'}$, for $1 \leq j, j' \leq k$.

We say that a window C_s^i *belongs to* cell $c_s^{j,j'}$ if the intersection point of $t_s^{p_i}$ and C_s^i is a point in the cell $c_s^{j,j'}$. The point $p_i \in \mathcal{S}_2^{\mathrm{S}}$ lies in a subset defined by four indices j_s, j'_s, $j_{s'}$, and $j'_{s'}$, if C_s^i belongs to $c_s^{j_s,j'_s}$ and $C_{s'}^i$ belongs to $c_{s'}^{j_{s'},j'_{s'}}$. In this way we have partitioned all the points in $\mathcal{S}_2^{\mathrm{S}}$ into at most k^4 distinct subsets.

For each non-empty subset, we select one point as the representative for the subset and let $\mathcal{R} \subseteq \mathcal{S}_2^{\mathrm{S}}$ be the set of all the chosen representative points. By trying all $2^{|\mathcal{R}|} \leq 2^{k^4}$ partitions of the representative points of $\mathcal{R}$ into two subsets, we compute the two shortest tours V_s^x and $V_{s'}^x$ that visit the visibility polygons of the points in $\mathcal{R} \cup \mathcal{S}_2^{\mathrm{L}}$.

Let $d = \max\{\|V_s^x\|, \|V_{s'}^x\|\}$. We construct V_s^ε from V_s^x as follows ($V_{s'}^\varepsilon$ is constructed from $V_{s'}^x$ in the same way). Let $d_{\max}$ be the supremum value for which the geodesic half circle $H_s^{d_{\max}}$ intersects V_s^x, i.e., $d_{\max} = \sup_{L>0}\{H_s^L \cap V_s^x \neq \emptyset\}$. $H_s^{d_{\max}}$ intersects the shortest path from s to s'. Assume that P is rotated so that the intersection point lies on a horizontal segment of $SP(s,s')$, with s locally to the left of s'; see Figure 3(b). Let v_{l_U} and v_{l_D} be the topmost and bottommost convex vertices of V_s^x, respectively.

Let the clockwise subpath from s to v_{l_U} of V_s^x also be the first part of V_s^ε, and the clockwise subpath from v_{l_D} to s of V_s^x also be the last part of V_s^ε. We shift the edges of V_s^x from v_{l_U} to v_{l_D} outwards by a length of $O\big((r+d)/k\big)$ and connect consecutive edges with circular arcs of this same radius, centered at the vertices of V_s^x. Finally, we connect the shifted edges with v_{l_U} and v_{l_D}. See Figures 3(b) and (c) for a detailed illustration. If at some point, the construction intersects the polygon boundary we shortcut the path so that it is the shortest to connect to in the free space of the polygon.

▶ **Lemma 3.4** ($\star$). $(V_s^\varepsilon, V_{s'}^\varepsilon)$ sees all points in $\mathcal{S}_2$ and

$$\max\{\|V_s^\varepsilon\|, \|V_{s'}^\varepsilon\|\} \leq d + O\left(\frac{r+d}{k}\right).$$

Since $r \in O(OPT)$ and $d \leq OPT$, we obtain $\max\{\|V_s^\varepsilon\|, \|V_{s'}^\varepsilon\|\} \leq d + O\big((r+d)/k\big) \leq OPT + O(OPT)/k \leq \big(1 + O(1)/k\big)OPT \leq (1+\varepsilon)OPT$ by our choice of k.

3.3 Putting It All Together

To obtain a $(2+\varepsilon)$-approximation, we consider the two different combinations of routes. Either one watchman that sees all points and one watchman that stays at its starting point; this is one of (W_s, s') and $(s, W_{s'})$. We denote the length of the watchman staying at a point q by $\|q\|$. In the other case, there are two watchmen $(V_s^1 \cup V_s^\varepsilon, V_{s'}^1 \cup V_{s'}^\varepsilon)$ that each visit a

subset of the points in $\mathcal{S}_1$ and in $\mathcal{S}_2$. We then return the shortest pair of routes

$$(V_s, V_{s'}) = \arg\min\Big\{\max\big\{\|W_s\|, \|s'\|\big\}, \max\big\{\|s\|, \|W_{s'}\|\big\}, \max\big\{\|V_s^1 \cup V_s^\varepsilon\|, \|V_{s'}^1 \cup V_{s'}^\varepsilon\|\big\}\Big\}.$$

By combining the algorithms developed in this section together with the bounds in Lemmas 3.3 and 3.4, we immediately have the following theorem.

▶ **Theorem 3.5.** *The two routes V_s and $V_{s'}$ see all points in $\mathcal{S}$, can be computed in $O(m^5 n)$ time, and, for any fixed $\varepsilon > 0$,*

$$\max\big\{\|V_s\|, \|V_{s'}\|\big\} \leq (2+\varepsilon)\, OPT.$$

4 Computing the Shortest Watchman Route That Sees All of P

We can generalize the method obtained in Section 3 and Lemma 3.3 to find two watchman routes $(V_s, V_{s'})$ that together see all of P. To do so, we choose a set $\mathcal{S}$ of $O(n)$ points along the boundary of P with the property that whenever V_s and $V_{s'}$ see these points, they see all of the boundary of P, and for two watchmen, this is enough to see the polygon.

▶ **Theorem 4.1** (Lemma 3.1 in [5])**.** *If two watchmen together see all of the boundary of a polygon P, then they see all of P.*

The set $\mathcal{S}$ comprises of all the vertices of P, the vertices of $VP(s)$ and $VP(s')$, and up to one point p per edge of P for which the shortest path t_s^p from s to see p and the shortest path $t_{s'}^p$ from s' to see p are equally long ($\|t_s^p\| = \|t_{s'}^p\|$).

Finally, as in Section 3, we choose the shortest set of watchman routes as our solution

$$(V_s, V_{s'}) = \arg\min\Big\{\max\big\{\|W_s\|, \|s'\|\big\}, \max\big\{\|s\|, \|W_{s'}\|\big\}, \max\big\{\|V_s^1 \cup V_s^{\text{std}}\|, \|V_{s'}^1 \cup V_{s'}^{\text{std}}\|\big\}\Big\}.$$

▶ **Theorem 4.2** ($\star$)**.** *V_s and $V_{s'}$ see all of P, can be computed in $O(n^6)$ time, and*

$$\max\big\{\|V_s\|, \|V_{s'}\|\big\} \leq \left(2 + \frac{\pi}{2}\right) OPT.$$

References

1 Svante Carlsson and Håkan Jonsson. Guarding a treasury. In *Proceedings of the 5th Canadian Conference on Computational Geometry, Waterloo, Ontario, Canada*, pages 85–90. University of Waterloo, 1993.

2 Wei-pang Chin and Simeon C. Ntafos. Optimum watchman routes. *Inf. Process. Lett.*, 28(1):39–44, 1988. doi:10.1016/0020-0190(88)90141-X.

3 Linda L. Deneen and Shashikant Joshi. Treasures in an art gallery. In *Proceedings of the 4th Canadian Conference on Computational Geometry, St. John's, Newfoundland, Canada*, pages 17–22. Memorial University of Newfoundland, 1992.

4 Joseph S. B. Mitchell and Erik L. Wynters. Watchman routes for multiple guards. In *Proceedings of the 3rd Canadian Conference on Computational Geometry*, pages 126–129, 1991.

5 Bengt J. Nilsson and Eli Packer. Approximation algorithms for the two-watchman route in a simple polygon. *Algorithmica*, 86(9):2845–2884, 2024. doi:10.1007/S00453-024-01245-0.

6 Xuehou Tan and Bo Jiang. Efficient algorithms for touring a sequence of convex polygons and related problems. In *Proceedings of Theory and Applications of Models of Computation, TAMC 2017*, pages 614–627. Springer Verlag, Lecture Notes in Computer Science 10185, 2017. doi:doi.org/10.1007/978-3-319-5.

Simplex volumes in hyperplane arrangements*

Koki Furukawa[1]

1 **Faculty of Mathematics and Physics, Charles University**
koki@kam.mff.cuni.cz

Abstract

The Erdős's distinct distances problem and unit distance problem are classical problems posed by Erdős [7] in 1946, which ask for the minimum number of distinct distances determined by n points in $\mathbb{R}^d$ and the maximum number of pairs at a given distance that such a set can form, respectively. In this paper, we study the following dual setting.

- What is the maximum number of d-simplices of unit, minimum or maximum-volume determined by n hyperplanes in $\mathbb{R}^d$?
- What is the maximum number $D_d(n)$ such that any arrangement of n hyperplanes in $\mathbb{R}^d$ in general position contains a subset of size $D_d(n)$ whose induced d-simplices have all distinct volumes?

Related Version arXiv:2512.12757

1 Introduction

For $d \geq 2$, a set of points X in $\mathbb{R}^d$ with $|X| \geq d+1$ is said to be in *general position* if no $d+1$ points from X lie on the same $(d-1)$-dimensional hyperplane, and we say that an arrangement of hyperplanes $\mathcal{A}$ in $\mathbb{R}^d$ with $|\mathcal{A}| \geq d+1$ is in *general position* if every d hyperplanes intersect at a single point and no $d+1$ have a common point. Throughout the paper, we use O_d to denote the standard big-O notation where the constant may depend on d. We adopt similar conventions for Ω_d and Θ_d.

Erdős's distinct distances problem. In 1946, Erdős [7] posed the *distinct distances problem*, which asks for the minimum number $g_d(n)$ of distinct distances determined by n points in $\mathbb{R}^d$. In the planar case $d = 2$, an integer grid $\sqrt{n} \times \sqrt{n}$ yields the upper bound. While there had been earlier progress on lower bounds, a major breakthrough by Guth and Katz [17] proved $g_2(n) = \Omega(n/\log n)$. In higher dimensions, the d-dimensional lattice $n^{1/d} \times \cdots \times n^{1/d}$ construction implies $g_d(n) = O_d(n^{2/d})$. The best known lower bound to date is $g_d(n) = \Omega_d\big(n^{2/d-2/(d(d+2))}\big)$ due to Solymosi and Vu [24].

A related question asks for the largest integer $h_d(n)$ such that any set of n points in $\mathbb{R}^d$ contains a subset of size $h_d(n)$ whose induced d-simplices all have distinct volumes. Again, a grid construction give the upper bounds $h_2(n) = O(n^{2/3})$ and $h_d(n) = O_d(n^{(d-1)/d})$ for $d \geq 3$. Conlon et al. [3] proved the lower bound $h_d(n) = \Omega_d(n^{1/(2d+2)})$ for all $d \geq 2$.

Erdős's unit distance problem. Erdős's *unit distance problem* asks for the maximum number $u_d(n)$ of unit distances determined by n points in $\mathbb{R}^d$. In the plane, an integer grid $\sqrt{n} \times \sqrt{n}$ gives $u_2(n) > n^{1+c/\log\log n}$ for some $c > 0$. The best known upper bound is $u_2(n) = O(n^{4/3})$, due to Spencer, Szemerédi, and Trotter [25]. In three dimensions, a lattice construction $n^{1/3} \times n^{1/3} \times n^{1/3}$ gives $u_3(n) = \Omega(n^{4/3}\log\log n)$ and one has $u_3(n) = O(n^{3/2})$ [18, 28]. For $d \geq 4$, it is known that $u_d(n) = \Theta_d(n^2)$ [8, 20] holds.

* Supported by grant no. 23-04949X of the Czech Science Foundation (GAR).

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

This problem also has a well-studied direction of generalization. Oppenheim (see [10]) posed the problem of determining the maximum number of unit-area triangles determined by n points in the plane. A lattice construction yields a lower bound of $\Omega(n^2 \log\log n)$ [9], while the best known upper bound is $O(n^{2+2/9})$, due to Raz and Sharir [22]. Dumitrescu and Cs.Tóth [5] extended this problem to three dimensions and obtained the following result: the number of unit volume tetrahedra (3-simplices) determined by n points in $\mathbb{R}^3$ is $O(n^{7/2})$, and there is an n-point set achieving $\Omega(n^3 \log\log n)$.

Dual setting. In this paper, we investigate the following dual setting.

▶ **Problem 1.** *What is the maximum number $f_d(n)$ of unit-volume d-simplices formed by n hyperplanes in $\mathbb{R}^d$?*

▶ **Problem 2.** *What is the maximum number $m_d(n)$ of minimum-volume d-simplices formed by n hyperplanes in $\mathbb{R}^d$?*

For the case $d = 2$, Damásdi et al. [4] proved that $m_2(n) = \Theta(n^2)$.

▶ **Problem 3.** *What is the maximum number $M_d(n)$ of maximum-volume d-simplices formed by n hyperplanes in $\mathbb{R}^d$?*

▶ **Problem 4.** *What is the maximum number $D_d(n)$ such that any arrangement of n hyperplanes in $\mathbb{R}^d$ in general position contains a subset of size $D_d(n)$ whose induced d-simplices have all distinct volumes?*

Damásdi et al. [4] proved the following results for Problems 1-3 for $d = 2$: $f_2(n) = O(n^{9/4+\varepsilon})$ for every fixed $\varepsilon > 0$, while $f_2(n) = \Omega(n^2)$, $7n/5 - O(1) < M_2(n) \le n(n-1)/3$, $m_2(n) = \Theta(n^2)$. For Problem 4, they obtained a lower bound $D_2(n) > n^{1/5}$ under the stronger assumption that there are no six lines that are tangent to a common conic (note that five lines always have a common tangent conic).

Our results We provide partial results for Problems 1-4, especially for $d \ge 3$.

▶ **Theorem 1.** *For the maximum number $f_d(n)$ of unit-volume d-simplices formed by n hyperplanes in $\mathbb{R}^d$, we have*

$$f_d(n) = O_d\big(n^{\,d+1-\frac{d}{d^d+1}}\big).$$

▶ **Theorem 2.** *For the maximum number $f_d(n)$ of unit-volume d-simplices formed by n hyperplanes in $\mathbb{R}^d$, we have*

$$f_d(n) = \Omega_d(n^d).$$

▶ **Theorem 3.** *For the maximum number $m_d(n)$ of minimum-volume d-simplices formed by n hyperplanes in $\mathbb{R}^d$, we have*

$$m_d(n) = \Theta_d(n^d).$$

▶ **Theorem 4.** *For the maximum number $M_3(n)$ of maximum-volume tetrahedra formed by n planes in $\mathbb{R}^3$, we have*

$$M_3(n) > \frac{7}{6}n - O(1).$$

▶ **Theorem 5.** *For the largest subset of hyperplanes forming d-simplices of distinct d-volumes, we have*

$$D_2(n) \ll n(\log n)^{-c}$$

for some absolute constant $c > 0$, For every $d \ge 3$, there is $c_d \in (0,1)$ such that

$$D_d(n) \ll n e^{-(\log\log n)^{c_d}}.$$

Theorem 5 gives partial answers for Problem 5.4 and Problem 5.5 in [4].

2 Unit volume d-simplices

In this section, we consider the number of d-simplices with unit volume determined by an arrangement of n hyperplanes, namely we investigate the magnitude of $f_d(n)$. Clearly, $m_d(n) \leq f_d(n)$ holds, and Theorem 3 gives $f_d(n) = \Omega_d(n^d)$. See the next section. We have a trivial upper bound $O_d(n^{d+1})$ by selecting generic $d+1$ hyperplanes from the arrangement. The following theorem provides a slightly better upper bound.

▶ **Theorem 1.** *For the maximum number $f_d(n)$ of unit-volume d-simplices formed by n hyperplanes in $\mathbb{R}^d$, we have*

$$f_d(n) = O_d\big(n^{\,d+1-\frac{d}{d^d+1}}\big).$$

▶ **Proposition 1.** *Let $H_1, \ldots, H_d$ be hyperplanes in $\mathbb{R}^d$ meeting at a unique point $q \in \mathbb{R}^d$ and fix $V_0 > 0$. Then, every hyperplane that form a d-simplex with $H_1, \ldots, H_d$ of volume V_0 are all tangent to one of two fixed algebraic hypersurfaces of degree d which have $H_1, \ldots, H_d$ as asymptotes and each has exactly 2^{d-1} connected components. Moreover, the 2^d connected components of these two hypersurfaces are contained one each in the 2^d connected components determined by $H_1, \ldots, H_d$.*

A proof for Proposition 1 can be found in the full version [12].

Outline of the proof of Theorem 1. The proof is based on [6]. Let $\mathcal{H}$ be an arrangement of n hyperplanes in $\mathbb{R}^d$. For every d-tuple $(H_{i_1}, \ldots, H_{i_d})$ of hyperplanes in $\mathcal{H}$ with $\bigcap_{j=1}^{d} H_{i_j}$ consisting of a single point, apply the above Proposition with $V_0 = 1$ to obtain the two algebraic hypersurfaces of degree d. Let $\sum_d$ be the union of all such connected components by choosing d hyperplanes intersecting at a single point. Then, we have $|\sum_d| \leq 2^d\binom{n}{d}$. Consider two bipartite graphs G_1, G_2 with vertex set $\mathcal{H} \times \sum_d$ as follows. If $H \in \mathcal{H}$ is tangent to $\sigma \in \sum_d$ from above (resp., below) (in the sense of the last axis), we define an edge $(H, \sigma) \in \mathcal{H} \times \sum_d$ in G_1 (resp.G_2). The edges of G_1 and G_2 correspond to all hyperplane-hypersurface tangencies. Thus, to obtain an upper bound for $f_d(n)$, it suffices to bound the number of edges $E(G_1)$ in G_1 and the number of edges $E(G_2)$ in G_2. By higher-dimensional Bézout's theorem ([27]), we can observe that neither G_1 nor G_2 contains $K_{d^d+1,d}$ as a subgraph. Therefore, by the Kövári-Sós–Turán theorem,

$$|E(G_1)| = O_d\big(n \cdot n^{d(1-\frac{1}{d^d+1})} + n^d\big) = O_d\big(n^{\,d+1-\frac{d}{d^d+1}}\big).$$

The same argument applies to G_2. ◀

3 Minimum volume d-simplices

▶ **Theorem 3.** *For the maximum number $m_d(n)$ of minimum-volume d-simplices formed by n hyperplanes in $\mathbb{R}^d$, we have*

$$m_d(n) = \Theta_d(n^d).$$

Proof of Theorem 3. *Upper bound*: Let $\mathcal{H} = \{H_1, \ldots, H_n\}$ be an arrangement of hyperplanes in $\mathbb{R}^d$, and let m_0 be the minimum volume of a d-simplex determined by $\mathcal{H}$. Assume that there exists a d-tuple $(H_{i_1}, \ldots, H_{i_d})$ of hyperplanes in $\mathcal{H}$ meeting at exactly one point. Applying Proposition 1 to $(H_{i_1}, \ldots, H_{i_d})$ with $V_0 = m_0$, any hyperplane T such that $\{H_{i_1}, \ldots, H_{i_d}, T\}$ forms a d-simplex of volume m_0 must be tangent to one of the two fixed algebraic hypersurfaces of degree d. These two hypersurfaces have 2^d connected components

in total, and each of the 2^d connected components determined by $H_{i_1}, \ldots, H_{i_d}$ contains exactly one of them.

Let C_R be the unique connected component contained in a fixed connected component R in $\{H_{i_1}, \ldots, H_{i_d}\}$. It suffices to show that at most one hyperplane in $\mathcal{H} \setminus \{H_{i_1}, \ldots, H_{i_d}\}$ is tangent to C_R. For a contradiction, we assume that there exist two distinct hyperplanes $T, T' \subset \mathcal{H} \setminus \{H_{i_1}, \ldots, H_{i_d}\}$ tangent to C_R. Let τ be the d-simplex formed by $H_{i_1}, \ldots, H_{i_d}, T$, and let τ' be the d-simplex formed by $H_{i_1}, \ldots, H_{i_d}, T'$. Then τ and τ' are d-simplices of volume m_0.

Now consider the arrangement $\mathcal{A} = \{H_{i_1}, \ldots, H_{i_d}, T, T'\}$. A well-known result of Shannon [23] states that an arrangement of $N(\geq d+1)$ hyperplanes in general position in $\mathbb{R}^d$ has $N-d$ simplicial cells, and hence $\mathcal{A}$ has two simplicial cells. Observe that these two simplicial cells have T as a facet, and they do not lie on the same side of T. Therefore, at least one of them is strictly contained in τ or τ'. This contradicts the minimality of m_0. Hence, we have $m_d(n) \leq 2^d \binom{n}{d} = O_d(n^d)$.

Lower bound: We use the Coxeter–Freudenthal–Kuhn (CFK) triangulation of the unit hypercube $[0,1]^d$, which decomposes $[0,1]^d$ into $d!$ congruent d-simplices, each of volume $1/d!$. See Figure 1. This triangulation is induced by the hyperplanes $\{x_p - x_q = 0 : 1 \leq p < q \leq d\}$. Consider $(n+1)d$ hyperplanes $x_i = k \quad (i = 1, \ldots, d,\ k = 0, \ldots, n)$, which decomposes $[0,n]^d$ into n^d unit hypercubes. By further adding the hyperplanes $\{x_p - x_q = t : 1 \leq p < q \leq d,\ t \in \mathbb{Z},\ |t| \leq n-1\}$, each unit hypercube admits a CFK triangulation (the so-called K1 CFK triangulation). Refer to [21]. The total number of hyperplanes in the resulting arrangement is $d(n+1) + \binom{d}{2}(2n-1)$ and the number of resulting minimum-volume d-simplices is $d!\, n^d$. This construction gives a lower bound of $\Omega_d(n^d)$ on the number of minimum-volume d-simplices.

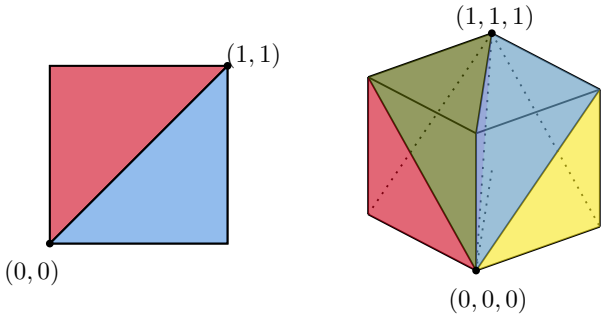


Figure 1 The CFK triangulation of $[0,1]^2$ and $[0,1]^3$.

◀

4 Maximum volume tetrahedra

In this section, we give a lower bound for $M_3(n)$. A linear lower bound for $M_3(n)$ can be easily obtained using three planes H_1, H_2, H_3 together with $n-3$ tangent planes to a hypersurface having H_1, H_2, H_3 as asymptotic. By generalizing the argument in [4] to 3 dimensions, we obtain the following lower bound.

▶ **Theorem 4.** *For the maximum number $M_3(n)$ of maximum-volume tetrahedra formed by n planes in $\mathbb{R}^3$, we have*

$$M_3(n) > \frac{7}{6}n - O(1).$$

For a plane arrangement $\mathcal{A}$, let $T(\mathcal{A})$ be the number of maximum volume tetrahedra in $\mathcal{A}$.

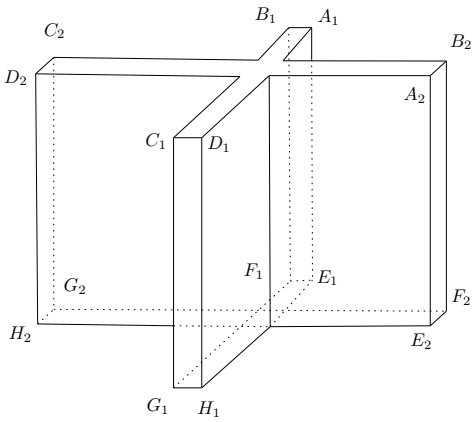


Figure 2 Rectangular boxes

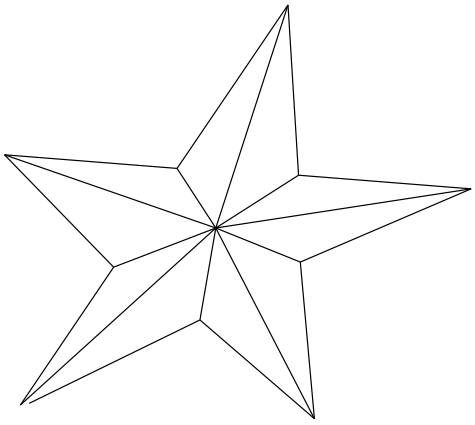


Figure 3 A star-shaped badge

▶ **Proposition 2** (3-dimensional analogue of Proposition 4.2 in [4]). *If $\mathcal{H}_1$ and $\mathcal{H}_2$ are arrangements of planes neither of which contains two parallel planes, then there exists affine transformations φ, ψ such that*

$$T(\varphi(\mathcal{H}_1) \cup \psi(\mathcal{H}_2)) \geq T(\mathcal{H}_1) + T(\mathcal{H}_2) + 2.$$

Outline of the proof. We apply affine transformations φ and ψ to two arrangements $\mathcal{H}_1$ and $\mathcal{H}_2$ to equalize the volumes of their respective maximum-volume tetrahedra and position their rectangular boxes to intersect as shown in Figure 2. By translating planes of $\psi(\mathcal{H}_2)$, we create a new maximum-volume tetrahedron S. Depending on whether 1, 2, or 3 of the four planes defining S belong to $\varphi(\mathcal{H}_1)$, we can create additional maximum-volume tetrahedra by moving the entire arrangement along an appropriate plane or line. ◀

We observe that the arrangement of planes $\mathcal{S} \subset \mathbb{R}^3$ forming a star-shaped badge as shown in Figure 3 consists of 6 planes forming 5 maximum volume tetrahedra. Starting from $\mathcal{S}$, we repeatedly add suitably transformed copies of $\mathcal{S}$ so that the above proposition applies at each step. Consequently, for every $k \geq 1$ we obtain an arrangement of $6k$ planes in $\mathbb{R}^3$ that determines at least $7k - 2$ maximum-volume tetrahedra, which proves Theorem 4.

We are not aware of any upper bound for $M_d(n)$ that holds for all $d \geq 3$. Moreover, we do not have any reason to believe that the correct upper bound is also close to linear. Whether $M_d(n)$ is linear in n is open even in the case $d = 2$. Note that Damásdi et al. [4] conjectured that $M_2(n) = O(n^{1+\varepsilon})$ for every $\varepsilon > 0$.

5 Hyperplanes defining distinct volume d-simplices

In this section, we give an upper bound construction for $D_d(n)$.

▶ **Theorem 5.** *For the largest subset of hyperplanes forming d-simplices of distinct d-volumes, we have*

$$D_2(n) \ll n(\log n)^{-c}$$

for some absolute constant $c > 0$, For every $d \geq 3$, there is $c_d \in (0, 1)$ such that

$$D_d(n) \ll ne^{-(\log \log n)^{c_d}}.$$

Let $[n] = \{1, \ldots, n\}$ and denote by $r_k(n)$ the size of the largest $S \subseteq [n]$ such that S contains no k elements in arithmetic progression. Here is the key proposition.

▶ **Proposition 3.** *$D_d(n) < r_{d+2}(n)$ for all $d \geq 2$.*

In 2009, Green and Tao [15, 16] proved that there exists a constant $c > 0$ such that $r_4(n) \ll n(\log n)^{-c}$. For $k \geq 4$, Leng et al. [19] recently obtained an upper bound $r_k(n) << ne^{-(\log\log n)^{c_k}}$ for some $c_k \in (0, 1)$. See [26, 13, 14] for further details. Theorem 5 can be immediately derived from these bounds and Proposition 3. Therefore, any improvement of upper bound for $r_k(n)$ automatically improves that for $D_d(n)$.

Outline of the proof of Proposition 3. First assume that d is even. Write $\mathbb{R}^d$ as the orthogonal sum of two-dimensional subspaces: $\mathbb{R}^d = \bigoplus_{j=1}^{d/2} V_j$. For each j, let C_j denote the unit circle centered at the origin in V_j. We choose angle parameters $\theta_1, \ldots, \theta_{d/2} > 0$ so that for each j, one has $n\theta_j \ll \pi/2$, and $\theta_j \neq \theta_{j'}$ for $j \neq j'$. For each $i \in \{0, \ldots, n\}$ and each $j \in \{1, \ldots, d/2\}$, we define $P_{i,j} = (\cos(i\theta_j), \sin(i\theta_j)) \in V_j$ (viewed in the standard coordinates of $\mathbb{R}^d$, the point $P_{i,j}$ has nonzero coordinates only in its V_j-component, which is $(\cos(i\theta_j), \sin(i\theta_j))$, while all other components are zero). For each $i \in \{1, \ldots, n\}$, let H_i denote the affine hull of $\bigcup_{1 \leq j \leq d/2} \{P_{i-1,j}, P_{i,j}\}$. Let $\mathcal{H} = \{H_1, \ldots, H_n\}$ denote the resulting hyperplane arrangement. One can verify that $\mathcal{H}$ is in general position for almost all choices of the angle parameters $\theta_1, \ldots, \theta_{d/2}$. We can check that for any $(d+2)$-arithmetic progression $\{i_1 < i_2 < \cdots < i_{d+2}\}$ in $[n]$, $\{H_{i_1}, \ldots, H_{i_{d+1}}\}$ and $\{H_{i_2}, \ldots, H_{i_{d+2}}\}$ both form the d-simplex of same volume which shows that any subfamily of $\mathcal{H}$ with size $r_{d+2}(n) + 1$ contains two d-simplices with the same volume. The proof is given in a full version [12].

Now, we assume d is odd, For odd d, we argue as in the even case, but work in $\mathbb{R}^d = \bigoplus_{j=1}^{(d-1)/2} V_j \oplus L$ with $\dim V_j = 2$ and $\dim L = 1$. Choose distinct small angles $\theta_1, \ldots, \theta_{(d-1)/2}$ and replace each circle $C_j \subset V_j$ by a helix in $V_j \oplus L$ whose L-coordinate equals the parameter t. Taking the points with $t = 0, 1, \ldots, n$ on each helix together with $Q_i = i \in L$, we define H_i as the affine hull of the corresponding pairs at levels $i - 1$ and i (and Q_i). See the full version [12] for a detailed proof. ◀

References

1. P. Braß, G. Rote, and K. J. Swanepoel. Triangles of extremal area or perimeter in a finite planar point set. *Discrete & Computational Geometry*, 26(1):51–58, 2001.
2. G. Clément and J. Bader. Tighter upper bound for the number of Kobon triangles. *Preprint*, 6066, 2007. Preprint; available online.
3. D. Conlon, J. Fox, W. Gasarch, D.G. Harris, D. Ulrich, and S. Zbarsky. Distinct volume subsets. *SIAM Journal on Discrete Mathematics*, 29(1):472–480, 2015.
4. G. Damásdi, L. Martínez-Sandoval, D. T. Nagy, and Z. L. Nagy. Triangle areas in line arrangements. *Discrete Mathematics*, 343(12):112105, 2020.
5. A. Dumitrescu and C.D. Tóth. On the number of tetrahedra with minimum, unit, and distinct volumes in three-space. *Combinatorics, Probability and Computing*, 17(2):203–224, 2008.
6. Adrian Dumitrescu, Micha Sharir, and Csaba D Tóth. Extremal problems on triangle areas in two and three dimensions. *Journal of Combinatorial Theory Series A*, 116(7):1177–1198, 2009.
7. P. Erdős. On sets of distances of n points. *The American Mathematical Monthly*, 53(5):248–250, 1946.
8. P. Erdős and J. Pach. Variations on the theme of repeated distances. *Combinatorica*, 10(3):261–269, 1990.

9 P. Erdős and G. Purdy. Some extremal problems in geometry. *Journal of Combinatorial Theory, Series A*, 10(3):246–252, 1971.

10 Paul Erdős and George Purdy. Extremal problems in combinatorial geometry. In *Handbook of combinatorics (vol. 1)*, pages 809–874. 1996.

11 Z. Füredi and I. Palásti. Arrangements of lines with a large number of triangles. *Proceedings of the American Mathematical Society*, 92(4):561–566, 1984.

12 K. Furukawa. Simplex volumes in hyperplane arrangements. *arXiv preprint arXiv:2512.12757*, 2025.

13 W. T. Gowers. A new proof of Szemerédi's theorem for arithmetic progressions of length four. *Geometric & Functional Analysis (GAFA)*, 8(3):529–551, 1998.

14 W. T. Gowers. Arithmetic progressions in sparse sets. *Current Developments in Mathematics*, 2000(1):149–196, 2000.

15 B. Green and T. Tao. New bounds for Szemerédi's theorem, ii: A new bound for $r_4(n)$. *Analytic Number Theory*, pages 180–204, 2009.

16 B. Green and T. Tao. New bounds for Szemerédi's theorem. iii: A polylogarithmic bound for $r_4(n)$. *Mathematika*, 63(3), 2017.

17 L. Guth and N. H. Katz. On the Erdős distinct distances problem in the plane. *Annals of Mathematics*, pages 155–190, 2015.

18 H. Kaplan, J. Matoušek, Z. Safernová, and M. Sharir. Unit distances in three dimensions. *Combinatorics, Probability and Computing*, 21(4):597–610, 2012.

19 J. Leng, A. Sah, and M. Sawhney. Improved bounds for Szemerédi's theorem. *arXiv preprint arXiv:2402.17995*, 2024.

20 H. Lenz. Zur zerlegung von punktmengen in solche kleineren durchmessers. *Archiv der Mathematik*, 6(5):413–416, 1955.

21 L. Martinelli Reia, M. Gameiro, T. Bueno Moraes Ribeiro, and A. Castelo. A fast high-dimensional continuation hypercubes algorithm. *Computers & Graphics*, 129:104237, 2025.

22 O. E. Raz and M. Sharir. The number of unit-area triangles in the plane: Theme and variation. *Combinatorica*, 37(6):1221–1240, 2017.

23 R.W. Shannon. Simplicial cells in arrangements of hyperplanes. *Geometriae Dedicata*, 8(2):179, 1979.

24 J. Solymosi and V. H. Vũ. Near optimal bounds for the Erdős distinct distances problem in high dimensions. *Combinatorica*, 28(1):113–125, 2008.

25 E. Spencer, H. Szemeredi and W. T. Trotter. Unit distances in the Euclidean plane. pages 293–308. Academic Press, 1984.

26 E. Szemerédi. On sets of integers containing no k elements in arithmetic progression. *Acta Arithmetica*, 27:199–245, 1975.

27 T. Tao. Bezout's inequality. (post in blog "what's new"), 2011. URL: `https://terrytao.wordpress.com/2011/03/23/bezouts-inequality/`.

28 J. Zahl. An improved bound on the number of point-surface incidences in three dimensions. *Contributions to Discrete Mathematics*, 8(1), 2013.

What induces plane structures in complete graph drawings?*

Alexandra Weinberger[1] and Ji Zeng[2]

1 **FernUniversität in Hagen, Germany**
`alexandra.weinberger@fernuni-hagen.de`
2 **Alfréd Rényi Institute of Mathematics, Hungary**
`zeng.ji@renyi.hu`

Abstract

This paper considers the task of connecting points on a piece of paper by drawing a curve between each pair of them. Under mild assumptions, we prove that many pairwise disjoint curves are unavoidable if either of the following rules is obeyed: any two adjacent curves do not cross, or any two non-adjacent curves cross at most once. Here, two curves are called adjacent if they share an endpoint. On the other hand, we demonstrate how to draw all curves such that any two adjacent curves cross exactly once, any two non-adjacent curves cross at least once and at most twice, and thus no two curves are disjoint. Furthermore, we analyze the emergence of disjoint curves without these mild assumptions, and characterize the plane structures in complete graph drawings guaranteed by each of the rules above.

Related Version Full version available at arXiv:2603.05208 [39].

1 Introduction

In the plane, a *complete graph drawing* is a collection of points and curves connecting each pair of these points. The points are called vertices and the curves are called edges of this drawing. More precisely, an edge is the image of a piecewise smooth map from a closed interval on the real line to the plane which starts and ends at its corresponding pair of vertices. In this paper, we make *mild assumptions* for complete graph drawings: we assume no edge contains vertices other than its endpoints; we assume no edge self-intersects, that is, its defining map must be injective; we assume edges cross properly whenever they intersect, in other words, edges do not merely touch. It is easy to imagine that a complete graph drawing can be tangled like a mass of spaghetti noodles. Our main result asserts that the breaking point of "tangledness" in complete graph drawings is the logical disjunction of the two rules below.

Adjacent-simple: any two adjacent edges do not cross;

Separate-simple: any two non-adjacent edges cross at most once.

▶ **Theorem 1.1.** *For any positive integer m, there exists n such that every complete graph drawing of n vertices that is either adjacent-simple or separate-simple must contain m pairwise disjoint edges. On the other hand, there exist complete graph drawings on arbitrarily many vertices in which any two adjacent edges cross exactly once, and any two non-adjacent edges cross at least once and at most twice.*

* Ji Zeng is supported by ERC Advanced Grants "GeoScape", no. 882971 and "ERMiD", no. 101054936, and is also partly supported by NSF grant DMS-1928930 while in residence at the Simons–Laufer Mathematical Sciences Institute during the Spring 2025 semester. We wish to thank Jeck Lim, Jonathan Rollin, and André Schulz for helpful discussions, and an anonymous reviewer who provided the idea to simplify Lemma 3.1.

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

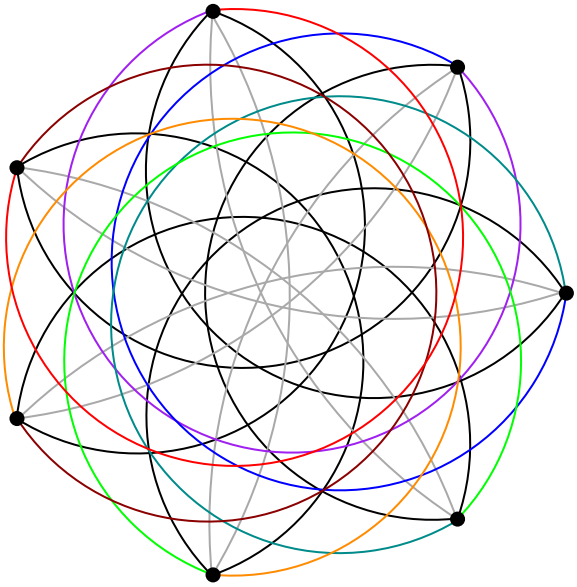


Figure 1 Any two adjacent edges cross exactly once, any two non-adjacent edges cross at least once and at most twice.

Drawings that are only adjacent-simple or only separate-simple, have been studied under the names semi-simple [5] and star-simple [12], or tolerable [10] respectively.[1]

In general, a *graph* in the abstract sense consists of a vertex set and an edge set, where each edge is a vertex pair (unordered). A *drawing* of a graph is a representation of its vertices as points in the plane, and its edges as curves connecting the corresponding pair of vertices. We make the same mild assumptions for all drawings of graphs. A drawing is said to be *plane* if it has no crossings. Planarity has been a main theme in the development of graph theory featured by famous theorems such as Euler's formula and Kuratowski's theorem. A drawing is said to be *simple* if it is adjacent-simple and separate-simple simultaneously. Rafla [28] conjectured that every simple complete graph drawing on at least three vertices must contain a plane Hamiltonian cycle, that is, a cycle that visits every vertex exactly once. Motivated by this conjecture or not, unavoidable plane structures in simple drawings are extensively studied. We emphasize the work of Pach and Tóth in 2010 [26], in which they proved that simple complete graph drawings induce many pairwise disjoint edges and constructed complete graph drawings where any two edges cross at least once and at most twice. However, some adjacent edges cross twice in their construction. Much research on plane structures in simple drawings has been done since, with several papers focused on improving the quantitative values, see our incomplete list on plane structures in simple drawings [1, 3, 15, 16, 17, 18, 26, 30, 34, 36, 40]. Further work on unavoidable plane structures also addresses simple complete graph drawings that satisfy additional conditions [4, 7, 9], and especially those where the edges have to be straight [23, 32, 38]. In this work, we follow a different direction and rather than aiming at quantity or putting additional conditions,

[1] Since we consider both types of drawings, we use adjacent-simple and separate-simple because it gives more intuition to which condition is meant than semi-simple and tolerable, and seems more intuitive than matching-simple (which would be the pendant to star-simple).

this paper attempts to capture the earliest point at which one can expect unavoidable plane structures. With respect to the conditions, Theorem 1.1 can be considered a generalization of the result by Pach and Tóth [26].

We shall state the two cases of Theorem 1.1 as theorems on their own. We call a graph S a *plane structure guaranteed* in a collection $\mathcal{C}$ of graph drawings if there exists an integer n such that any drawing in $\mathcal{C}$ of at least n vertices must contain a plane sub-drawing of S. Here, a drawing A is called a sub-drawing of another drawing B if the vertex set and edge set of A are subsets of those of B respectively. A *squid* is a connected graph that contains exactly one cycle which is a triangle, and all other vertices only form edges with two fixed vertices on this triangle. A *caterpillar* is a connected graph without cycles that contains a central path, and all other vertices only form edges with vertices on this path. An *isolated vertex* in a graph is a vertex that does not form any edges.

Figure 2 A squid (left) and a caterpillar (right).

▶ **Theorem 1.2.** *The plane structures guaranteed in adjacent-simple complete graph drawings are exactly squids with extra isolated vertices, and unions of disjoint caterpillars.*

▶ **Theorem 1.3.** *The plane structures guaranteed in separate-simple complete graph drawings are exactly unions of disjoint edges and isolated vertices.*

Naturally, one may ask to what extent our main result still holds if the mild assumptions are dropped. To denote an edge in a drawing without mild assumptions, we use the image $\mathrm{Im}(f)$ of a function f from the unit interval $[0,1]$ to the plane $\mathbb{R}^2$. We also define self-intersection of f as $\Delta(f) = \{p : p = f(x) = f(y) \text{ for } x \neq y\}$. A family of functions $\{f_i\}$ is called *stroke-like* if all the pre-images $f_i^{-1}(\mathrm{Im}(f_j))$ and $f_i^{-1}(\Delta(f_i))$ are unions of finitely many sub-intervals of $[0,1]$. Although mathematically two curves can intersect infinitely many times without coincide along any segment, it is not physically possible to draw non-stroke-like curves on a piece of paper using a pen. We can generalize Theorem 1.1 to the following statement.

▶ **Corollary 1.4.** *For any positive integer m, there exists n such that the following is true. Let $p_1, p_2, \ldots, p_n$ be distinct points in the plane, $c_{ij} : [0,1] \to \mathbb{R}^2$ be piecewise smooth function with $c_{ij}(0) = p_i$ and $c_{ij}(1) = p_j$ for all $1 \leq i < j \leq n$, and suppose the family $\{c_{ij}\}$ is stroke-like. If either of the following conditions holds:*

(A) *$\forall\, i < j$ and $k < \ell$ with $|\{i,j,k,\ell\}| = 3$, we have $|c_{ij}^{-1}(Im(c_{k\ell}))| = |c_{k\ell}^{-1}(Im(c_{ij}))| = 1$;*

(S) *$\forall\, i < j$ and $k < \ell$ with $|\{i,j,k,\ell\}| = 4$, we have $|c_{ij}^{-1}(Im(c_{k\ell}))| = |c_{k\ell}^{-1}(Im(c_{ij}))| \leq 1$, and there is no touching point between c_{ij} and $c_{k\ell}$;*

then there exist m functions in $\{c_{ij} : 1 \leq i < j \leq n\}$ whose images are pairwise disjoint.

Conditions (A) and (S) are similar to adjacent-simple and separate-simple drawings, respectively, however, less assumptions are used. Due to the mild assumptions of adjacent-simple drawings and separate-simple drawings, all touchings are forbidden – in particular, also non-adjacent edges are forbidden from touching in adjacent-simple drawings, and also adjacent

edges are forbidden from touching in separate simple drawings – and edges are forbidden from passing through any vertices – in particular, in separate-simple drawings the end-vertices of an edge are forbidden from lying on its interior. Those restrictions are not made in Corollary 1.4.

Outline Sections 2 and 3 are devoted to the proofs of Theorem 1.2 and Theorem 1.3 respectively. In Section 4, we prove Theorem 1.1 and Corollary 1.4 and show that in many ways the conditions of Corollary 1.4 are necessary. We conclude in Section 5 with some pointers to open directions for further research. *Proofs can be found in the full version [39].*

2 The adjacent-simple case

The starting point of the proof of Theorem 1.2 is the statement below. It can be shown by analyzing the partial drawings induced by the initial edge segments from vertices until the edge is crossed the first time.

▶ **Lemma 2.1.** *There exists at least one pair of disjoint edges in every adjacent-simple complete graph drawing on four vertices.*

We call an adjacent-simple complete graph drawing *Type I*, *Type II*, or *Type III* if we can order its vertices as $v_1, v_2, \ldots, v_n$ such that conditions I, II, or III below are satisfied respectively:

I. v_i, v_j, v_k, v_ℓ induce only one pair of disjoint edges v_iv_j and v_kv_ℓ for all $i < j < k < \ell$.
II. v_i, v_j, v_k, v_ℓ induce only one pair of disjoint edges v_iv_k and v_jv_ℓ for all $i < j < k < \ell$.
III. v_i, v_j, v_k, v_ℓ induce only one pair of disjoint edges v_iv_ℓ and v_jv_k for all $i < j < k < \ell$.

See Figures 3, 4, and 5 for illustrations of type-drawings.

▶ **Lemma 2.2.** *There exist adjacent-simple complete graph drawings of Type I, Type II, and Type III with arbitrarily many vertices.*

Our final ingredient for Theorem 1.2 is the celebrated theorem of Ramsey [29].

▶ **Theorem 2.3.** *For any positive integers k, m, r, there exists an integer n with the following property. For every partition of all size-r subsets of a size-n set S into k classes, there exists a size-m subset S' of S such that all size-r subsets of S' belong to the same class in this partition.*

Proof Sketch of Theorem 1.2. For the first part of Theorem 1.2, we consider an arbitrary adjacent-simple complete graph drawing. We partition its quadruples of vertices into three classes depending on their non-crossing edge pairs. At least one pair of edges that does not cross exists by Lemma 2.1. By Theorem 2.3, there is a subdrawing in which all quadruples are in the same class. For each of the three classes, these subdrawings contain a plane drawing of each structure with at most as many vertices that is either a squid or a union of disjoint caterpillars whose number of vertices is at most the number of vertices in the subdrawing. All other structures are not contained in one of the type-drawings, which exist by Lemma 2.2. This implies the second part of Theorem 1.2. ◀

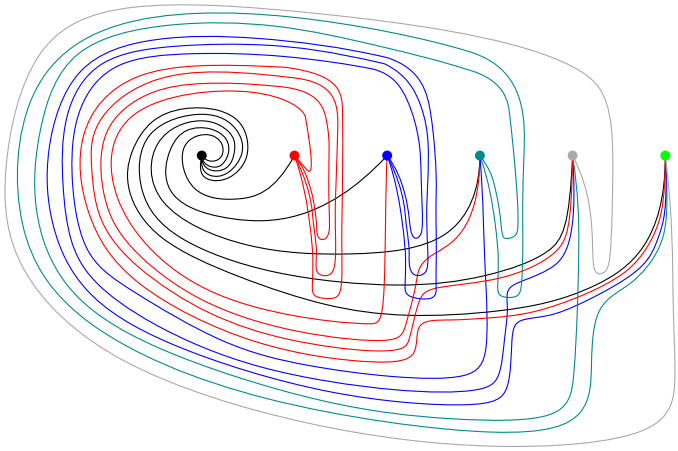


Figure 3 A Type I drawing of vertices $v_1, v_2, \ldots, v_6$ from left to right.

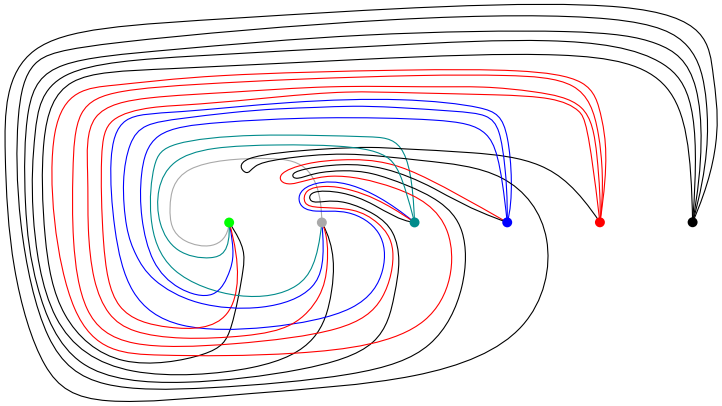


Figure 4 A Type II drawing of vertices $v_1, v_2, \ldots, v_6$ from left to right.

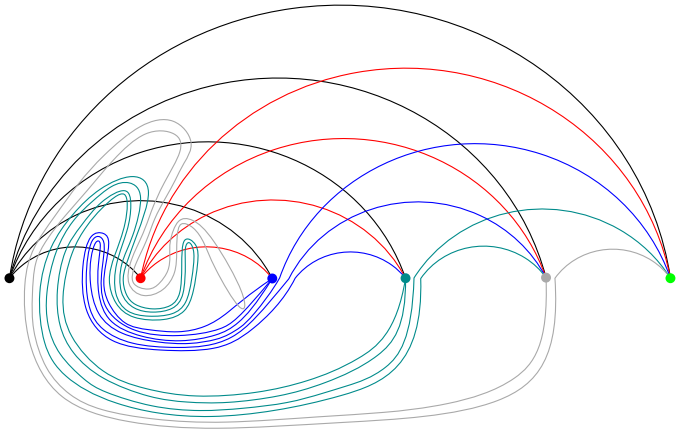


Figure 5 A Type III drawing of vertices $v_1, v_2, \ldots, v_6$ from left to right.

3 The separate-simple case

Our proof of Theorem 1.3 starts from Lemma 3.1. A *cell* of a drawing (not necessarily plane) refers to a connected component of the complement of this drawing inside the plane.

▶ **Lemma 3.1.** *In any separate-simple drawing of a triangle abc union an edge de, such that the vertices d, e are in the same cell of abc, the edge de does not cross one of ab, bc, ac.*

Proof Sketch. The cells of the triangle abc admits a "chessboard" coloring in the sense that they can be assigned black and white and adjacent cells have different colors. Since d, e are located over the same color, a parity argument shows de must crosses the edges of abc odd number of times. Hence the lemma follows from separate-simplicity. ◀

We then guarantee the existence of one disjoint pair of edges.

▶ **Lemma 3.2.** *There exists at least one pair of disjoint edges in every separate-simple complete graph drawing of forty-three vertices.*

Proof Sketch. Consider two triangles, abc and def selected from the forty-three vertices. By studying the interaction between abc and def, their possible crossings, and the behavior of cells, we can bound the number of cells in abc that are not contained in a cell of def and the number of cells in def that contain a cell of abc. Counting arguments then insure that a cell of either abc or def must contain at least two vertices. By Lemma 3.1, this guarantees a pair of disjoint edges. ◀

Proof Sketch of Theorem 1.3. The first part of Theorem 1.3 can be proven by Lemma 3.2 and Theorem 2.3 and arguing which edges of the sub-drawing obtained by Theorem 2.3 are pairwise disjoint. The second part can be proven redrawing the initial parts of all edges of any arbitrary simple drawing such that all adjacent edges cross; see Figure 6. ◀

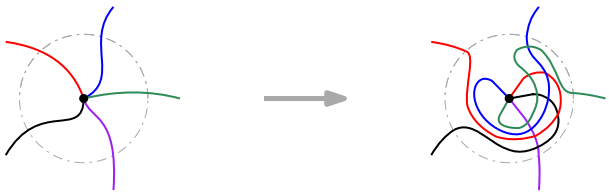


Figure 6 Redrawing the edges around a vertex to make them cross each other.

4 Conditions for plane structures

We now put the previous results together to obtain Theorem 1.1 and Corollary 1.4.

Proof Sketch of Theorem 1.1. The first part of Theorem 1.1 follows from combining Theorem 1.2 and Theorem 1.3. The second part of Theorem 1.1 can be shown by extending the drawing in Figure 1, in which all adjacent edges cross, to arbitrarily many vertices. ◀

Proof Sketch of Corollary 1.4. If Condition (A) holds, we can obtain an adjacent-simple drawing by making modifications that do not change the intersection relation between distinct edges. If Condition (S) holds, we can obtain a separate-simple drawing by making modifications that do not change the intersection relation between adjacent edges. Thus Corollary 1.4 can be derived from Theorem 1.1. ◀

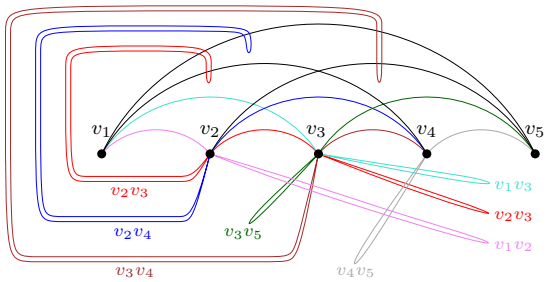


Figure 7 A loop labeled v_iv_j is part of the edge between vertices v_i and v_j.

We cannot relax Condition (A) to allow adjacent edges to touch nor relax Condition (S) to allow non-adjacent edges to touch, or a degenerate case of Figure 1 will lead to counterexamples. We cannot relax the equation in Condition (A) to $|\mathrm{Im}(c_{ij}) \cap \mathrm{Im}(c_{k\ell})| = 1$, because we need to forbid edges from passing through their own endpoints. Otherwise, a construction like in Figure 7 gives a counterexample. In contrast, perhaps interestingly, edges can pass through endpoints in Condition (S) for Corollary 1.4.

5 Conclusion

We gave sufficient conditions for the appearance of disjoint curves in complete graph drawings, but also (with Corollary 1.4) in constructions that are usually not considered as drawings. We showed that in many ways the conditions of Corollary 1.4 are necessary. Still, Corollary 1.4 may not capture the exact cause of disjoint curves in the process of connecting points on paper. For example, we did not study the scenario that forbids non-adjacent curves to touch but allows adjacent curves to touch. In fact, we did not consider whether a rule like "at least 64% pairs of non-adjacent edges cross less than twice" would force a pair of disjoint edges in complete graph drawings. We leave these obscure and interesting questions open.

For adjacent-simple and separate-simple drawings, we gave the specific plane structures guaranteed. It seems natural to ask for a characterization of the plane structures guaranteed in simple complete graph drawings. According to a result of Pach, Solymosi, and Tóth [21], there are two canonical sub-structures in large simple complete graph drawings: convex geometric drawings and twisted drawings. Based on their work, one can easily argue that the plane structures guaranteed in simple complete graph drawings are graphs with stack number and queue number both being one. We refer our readers to [11, 20, 27] for information on stack and queue numbers. It is known that graphs with stack number one have nice characterizations [37] and can be recognized in linear time, but the task of recognizing graphs with queue number one is NP-complete [19]. It seems to be an open problem whether graphs with stack number and queue number both being one can be recognized efficiently.

References

1 Oswin Aichholzer, Alfredo García, Irene Parada, Birgit Vogtenhuber, and Alexandra Weinberger. Shooting stars in simple drawings of $K_{m,n}$. In *Proceedings of the 30th International Symposium on Graph Drawing and Network Visualization (GD 2022)*, pages 49–57. Springer, 2022. doi:10.1007/978-3-031-22203-0_5.

2 Oswin Aichholzer, Alfredo García, Javier Tejel, Birgit Vogtenhuber, and Alexandra Weinberger. Twisted ways to find plane structures in simple drawings of complete graphs. In *Proceedings of the 38th International Symposium on Computational Geometry (SoCG 2022)*, pages 5:1–5:18, 2022. doi:10.4230/LIPICS.SOCG.2022.5.

3 Oswin Aichholzer, Alfredo García, Javier Tejel, Birgit Vogtenhuber, and Alexandra Weinberger. Twisted ways to find plane structures in simple drawings of complete graphs. *Discrete & Computational Geometry*, 71(1):40–66, 2024. *Conference Version: [2].* doi:10.1007/S00454-023-00610-0.

4 Oswin Aichholzer, Joachim Orthaber, and Birgit Vogtenhuber. Separable drawings: Extendability and crossing-free Hamiltonian cycles. In *Prooceedings of the 32nd International Symposium on Graph Drawing and Network Visualization (GD 2024)*, pages 34:1–34:17. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2024. doi:10.4230/LIPICS.GD.2024.34.

5 Martin Balko, Radoslav Fulek, and Jan Kynčl. Crossing numbers and combinatorial characterization of monotone drawings of K_n. *Discrete & Computational Geometry*, 53(1):107–143, 2015. doi:10.1007/S00454-014-9644-Z.

6 Helena Bergold, Stefan Felsner, Meghana M. Reddy, Joachim Orthaber, and Manfred Scheucher. Plane Hamiltonian cycles in convex drawings. In *Proceedings of the 40th International Symposium on Computational Geometry (SoCG 2024)*, pages 18:1–18:16, 2024. doi:10.4230/LIPICS.SOCG.2024.18.

7 Helena Bergold, Stefan Felsner, Meghana M. Reddy, Joachim Orthaber, and Manfred Scheucher. Plane Hamiltonian cycles in convex drawings. *Discrete & Computational Geometry*, 2025. *Conference Version: [6].* doi:10.1007/s00454-025-00752-3.

8 Helena Bergold, Joachim Orthaber, Manfred Scheucher, and Felix Schröder. Holes in convex and simple drawings. In *Proceedings of the 32nd International Symposium on Graph Drawing and Network Visualization (GD 2024)*, pages 5:1–5:9. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2024. doi:10.4230/LIPICS.GD.2024.5.

9 Helena Bergold, Joachim Orthaber, Manfred Scheucher, and Felix Schröder. Holes in convex and simple drawings. *Journal of Graph Algorithms and Applications*, 29(3):23–38, 2025. *Conference Version: [8].* doi:10.7155/JGAA.V29I3.2999.

10 Grant Cairns, Emily Groves, and Yuri Nikolayevsky. Bad drawings of small complete graphs. *Australasian Journal of Combinatorics*, 75:322–342, 2019.

11 Vida Dujmovic and David R. Wood. On linear layouts of graphs. *Discret. Math. Theor. Comput. Sci.*, 6(2):339–358, 2004. doi:10.46298/DMTCS.317.

12 Stefan Felsner, Michael Hoffmann, Kristin Knorr, Jan Kynčl, and Irene Parada. On the maximum number of crossings in star-simple drawings of K_n with no empty lens. *Journal of Graph Algorithms and Applications*, 26(3):381–399, 2022. *Conference Version: [13].* doi:10.7155/jgaa.00600.

13 Stefan Felsner, Michael Hoffmann, Kristin Knorr, and Irene Parada. On the maximum number of crossings in star-simple drawings of K_n with no empty lens. In *Proceedings of 28th International Symposium on Graph Drawing and Network Visualization (GD 2020)*, pages 382–389, 2020. doi:10.1007/978-3-030-68766-3_30.

14 Jacob Fox, János Pach, and Andrew Suk. A structure theorem for pseudo-segments and its applications. In *Proceedings of the 40th International Symposium on Computational Geometry (SoCG 2024)*, pages 59:1–59:14. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2024. doi:10.4230/LIPICS.SOCG.2024.59.

15 Jacob Fox, János Pach, and Andrew Suk. A structure theorem for pseudosegments and its applications. *Journal of Combinatorial Theory, Series B*, 174:99–132, 2025. *Conference Version: [14].* doi:10.1016/J.JCTB.2025.04.007.

16 Radoslav Fulek. Estimating the number of disjoint edges in simple topological graphs via cylindrical drawings. *SIAM Journal on Discrete Mathematics*, 28(1):116–121, 2014. doi:10.1137/130925554.

17 Radoslav Fulek and Andres J Ruiz-Vargas. Topological graphs: empty triangles and disjoint matchings. In *Proceedings of the twenty-ninth annual symposium on Computational geometry*, pages 259–266, 2013. doi:10.1145/2462356.2462394.

18 Alfredo García, Alexander Pilz, and Javier Tejel. On plane subgraphs of complete topological drawings. *Ars Mathematica Contemporanea*, 20(1):69–87, 2021. doi:10.26493/1855-3974.2226.E93.

19 Lenwood S. Heath and Arnold L. Rosenberg. Laying out graphs using queues. *SIAM Journal on Computing*, 21(5):927–958, 1992. doi:10.1137/0221055.

20 Petr Hliněnỳ and Adam Straka. Stack and queue numbers of graphs revisited. *European Journal of Combinatorics*, 129:104094, 2025. doi:10.1016/j.ejc.2024.104094.

21 János Pach, József Solymosi, and Géza Tóth. Unavoidable configurations in complete topological graphs. *Discrete & Computational Geometry*, 30(2):311–320, 2003. *Conference Version: [24].* doi:10.1007/S00454-003-0012-9.

22 János Pach and Jenő Tőrőcsik. Some geometric applications of Dilworth's theorem. In *Proceedings of the ninth annual symposium on Computational geometry*, pages 264–269, 1993. doi:10.1007/BF02574361.

23 János Pach and Jenő Tőrőcsik. Some geometric applications of Dilworth's theorem. *Discrete & Computational Geometry*, 12:1–7, 1994. *Conference Version: [22].* doi:10.1007/BF02574361.

24 János Pach and Géza Tóth. Unavoidable configurations in complete topological graphs. In *Proceedings of the 8th International Symposium on Graph Drawing (GD 2000)*, pages 328–337, 2000. doi:10.1007/3-540-44541-2_31.

25 János Pach and Géza Tóth. Disjoint edges in topological graphs. In *Proceedings of the Indonesia-Japan Joint Conference on Combinatorial Geometry and Graph Theory (IJCCGGT 2003)*, pages 133–140, 2003. doi:10.1007/978-3-540-30540-8_15.

26 János Pach and Géza Tóth. Disjoint edges in topological graphs. *Journal of Combinatorics*, 1:335–344, 2010. *Conference Version: [25].* doi:10.4310/JOC.2010.v1.n3.a4.

27 Sergey Pupyrev. A collection of existing results on stack and queue numbers. https://spupyrev.github.io/linearlayouts.html, 2020. Accessed: 2/19/2026.

28 Nabil H Rafla. *The good drawings of the complete graph.* PhD thesis, McGill University, 1988.

29 Frank P. Ramsey. On a problem of formal logic. *Proceedings of the London Mathematical Society*, 2(1):264–286, 1930. doi:10.1112/plms/s2-30.1.264.

30 Andres J. Ruiz-Vargas. Many disjoint edges in topological graphs. *Computational Geometry*, 62:1–13, 2017. doi:10.1016/J.COMGEO.2016.11.003.

31 Micha Sharir, Adam Sheffer, and Emo Welzl. Counting plane graphs: perfect matchings, spanning cycles, and Kasteleyn's technique. In *Proceedings of the twenty-eighth annual symposium on Computational geometry*, pages 189–198, 2012. doi:10.1145/2261250.2261277.

32 Micha Sharir, Adam Sheffer, and Emo Welzl. Counting plane graphs: Perfect matchings, spanning cycles, and Kasteleyn's technique. *Journal of Combinatorial Theory, Series A*, 120(4):777–794, 2013. *Conference Version: [31].* doi:10.1016/J.JCTA.2013.01.002.

33 Andrew Suk. Disjoint edges in complete topological graphs. In *Proceedings of the twenty-eighth annual symposium on Computational geometry*, pages 383–386, 2012. doi:10.1145/2261250.2261308.

34 Andrew Suk. Disjoint edges in complete topological graphs. *Discrete & Computational Geometry*, 49(2):280–286, 2013. *Conference Version: [33].* doi:10.1007/S00454-012-9481-X.

35 Andrew Suk and Ji Zeng. Unavoidable patterns in complete simple topological graphs. In *Proceedings of the 30th International Symposium on Graph Drawing and Network Visualization (GD 2022)*, pages 3–15, 2022. doi:10.1007/978-3-031-22203-0_1.

36 Andrew Suk and Ji Zeng. Unavoidable patterns in complete simple topological graphs. *Discrete & Computational Geometry*, 73(1):79–91, 2025. *Conference Version: [35].* doi:10.1007/S00454-024-00658-6.

37 Maciej M. Syslo. Characterizations of outerplanar graphs. *Discrete Mathematics*, 26(1):47–53, 1979. doi:10.1016/0012-365X(79)90060-8.

38 Géza Tóth. Note on geometric graphs. *Journal of Combinatorial Theory, Series A*, 89(1):126–132, 2000. doi:10.1006/JCTA.1999.3001.

39 Alexandra Weinberger and Ji Zeng. What induces plane structures in complete graph drawings?, 2026. URL: https://arxiv.org/abs/2603.05208.

40 Ji Zeng. Note on disjoint faces in simple topological graphs. *Journal of Combinatorial Theory, Series B*, 171:28–35, 2025. doi:10.1016/J.JCTB.2024.11.002.

Antipodal Pairs and Crossing Numbers of Complete Graphs

Stefan Felsner

Technische Universität Berlin, Germany felsner@math.tu-berlin.de

Abstract

This paper makes contributions to the theory of crossing numbers of complete graphs. An *antipodal pair* in a drawing is a pair of vertices such that the stars of the two vertices induce a plane subgraph of the drawing. An *optimal* drawing of a graph is a drawing with a minimum number of crossings. We conjecture that optimal drawings of complete graphs with $n > 14$ have antipodal pairs. If true this implies Hill's conjecture:

$$cr(K_n) = H(n) = \frac{1}{4}\left\lfloor \frac{n}{2} \right\rfloor \left\lfloor \frac{n-1}{2} \right\rfloor \left\lfloor \frac{n-2}{2} \right\rfloor \left\lfloor \frac{n-3}{2} \right\rfloor.$$

Our second main contribution is the construction of a new family of Hill drawings, i.e., a family of drawings of K_n with crossing number equal to $H(n)$. A consequence of this new construction is that every spherical drawing of K_k is a subdrawing of a Hill drawing of K_n for all $n \geq 2k + 1$.

1 Introduction

We consider drawings of graphs in the plane with vertices being represented by points and edges by simple curves not containing vertices other than their endpoints. We also assume that intersections are proper crossings and that two edges share at most one point which may be a common vertex or a crossing point. Such drawings are known as *simple drawings*.

A drawing Γ of G is called *optimal* if $cr(\Gamma) = cr(G)$, i.e., if Γ minimizes the number of crossings among all drawings of G. It is known that optimal drawings are simple.

The crossing number of the complete graph has been studied since the late 1950s. British artist Anthony Hill conjectured that

$$cr(K_n) = H(n) = \begin{cases} \frac{1}{4}k(k-1)^2(k-2) & \text{if } n = 2k \qquad \text{(even)} \\ \frac{1}{4}k^2(k-1)^2 & \text{if } n = 2k+1 \qquad \text{(odd)} \end{cases}$$

Hill's conjecture has been verified with massive computer power for $n \leq 14$, see Aichholzer [7].

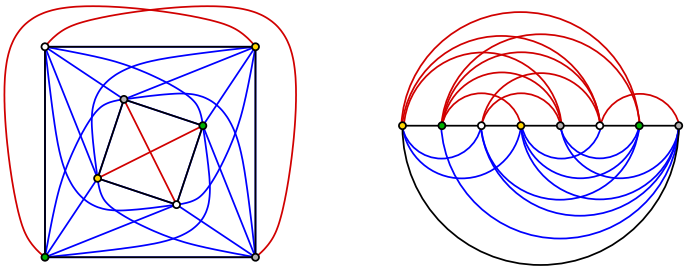


Figure 1 Antipodal pairs (pairs of vertices of the same color) in the cylindrical drawing and in the optimal 2-page drawing of K_8.

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

With $N(v)$ we denote the neighbor set of a vertex v in a graph, i.e., $N(v) = \{w : vw \in E_G\}$. The star $S(v)$ of a vertex v in a graph is the tree consisting of all edges incident to v, i.e., $S(v) = (\{v\} \cup N(v), \{vw : w \in N(v)\}$.

antipodal pair

▶ **Definition 1.1.** A pair x, x' of vertices in a drawing is an *antipodal pair* if the drawing induced by the union of the stars of x and x' is crossing free and $N(x) - \{x'\} = N(x') - \{x\} = V_G - \{x, x'\}$.

doublestar

In other words: The two stars of an antipodal pair induce a plane spanning *doublestar* of minimum degree 2. The drawings of K_8 in Figure 1 have antipodal pairs.

▶ **Conjecture 1.2.** *There is some integer N such that every optimal drawing of K_n with $n > N$ has an antipodal pair.*

Remark. Originally the conjecture was formulated without restriction on n. However, Matthew Sunohara found an optimal drawing of K_9, i.e., a drawing of K_9 with 36 crossings, which has no antipodal pair, see Figure 2(a). Joachim Orthaber later pointed out that an optimal drawing of K_9 shown in [7] has the same properties, this drawing is isomorphic to the drawing shown in Figure 2(b). Both optimal drawings of K_9 have the property that they have no optimal subdrawings of K_8. Oswin Aichholzer performed a search on his database of drawings of complete graphs and provided the following numbers: There are exactly 2 optimal drawings of K_9, 13.180 optimal drawings of K_{11}, and 3 optimal drawings of K_{12} without antipodal pair.

Despite these counterexamples to the original conjecture we think that it may be true that all counterexamples to the conjecture are sporadic and small.

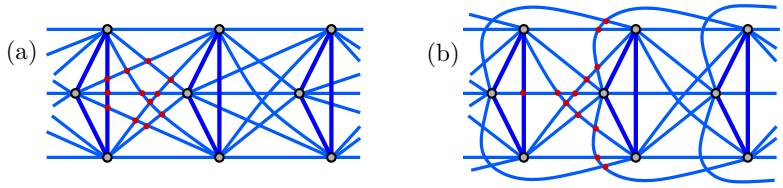


Figure 2 Two optimal drawings of K_9 without antipodal pairs. Both drawings have a threefold rotational symmetry which maps dark triangles to dark triangles. In the drawings we have emphasized 12 crossings in one of the three blocks of the symmetry.

In the next section (Section 2) we discuss how Conjecture 1.2 relates to Hill's conjecture and point to some additional consequences of the conjecture. In the final Section 4 we survey what is known about the conjecture of Hill.

2 Antipodal pairs in K_n

While it is nice to have a closed formula for the Hill number $H(n)$ it is more convenient in practice to work with formulas avoiding rounding.

$$H(n) = \begin{cases} \frac{1}{4}k(k-1)^2(k-2) = \binom{k}{2}\binom{k-1}{2} = 6\binom{k}{4} + 3\binom{k}{3} & \text{if } n = 2k \qquad \text{(even)} \\ \frac{1}{4}k^2(k-1)^2 = \binom{k}{2}^2 & \text{if } n = 2k+1 \qquad \text{(odd).} \end{cases}$$

For a uniform statement of the following proposition we define $\Delta(n) = H(n+2) - H(n)$ and note that for n even $\Delta(n) = \Delta(2k) = H(2k+2) - H(2k) = (2k-1)\binom{k}{2}$ and for n odd $\Delta(n) = \Delta(2k+1) = H(2k+3) - H(2k+1) = k^3$.

▶ Proposition 2.1. If Γ is a drawing of K_{n+2} with an antipodal pair x, x' and Γ_Θ is the induced drawing of K_n on $V - \{x, x'\}$, then

$$cr(\Gamma) \geq \Delta(n) + cr(\Gamma_\Theta).$$

Before proving the proposition we look at some of its consequences.

We first restate Conjecture 1.2 from the introduction with a venturously chosen constant.

▶ **Conjecture 2.2.** *Every optimal drawing of K_n with $n > 14$ has an antipodal pair.*

If we assume the conjecture and the induction hypothesis $cr(K_n) = H(n)$, then the proposition implies: $cr(K_{n+2}) \geq \Delta(n) + H(n) = H(n+2)$. Since $cr(K_n) = H(n)$ is known for all $n \leq 14$ we have:

▶ **Corollary 2.3.** *Conjecture 2.2 implies Hill's conjecture ($cr(K_n) = H(n)$ for all n).*

This directly implies that if Hill's conjecture is wrong, then a minimal counterexample must be an optimal drawing of a complete graph without antipodal pairs.

A drawing of K_n is *antipodally shellable* if the vertices of the graph can be listed $v_1, \ldots, v_n$ such that if Γ_i is the drawing induced by $v_1, \ldots, v_{n-2i}$ for $i = 0, \ldots, \lfloor \frac{n}{2} \rfloor$, then v_{n-2i-1}, v_{n-2i} is an antipodal pair in Γ_i.

antipodally shellable

▶ **Corollary 2.4.** *Crossing minimal antipodally shellable drawings of K_n have $H(n)$ crossings.*

The class of antipodally shellable drawings of complete graphs is a new and fairly large class of drawings where the Hill conjecture is true. In Section 4 we point to some other classes of drawings where the Hill conjecture has been shown.

2.1 Proof of the proposition

Let Γ be a drawing of K_{n+2} with an antipodal pair x, x' Let D be the doublestar of the pair x, x' and Γ_Θ be the drawing of K_n obtained by removing x, x', and the edges of D from Γ. We have

$$cr(\Gamma) = \#(\text{crossings on edges of } D) + cr(\Gamma_\Theta).$$

Hence, we focus on crossings on edges of D. The doublestar D has an edge directly connecting x and x'. To simplify the analysis we subdivide the edge xx' with an auxiliary vertex o. The doublestar D now consists of $n+1$ faces of degree 4. Let $x_0, x_1, \ldots, x_n$ be a labeling of the vertices of Γ different from x, x' but including o such that for $i = 1, \ldots, n$ the vertices x_{i-1}, x, x_i, x' belong to a 4-face of D. An edge $x_i x_j$ of Γ with $j > i$ has at least $c_{ij} = \min[j - i - 1\,,\, i + (n - j)]$ crossings with edges of D. For a fixed x_i we are interested in $\sum_j c_{ij}$ where j runs through all values such that $x_i x_j$ is an edge. To simplify the analysis we assume virtual edges between o and all the other vertices x_j. To evaluate $\sum_j c_{ij}$ we recall that with the subdivision vertex the doublestar has $n+1$ vertices of degree 2. For the following Figure 3 may help.

$$\sum_j c_{ij} = \begin{cases} 2\sum_{\ell=0}^{k-1} \ell = k(k-1) & \text{if } n = 2k \quad \text{(even)} \\ \sum_0^k \ell + \sum_0^{k-1} \ell = \binom{k+1}{2} + \binom{k}{2} = k^2 & \text{if } n = 2k+1 \quad \text{(odd)} \end{cases}$$

In the even case we have $n + 1 = 2k + 1$ vertices x_i. In the expression $(2k+1)k(k-1)$ we count the crossings of each edge $x_i x_j$ twice and we also count the contribution of the virtual edges of o twice. Hence, $\#(\text{crossings on edges of } D) \geq \frac{1}{2}(2k-1)k(k-1) = \Delta(2k) = \Delta(n)$.

In the odd case we have $n + 1 = 2k + 2$ vertices x_i. In the expression $(2k+2)k^2$ we count the crossings of each edge $x_i x_j$ twice and we also count the contribution of the virtual edges of o twice. Hence, $\#(\text{crossings on edges of } D) \geq k^3 = \Delta(2k+1) = \Delta(n)$. □

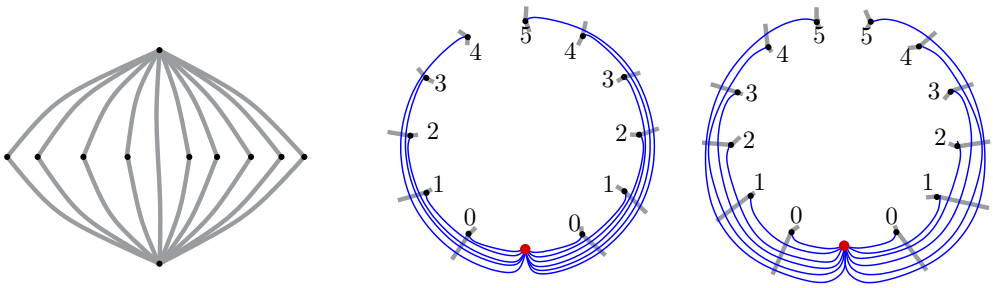


Figure 3 Left: A plane doublestar. Right: The $t = n + 1$ degree 2 vertices of a doublestar including o cyclically arranged. The left example shows $t = 12$ and the right examples shows $t = 13$. Labels at vertices reflect how many crossings with the doublestar an edge joining to the red bottom vertex must at least have.

3 A new family of Hill drawings

Hill drawing

A *Hill drawing* is a drawing of K_n with $H(n)$ crossings. There are two classical families of Hill drawings, *cylindrical drawings* (a.k.a. *can drawings*) and *optimal 2-page drawings*, see Figure 1. Both types of drawings were already known to Hill, see [12]. In the following section we survey the known Hill drawings. Now we introduce a new family of Hill drawings.

Let M_{2k} be the complete graph K_{2k} minus a perfect matching μ. Mohar [14] constructs spherical arc drawings of M_{2k} by taking for each edge of μ a pair of antipodal points on the sphere and using geodesic arcs for the edges. Let Γ be such a drawing. Assuming general position we see that the edges of Γ form $\binom{k}{2}$ great circles, with each great circle representing four edges involving two antipodal pairs, i.e., edges $ab, b\bar{a}, \bar{a}\bar{b}, \bar{b}a$. Every pair of great circles has two points of intersection which may be antipodal vertices or crossings. Hence, the spherical arc drawing Γ of M_{2k} has $M(2k) = \binom{k}{2}\binom{k-2}{2} = 6\binom{k}{4}$ crossings. Mohar conjectures that $cr(M_{2k}) = M(2k)$.

Let $z, \bar{z}$ be an antipodal pair of Γ. We construct a drawing $\Gamma_\star$ of K_{2k-1} by adding for each antipodal pair $p, \bar{p}$ the *great arc* (half of great circle) $p\bar{p}$ which contains $\bar{z}$ and by removing $\bar{z}$ together with all its edges. An example of the construction for $k = 6$ is shown in Figure **??**. Every crossing of Γ has a corresponding crossing in $\Gamma_\star$ in addition the $k - 1$ great arcs of $\Gamma_\star$ pairwise cross in the point of the sphere where $\bar{z}$ was located. Hence

$$cr(\Gamma_\star) = cr(\Gamma) + \binom{k-1}{2} = M(2k) + \binom{k-1}{2} = H(2k-1).$$

This shows that $\Gamma_\star$ is a Hill drawing.

Next we use such a drawing $\Gamma_\star$ of K_n with $n = 2k - 1$ to construct Hill drawings for $n = 2k$: We add a new vertex z^+ very close to the antipode z of the removed vertex $\bar{z}$ and use geodesics for the edges, this yields the drawing $\Gamma_\star^+$. In the original drawing Γ the edges incident to z and $\bar{z}$ both participated in $(k-1)\binom{k-2}{2}$ crossings: there are $k - 1$ choices for a point r to select a great circle $zr, r\bar{z}, \bar{z}\bar{r}, \bar{r}z$ and $\binom{k-2}{2}$ choices for a great circle $ab, b\bar{a}, \bar{a}\bar{b}, \bar{b}a$ with $\{r, z\} \cup \{a, b\} = \emptyset$. The two great circles cross at an edge incident to z and at the antipodal edge incident to $\bar{z}$. The edges of the new vertex z^+ have the same number of crossings with edges not incident to z. The great circle spanned by zz^+ splits each antipodal pair, hence, the great circle has $k - 1$ vertices on each side and the stars of z and z^+ have $2\binom{k-1}{2}$ crossings. Since $(k-1)\binom{k-2}{2} + 2\binom{k-1}{2} = (k-1)\binom{k-1}{2}$ this shows that the crossing

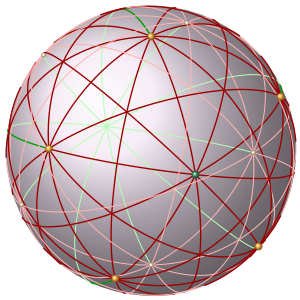


Figure 4 A drawing of K_{11} with $H(11) = 100$ crossings. The green edges are the great arcs.

number of the drawing $\Gamma_\star^+$ of K_{2k} is

$$cr(\Gamma_\star^+) = cr(\Gamma_\star)+(k-1)\binom{k-1}{2} = H(2k-1)+(k-1)\binom{k-1}{2} = \frac{1}{4}k(k-1)^2(k-2) = H(2k)$$

and $\Gamma_\star^+$ is a Hill drawing for $n = 2k$ even.

▶ **Theorem 3.1.** *The Hill drawings $\Gamma_\star$ and $\Gamma_\star^+$ of K_{2k-1} and K_{2k} respectively are distinct from the known Hill drawings.*

Proof. In both drawings the great arcs replacing the vertex $\bar{z}$ of M_{2k} are a set of $k-1$ pairwise crossing edges. Previously known Hill drawings of K_n have no set of pairwise crossing edges of size $\lfloor \frac{n-1}{2} \rfloor$. □

From the construction we immediately get the following surprising fact:

▶ **Fact 3.2.** Every spherical arc drawing of K_k is an induced subdrawing of a Hill drawing of K_{2k-1} and of K_{2k}.

4 Status of the conjecture of Hill

Hill drawings. The artist Anthony Hill was the first who found drawings of K_n with $H(n)$ crossings. His drawings can be described using a can of tomato soup. Place half the vertices evenly spaced on the top rim of the can, and the other half on the bottom rim. Vertices on the same rim are connected using only the top and bottom of the can. Edges between the two rims are routed as geodesics (shortest connecting curves) on the cylinder of the can. Some authors refer to these drawings as *cylindrical drawings*. In [5] the authors introduce new families of Hill drawings which are obtained by small modifications of cylindrical drawings.

cylindrical drawings

Cylindrical drawings can be realized as antipodal[1] spherical arc drawings, see Figure 5. Mohar [14] describes a construction of many pairwise non-isomorphic spherical antipodal Hill drawings. With the spherical representations of cylindrical drawings they share the property that they contain a perfect matching of pairwise disjoint great arcs. Streltsova and Wagner [17] prove that a point configuration on the sphere in general position (no three

[1] Note that we use the attribute antipodal in two ways. A meaning of antipodal has been established with Definition 1.1, however, on the sphere an antipodal drawing is a geodesic drawing where the points come in pairs $p, \bar{p}$ which are antipodal in the geometry of the sphere.

points on a common great circle) induces a spherical arc Hill drawing exactly if the point configuration is *coneighborly*, i.e., every open halfsphere contains at least $\lfloor (n-2)/2 \rfloor$ of the n points.

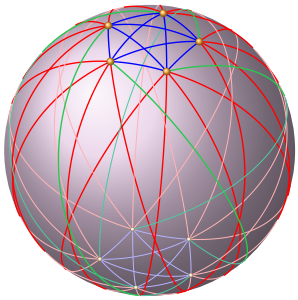


Figure 5 A spherical antipodal realization of the cylindrical drawing of K_{10}.

A second construction of drawings of K_n with $H(n)$ crossings is known as the 2-page drawing. A drawing appeared in the classical paper by Harary and Hill [12], the formal description and analysis is by Blažek and Koman [10]. The authors of [2] prove the Hill conjecture for the class of 2-page drawings and investigate properties of all crossing minimal 2-page drawings. They show that in the even case there is a unique 2-page Hill drawing while in the odd case there are exponentially many. Arroyo, Richter, and Sunohara [8] have a complete classification of optimal 2-page drawings.

Hill conjecture. The conjecture has been verified by Guy [11] for $n \leq 10$, with computer support the bound was pushed to $n \leq 12$ by Pan and Richter [16], and to $n \leq 14$ by Aichholzer [7]. The question remains open in general but there has been substantial progress in the past two decades. In two independent papers [6] and [13] it was observed that bounds on the number of $\leq k$-sets can be used in lower bounds for the rectilinear crossing number of complete graphs. This idea for a lower bound was adapted in [2] for more general drawings and the concept of $\leq\leq k$-edges was introduced to simplify the involved formulas. With this toolkit it was shown that 2-page drawings of K_n have at least $H(n)$ crossings. In subsequent years the technique was used to show that several other classes of drawings of K_n have at least $H(n)$ crossings. Here is a list of the most relevant classes with references:

- x-monotone drawings ([3] and [9]).
- cylindrical drawings, x-bounded drawings, and shellable drawings ([4]).
- bishellable drawings ([1]).

Streltsova and Wagner [17] show that the Hill conjecture holds for the class of spherical arc drawings of complete graphs. Their proof is based on a connection with $\leq k$-sets in three dimensions.

Acknowledgments. I thank Alexandra Wesolek, Yan Alves Radtke, and Joachim Orthaber for inspiring discussions on the topic, and Oswin Aichholzer for his exhaustive search for counterexamples with $n \leq 12$.

References

1 Bernardo M. Ábrego, Oswin Aichholzer, Silvia Fernández-Merchant, Dan McQuillan, Bojan Mohar, Petra Mutzel, Pedro Ramos, R. Bruce Richter, and Birgit Vogtenhuber. Bishellable drawings of K_n. *SIAM J. Discret. Math.*, 32(4):2482–2492, 2018.

2 Bernardo M. Ábrego, Oswin Aichholzer, Silvia Fernández-Merchant, Pedro Ramos, and Gelasio Salazar. The 2-page crossing number of K_n. *Discret. Comput. Geom.*, 49:747–777, 2013.

3 Bernardo M. Ábrego, Oswin Aichholzer, Silvia Fernández-Merchant, Pedro Ramos, and Gelasio Salazar. More on the crossing number of K_n: Monotone drawings. *Electron. Notes Discret. Math.*, 44:411–414, 2013.

4 Bernardo M. Ábrego, Oswin Aichholzer, Silvia Fernández-Merchant, Pedro Ramos, and Gelasio Salazar. Shellable drawings and the cylindrical crossing number of K_n. *Discret. Comput. Geom.*, 52:743–753, 2014.

5 Bernardo M. Ábrego, Oswin Aichholzer, Silvia Fernández-Merchant, Pedro Ramos, and Birgit Vogtenhuber. Non-shellable drawings of K_n with few crossings. In *Proc. 26th Canad. Conf. Comput. Geom.*, pages 7:1–7:6. Carleton Univ., 2014. `paper07.pdf`.

6 Bernardo M. Ábrego and Silvia Fernández-Merchant. A lower bound for the rectilinear crossing number. *Graphs Comb.*, 21:293–300, 2005.

7 Oswin Aichholzer. Another small but long step for crossing numbers: cr(13) = 225 and cr(14) = 315. In *Proc. CCCG 2021*, pages 72–77, 2021.

8 Alan Arroyo, Bruce Richter, and Matthew Sunohara. Number and convexity of crossing-optimal 2-page drawings of complete graphs, 2025. in preparation.

9 Martin Balko, Radoslav Fulek, and Jan Kynčl. Crossing numbers and combinatorial characterization of monotone drawings of K_n. *Discret. Comput. Geom.*, 53:107–143, 2015.

10 Jaroslav Blažek and Milan Koman. A minimal problem concerning complete plane graphs. In M. Fiedler, editor, *Theory Graphs Appl., Proc. Symp. Smolenice 1963*, pages 113–117. Academic Press, 1964.

11 Richard K. Guy. Crossing numbers of graphs. In *Graph Theory Appl., Proc. Conf. Western Michigan Univ. 1972*, volume 303 of *Lect. Notes Math.*, pages 111–124. Springer, 1972.

12 Frank Harary and Anthony Hill. On the number of crossings in a complete graph. *Proc. Edinburgh Math. Soc.*, 13:333–338, 1963.

13 Lázló Lovász, Katalin Vesztergombi, Uli Wagner, and Emo Welzl. Convex quadrilaterals and k-sets. In J. Pach, editor, *Towards a Theory of Geometric Graphs*, volume 342 of *Contemp. Math.*, pages 139–148. AMS, 2004.

14 Bojan Mohar. On a conjecture by Anthony Hill, 2020. `arXiv 2009.03418`.

15 Bojan Mohar. From art and circuit design to geometry and combinatorics. In *Proc. Europ. Congr. Math (8ECM)*, pages 663–676. EMS, 2023. `doi:10.4171/8ECM/37`.

16 Shengjun Pan and R. Bruce Richter. The crossing number of K_{11} is 100. *J. Graph Theory*, 56(2):128–134, 2007.

17 Elizaveta Streltsova and Uli Wagner. Sublevels in arrangements and the spherical arc crossing number of complete graphs, 2025. `arXiv 2504.07770`.

Fréchet Distance for paths in a d-dimensional grid graphs

Ivor van der Hoog[1], Eva Rotenberg[1], and Frederikke Uldahl[1]

1 **IT University of Copenhagen, Denmark**
ivva@itu.dk, erot@itu.dk, freu@itu.dk

Abstract

The Fréchet distance is a common similarity measure for curves (including polylines, trajectories, and walks in graphs). The Fréchet distance $F_G(P, Q)$ between two walks P and Q in a graph G can be computed in time quadratic in the input size. Van der Hoog, van der Horst, Rotenberg, and Wulf [ESA'25] showed that this quadratic running time is likely unavoidable: Unless the Orthogonal Vectors Hypothesis fails, there is no strongly subquadratic-time 1.25-approximation algorithm for the Fréchet distance between two paths in an unweighted planar graph. They obtain positive results by imposing additional structure on the underlying graph: if G is a regular planar tiling and P and Q are paths in G, then $F_G(P, Q)$ can be computed in time $O((|P| + |Q|)^{1.5})$. We extend this line of work to d-dimensional grid graphs by designing a $(1 + \varepsilon)$-approximation algorithm. In particular, we present an $(1 + \varepsilon)$-approximate algorithm of $F_G(P, Q)$ with runtime $\tilde{O}\left(|Q|\,(|P|)^{1-\frac{2}{d}}\,\varepsilon^{-3}\right)$.

Funding This work was supported by the Danmarks Frie Forskningsfond Case no. 10.46540/4251-00004B "Dynamic Graphs — Distributed and Geometry", and by the VILLUM Foundation grant (VIL37507) "Efficient Recomputations for Changeful Problems".

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

1 Introduction

The Fréchet distance is a widely recognised measure of similarity between two curves or trajectories. Conceptually, it can be considered the shortest leash length needed to connect a human walking along one path and a dog walking along another, with both moving forward without backtracking. The Fréchet distance is used in various domains, including signature verification [10], the study of coastline retrievals [9], the study of protein structures [7], and is widely used in the analysis of trajectories of moving objects [1, 2, 3, 8, 11].

Driemel, van der Hoog, and Rotenberg [5] introduced Fréchet distance in the context of graphs. The input consists of a weighted graph G, where trajectories are represented as walks along the graph. The distance between any two vertices is defined as the length of the shortest path connecting them. They show a conditional quadratic-time lower bound on computing the Fréchet distance between paths in a weighted graph, even when the graph is planar. Van der Hoog, van der Horst, Rotenberg and Wulf [12] extend this result by showing that the Fréchet distance of two disjoint paths in an unweighted planar graph can not be approximated better than a factor 1.25 in sub-quadratic time, unless the Orthogonal Vector Hypothesis fails. Despite this lower bound, the authors also identified a setting in which faster algorithms are possible. When G is a regular planar tiling and P and Q are paths in G of length n and m, the Fréchet distance can be computed in time $O((n+m)^{1.5})$.

Buchin, Ophelders and Speckmann [4] prove that, unless SETH fails, no strongly subquadratic algorithm can approximate the Fréchet distance between curves within a factor better than 3. This reduction also holds for walks in any d-dimensional grid graph.

Contribution. We present an $(1+\varepsilon)$-approximate algorithm to compute the Fréchet distance in a d-dimensional grid graph between path of complexity n and m in time $\tilde{O}\left(m(n)^{1-\frac{2}{d}}\,\varepsilon^{-3}\right)$. These bounds have a matching conditional lower bound, for constant $\varepsilon \leq 1$ and $d > 2$, which we consider out of scope for this submission.

2 Preliminaries

In graph theory, the infinite d-dimensional grid graph is the graph $G = (V, E)$, where $V = \mathbb{Z}^d$ is the set of all integer lattice points and $E = \{(p, q) \in V \mid ||p-q||_d = 1\}$. For any $p, q \in V$ let $\text{dist}_G(p, q)$ denote the length of the shortest path in G from p to q. If G is a grid graph, then $\text{dist}_G(p, q)$ is simply the Manhattan distance. Let $G = (V, E)$ be an infinite grid graph, and let $P = (p_1, p_2, \cdots, p_n)$ and $Q = (q_1, q_2, \cdots, q_m)$ be two paths in G (note that a path cannot visit a vertex twice). We assume $m \leq n$. A sequence $(a_i, b_i)_{i\in[1\cdots k]}$ of pairs of indices is a *monotone walk*, if for all $i \in [1 \cdots k-1]$ we have $a_{i+1} \in \{a_i, a_i+1\}$ and $b_{i+1} \in \{b_i, b_i+1\}$. Let $\mathbb{F}_{n,m}$ be the set of all monotone walks from $(1, 1)$ to (n, m). The (discrete) Fréchet distance $F_G(P, Q)$ between P and Q is defined as:

$$F_G(P, Q) = \min_{F\in\mathbb{F}_{n,m}} \max_{(i,j)\in F} \text{dist}_G(p_i, q_j).$$

We note for the reader that the continuous Fréchet distance behaves somewhat degenerately in graphs. For example, let G be the unit square, and let P and Q be its top and bottom edges, respectively. The continuous Fréchet distance equals 2, not 1, since the shortest path in G between the midpoints of P and Q has length 2. While our algorithms could likely be adapted to handle this phenomenon, we restrict ourselves to the discrete Fréchet distance.

Let $\text{dist}(p, q)$ denote the L_1 distance between two points p and q. The discrete Fréchet distance under the L_1 norm is defined as: $F_D(P, Q) = \min_{F\in\mathbb{F}_{n,m}} \max_{(i,j)\in F} \text{dist}(p_i, q_j)$.

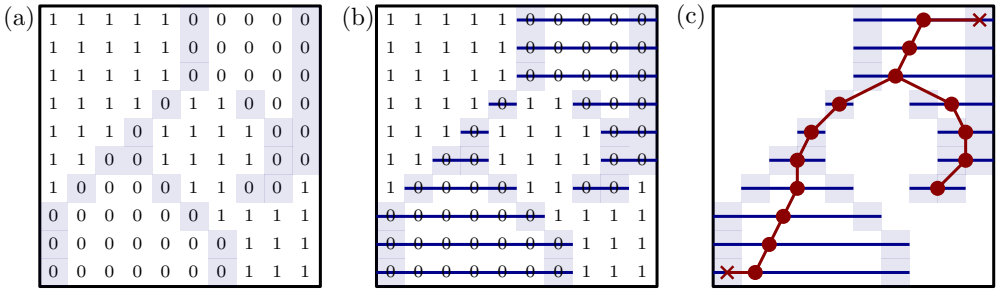


Figure 1 (a): A freespace matrix M_Δ, where the switching cells, S_Δ, are blue. (b): Maximal intervals of consecutive zeros in each row. (c): Building an adjacency graph over these intervals.

Free space matrix. Given two paths P, Q and some positive $\Delta \geq 0$, we wish to answer the decision variant if $F_G(P,Q) \leq \Delta$. A common tool for answering this is the Δ-free space matrix M_Δ: an n by m matrix where $M_\Delta[i,j] = 0$ if $\text{dist}_G(p_i, q_j) \leq \Delta$ and $M_\Delta[i,j] = 1$ otherwise. We follow geometric convention, where i denotes the column of M_Δ, corresponding to some $p_i \in P$. Column entries with value 0 correspond to $q_j \in Q$ with $\text{dist}_G(p_i, q_j) \leq \Delta$. Eiter and Mannila [6] define a directed graph over M_Δ where there is an edge from $M_\Delta[a,b]$ to $M_\Delta[c,d]$ if and only if $c \in \{a, a+1\}$, $d \in \{b, b+1\}$, and $M_\Delta[a,b] = M_\Delta[c,d] = 0$. They prove that $F_G(P,Q) \leq \Delta$ if and only if there exists a monotone path from $(1,1)$ to (n,m) in this graph. Calculating if such a path exists takes $O(nm)$ time.

Switching cells. There is an alternative approach for solving the decision variant. We call a cell $M_\Delta[i,j]$ a *switching cell* [12] if $M_\Delta[i,j] = 0$ and either $M_\Delta[i-1,j] = 1$, $M_\Delta[i+1,j] = 1$, or (i,j) lies on the boundary of M_Δ. Let S_Δ denote the set of all switching cells, and let $f(S_\Delta)$ be the time to compute S_Δ. One can decide whether $F_G(P,Q) \leq \Delta$ in $O(|S_\Delta| + f(S_\Delta))$ time as follows (see Figure 1). Firstly, we calculate the set of switching cells using $f(S_\Delta)$ time. For each row, we probe one cell between every pair of consecutive switching cells to identify all maximal horizontal intervals of zeroes. Two intervals are adjacent if they occur in consecutive rows and overlap horizontally. A depth-first search on the resulting adjacency graph determines whether a monotone path exists from $(1,1)$ to (n,m) in M_Δ.

Computing distances. In a d-dimensional grid graph, the distance $\text{dist}_G(p,q)$ between two vertices p and q is their Manhattan distance. The time required to compute this distance depends on the model of computation, the dimension d, and on how high the value $\text{dist}_G(p,q)$ is. Throughout this paper, we assume that this distance can be computed in constant time.

3 Approximating $F_G(P,Q)$ for paths in a d-dimensional grid graph

Given two paths P and Q in a d-dimensional grid graph G, we approximate $F_G(P,Q)$ by investigating the Fréchet distance for two simplified curves of P and Q, which we will show how to construct in the next section. In particular, we find a $j \in [O(\frac{1}{\varepsilon} \log n)]$ such that $\Delta_j = \frac{2(1+\varepsilon)^j}{\varepsilon}$ is a $(1+3\varepsilon)$-approximation of $F_G(P,Q)$. Invoking this algorithm with $\varepsilon' = \varepsilon/3$ yields a $(1+\varepsilon)$-approximation at no asymptotic overhead.

Simplified curves We view P as a curve, that is, as a continuous map $P : [1,n] \to \mathbb{R}^d$. We define a simplified curve P_j by some increasing sequence Σ of real values $i_t \in [1,n]$

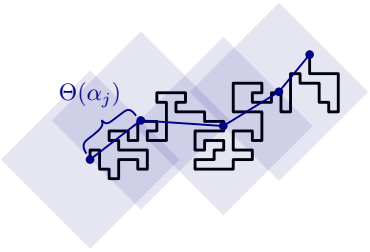


Figure 2 Construction of P_j (blue) given the path P (black)

such that P_j is a sequence of points $P(i_t)$ for $i_t \in \Sigma$. While constructing the curve P_j, we maintain a table T of size n that records, for each vertex $P(k)$, the edge $(P(i_t), P(i_{t+1}))$ with $k \in [i_t, i_{t+1})$ and say that this is the edge of P_j that this vertex is simplified to.

Define the curve P_j as follows: Set $\alpha_j = (1+\varepsilon)^j$. Initialise $i = 1$ and add the point $P(1)$ as the first point of P_j. Create the L_1 ball, $B(P(i), \alpha_j)$, with centre $P(i)$ and radius α_j. From $P(i)$ traverse P until the first point $P(i')$ that lies on the boundary of $B(P(i), \alpha_j)$. Add the point $P(i')$ to P_j. Denote in the table T that the vertices of P within the interval $[\lceil i \rceil \cdots \lfloor i' \rfloor]$ have been simplified to the edge from $(P(i)$ to $P(i'))$. Update i to i' and continue the simplification (see Figure 2). The edges of P_j have length $\Theta(\alpha_j)$, with the exception of the last edge, which may be shorter. As all edges in P have unit length, the curve P_j has $O(n/\alpha_j)$ vertices. By construction, $F_D(P, P_j) \leq \alpha_j$.

Combining methods Our algorithm relies on efficiently deciding if $F_D(P_j, Q_j) > 2\frac{\alpha_j}{\varepsilon} = \Delta_j$, where $\alpha_j = (1+\varepsilon)^j$. We will do so by using one of the two methods described in the preliminaries, which takes either $O(|P_j||Q_j|)$ or $O\left(f(S_{\Delta_j}) + |S_{\Delta_j}|\right)$ time. First, we will analyse the number of switching cells, $|S_{\Delta_j}|$, and how to efficiently compute the set S_{Δ_j}.

▶ **Lemma 3.1.** *Let P_j and Q_j be curves of size $\frac{n}{\alpha_j}$ and $\frac{m}{\alpha_j}$ in a d-dimensional grid graph. The free space matrix M_{Δ_j} for the curves P_j and Q_j, has $O\big((\Delta_j)^{d-1}\frac{m}{\alpha_j}\big)$ switching cells.*

Proof. Consider any vertex $q \in Q_j$. Firstly, we consider the number of switching cells in the row of M_{Δ_j} corresponding to the vertex q. The d-dimensional ball $B(q, \Delta_j)$, has a surface area of $O\left((\Delta_j)^{d-1}\right)$. As we move along the curve P_j, transitions between cells with value 1 and cells with value 0 of M_{Δ_j} occur when P_j crosses the boundary of $B(q, \Delta_j)$. As P is a path in a d-dimensional grid graph, P can intersect $B(q, \Delta_j)$ at most $O\left(\Delta_j^{d-1}\right)$ times. For every intersection of an edge $(P(i), P(i'))$ of P_j, there exists at least one unique intersection point $P(t)$ between P and $B(q, \Delta_j)$ with $t \in [i, i')$. Thus, P_j can intersect $B(q, \Delta_j)$ at most $O\left((\Delta_j)^{d-1}\right)$ times. The number of switching cells per $q \in Q_j$ is therefore $O\left((\Delta_j)^{d-1}\right)$ and there are thus $O\big((\Delta_j)^{d-1}\frac{m}{\alpha_j}\big)$ switching cells in total. ◀

▶ **Lemma 3.2.** *Given the curves P_j and Q_j and a membership query data structure of P and the table T mapping each vertex of P into its corresponding edge of P_j, we can calculate the set of switching cells S_{Δ_j} in M_{Δ_j}, for the curves P_j and Q_j, in $O\big(\Delta_j^{d-1}\frac{m}{\alpha_j}\log(n)\big)$ time.*

Proof. Consider any vertex $q \in Q_j$. Firstly, we consider finding the switching cells in the row of M_{Δ_j} corresponding to q. Let $(P(i), P(i'))$ be an edge in P_j. The cell in M_{Δ_j} corresponding to the two points $P(i)$ and q is a switching cell if the point $P(i) \in B(q, \Delta_j)$ and $P(i') \notin B(q, \Delta_j)$ and vice versa. If this is the case, there must exist a point $P(t)$ of P with $t \in [i, i')$ that lies on the boundary of $B(q, \Delta_j)$. We can detect these points as follows:

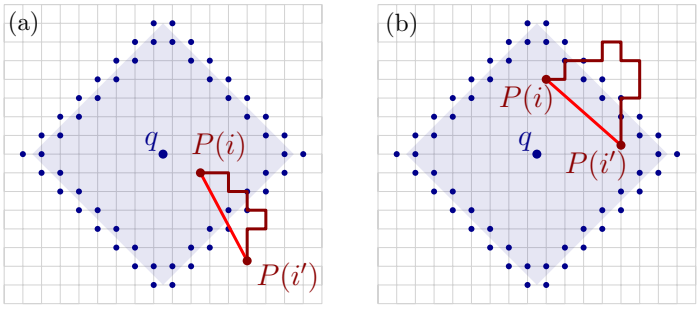


Figure 3 Finding a switching cells in a row of M_{Δ_j} corresponding to the point $q \in Q_j$. Shows the ball $B(q, \Delta_j)$ (blue area), the vertices V(blue dots), an edge $(P(i), P(i')) \in P_j$ (red) and a path segment of P (dark red). (a): Detection of a switching cell. Instance where $P(i) \in B(q, \Delta_j)$ and $P(i') \notin B(q, \Delta_j)$. The edge $(P(i), P(i'))$ will be considered, as there exist a vertex in P, in-between $P(i)$ and $P(i')$, that lies in V. (b): No detection of a switching cell, as both $P(i), P(i') \in B(q, \Delta_j)$.

For each $q \in Q_j$ we identify the set of vertices V in the d-dimensional grid graph that either intersect the ball $B(q, \Delta_j)$ or are endpoints of edges that intersect the ball $B(q, \Delta_j)$ (see figure 3). For each $v \in V$, we query the membership query data structure of P if v is in P. If v is in P, we use the table T to look up the edge $(P(i), P(i')) \in P_j$ that the vertex v has been simplified to. If $P(i) \in B(q, \Delta_j)$ and $P(i') \notin B(q, \Delta_j)$, we mark the cell corresponding to the points $P(i)$ and q as a switching cell.

The ball $B(q, \Delta_j)$ has surface area $O((\Delta_j)^{d-1})$, so $|V| \in O\left((\Delta_j)^{d-1}\right)$. For each $v \in V$, checking membership takes $O(\log n)$ time, while identifying the edge $(P(i), P(i'))$, computing $\text{dist}(P(i), q)$ and $\text{dist}(P(i'), q)$ take $O(1)$ time. Thus, identifying the switching cells in a row of M_{Δ_j} takes $O\left((\Delta_j)^{d-1} \log n\right)$ time, and $O\left((\Delta_j)^{d-1} \frac{m}{\alpha_j} \log n\right)$ time overall. ◀

▶ **Lemma 3.3.** *Deciding if* $F_D(P_j, Q_j) > 2\frac{\alpha_j}{\varepsilon} = \Delta_j$ *takes* $O\left(m\,(n)^{1-\frac{2}{d}}\,\varepsilon^{-2}\left(\log(n)\right)^{\frac{2}{d}}\right)$ *time.*

Proof. We decide whether $F_D(P_j, Q_j) > 2\frac{\alpha_j}{\varepsilon} = \Delta_j$ by using either the method with runtime $O\left(|P_j|\,|Q_j|\right) \in O\left(\frac{nm}{\alpha_j^2}\right)$ or the method with runtime $O\left(f(S_{\Delta_j}) + |S_{\Delta_j}|\right)$. Lemma 3.2 and Lemma 3.1 imply that $O\left(f(S_{\Delta_j}) + |S_{\Delta_j}|\right) \in O\left(\left(2\frac{\alpha_j}{\varepsilon}\right)^{d-1} \frac{m}{\alpha_j} \log(n)\right)$. Thus, for large values of α_j, the first method is faster until a switching point, which we now compute:

$$\left(2\frac{\alpha_j}{\varepsilon}\right)^{d-1} \frac{m}{\alpha_j} \log(n) = \frac{nm}{\alpha_j^2}$$

$$\alpha_j = \left(\left(\frac{\varepsilon}{2}\right)^{d-1} \frac{n}{\log(n)}\right)^{\frac{1}{d}} \qquad \text{(solve for } \alpha_j\text{)}$$

$$\alpha_j^2 = \left(\frac{\varepsilon}{2}\right)^{\frac{2(d-1)}{d}} \left(\frac{n}{\log(n)}\right)^{\frac{2}{d}} \qquad \text{(square both sides)}$$

$$\frac{nm}{\alpha_j^2} = m\,(n)^{1-\frac{2}{d}} \left(\frac{\varepsilon}{2}\right)^{-\frac{2(d-1)}{d}} \left(\log(n)\right)^{\frac{2}{d}} \qquad \text{(invert and multiply by } nm\text{)}$$

$$|P_j|\,|Q_j| = \frac{nm}{\alpha_j^2} < m\,(n)^{1-\frac{2}{d}}\,\varepsilon^{-2}\left(\log(n)\right)^{\frac{2}{d}} \qquad \text{(since } d > 0 \text{ and } 0 < \varepsilon < 1\text{)}.$$

Thus, we can decide if $F_D(P_j, Q_j) > 2\frac{\alpha_j}{\varepsilon} = \Delta_j$ in time $O\left(m\,(n)^{1-\frac{2}{d}}\,\varepsilon^{-2}\left(\log(n)\right)^{\frac{2}{d}}\right)$ ◀

Approximation factor We wish to find an appropriate j such that Δ_j will be a good approximation of $F_G(P,Q)$. In the following, we will look at properties of such a j. We can always translate the two curves P_j and Q_j such that they lie in a bounding box of size $n+m$. Indeed, if these curves do not lie in such a bounding box, then, because they have at most length n and m, they must be separated by a line. We can therefore translate them until they lie in such a box, and the Fréchet distance is decreased by the translation that we made. Hence, we only consider $j \in [1 \cdots \lceil \log_{(1+\varepsilon)}(n+m) \rceil] = [1 \cdots O(\frac{1}{\varepsilon}\log n)]$ as $\Delta_j = 2\frac{(1+\epsilon)^j}{\varepsilon}$ for any larger j would be greater than $n+m$ (and we assumed $m \leq n$).

▶ **Lemma 3.4.** *Let j be the largest integer in the interval $[1 \cdots O(\frac{1}{\varepsilon}\log n)]$ such that $F_D(P_j,Q_j) > 2\frac{\alpha_j}{\varepsilon} = \Delta_j$. Then Δ_j is a $(1+3\varepsilon)$ approximation of $F_G(P,Q)$ for $\varepsilon < \frac{1}{2}$.*

Proof. From the triangle inequality for Fréchet distance, we have: $F_G(P,Q) \geq F_D(P_j,Q_j) - F_D(P,P_j) - F_D(Q,Q_j) > 2\frac{\alpha_j}{\varepsilon} - 2\alpha_j = 2\frac{\alpha_j}{\varepsilon}(1-\varepsilon) = \Delta_j(1-\varepsilon)$. Thus, $\Delta_j < \frac{F_G(P,Q)}{(1-\varepsilon)}$.

As j is the largest integer such that $F_D(P_j,Q_j) > \Delta_j$ we have $F_D\left(P_{j+1},Q_{j+1}\right) \leq \Delta_{j+1} = (1+\varepsilon)\Delta_j$. The triangle inequality then gives $F_G(P,Q) \leq F_D(P,P_{\alpha_{j+1}}) + F_D(P_{\alpha_{j+1}},Q_{\alpha_{j+1}}) + F_D(Q_{\alpha_{j+1}},Q) \leq \Delta_{j+1} + 2\alpha_{j+1} = 2\frac{\alpha_{j+1}}{\varepsilon} + 2\alpha_{j+1} = 2\frac{\alpha_{j+1}}{\varepsilon}(1+\varepsilon) = 2\frac{\alpha_j}{\varepsilon}(1+\varepsilon)^2 = \Delta_j(1+\varepsilon)^2$. This gives:

$$\frac{F_G(P,Q)}{(1+3\varepsilon)} < \frac{F_G(P,Q)}{(1+\varepsilon)^2} \leq \Delta_j < \frac{F_G(P,Q)}{(1-\varepsilon)} < F_G(P,Q)(1+3\varepsilon) \quad \blacktriangleleft$$

▶ **Lemma 3.5.** *Assume for all integers $j \in [1 \cdots O(\frac{1}{\varepsilon}\log n)]$, that $F_D(P_j,Q_j) \leq \Delta_j$. Then $F_G(P,Q)$ can be calculated in $O(m\log(n))$ time.*

Proof. Suppose for all integers $j \in [O(\frac{1}{\varepsilon}\log n)]$, that $F_D(P_j,Q_j) \leq \Delta_j$. Then $F_D(P_1,Q_1) \leq 2\frac{(1+\varepsilon)^1}{\varepsilon} \Rightarrow F_G(P,Q) \leq 2\frac{(1+\varepsilon)^1}{\varepsilon} + 2(1+\varepsilon)^1$ which simply is a constant. In this case, we can use the method with runtime $O(|S_\Delta| + f(|S_\Delta|)$ to binary search over $O(1)$ possible values of $F_G(P,Q)$ to find the exact value for $F_G(P,Q)$. For a constant Δ, it takes $O(\Delta^{d-1}m\log(n)) \in O(m\log(n))$ time to calculate the set of switching cells S_Δ and $|S_\Delta| \in O(\Delta^{d-1}m) \in O(m)$. Thus, it takes $O(m\log(n))$ time to calculate $F_G(P,Q)$. ◀

Algorithm To approximate $F_G(P,Q)$ we will firstly translate the two paths P and Q so that they lie in a bounding box of size $n+m$. We then preprocess the translated P in a membership query data structure. Set $0 < \varepsilon < \frac{1}{2}$ and $\alpha_j = (1+\varepsilon)^j$. For $j \in [1 \cdots O(\frac{1}{\varepsilon}\log n)]$, iterating from largest to smallest, we will construct P_j and Q_j, and the table T mapping the simplification of P into P_j. We will return $\Delta_j = 2\frac{\alpha_j}{\varepsilon}$ as an estimate for $F_G(P,Q)$ for the largest value of j where $F_D(P_j,Q_j) > 2\frac{\alpha_j}{\varepsilon} = \Delta_j$. If $F_D(P_j,Q_j) \leq \Delta_j$ for all j, we will compute the exact value of $F_G(P,Q)$ by using the switching cells method.

Runtime analysis of the algorithm Translating the two paths P and Q and preprocessing the translated P in a membership query data structure can be done in $O(n\log(n))$ time. Lemma 3.3 tells deciding whether $F_D(P_j,Q_j) > \Delta_j$ takes $O\left(m\,(n)^{1-\frac{2}{d}}\,\varepsilon^{-2}\left(\log(n)\right)^{\frac{2}{d}}\right)$ time. Testing this for all $O(\frac{1}{\varepsilon}\log(n))$ values of j thus takes $O\left(m\,(n)^{1-\frac{2}{d}}\,\varepsilon^{-3}\left(\log(n)\right)^{1+\frac{2}{d}}\right)$ time. If we find a j such that $F_D(P_j,Q_j) > \Delta_j$, Lemma 3.4 implies that Δ_j is a $(1+3\varepsilon)$ approximation of $F_G(P,Q)$ If we don't find a j such that $F_D(P_j,Q_j) > \Delta_j$, Lemma 3.5 tells $F_D(P_j,Q_j)$ can be computed in $O(m\log(n))$ time. Therefore, we can compute a $(1+3\varepsilon)$ approximation of $F_G(P,Q)$ in $O\left(m\,(n)^{1-\frac{2}{d}}\,\varepsilon^{-3}\left(\log(n)\right)^{1+\frac{2}{d}}\right) \in \tilde{O}\left(m\,(n)^{1-\frac{2}{d}}\,\varepsilon^{-3}\right)$ time. Invoking this algorithm with $\varepsilon' = \frac{\varepsilon}{3}$ yields our main result:

▶ **Theorem 3.6.** *Let P and Q be paths in a d-dimensional grid graph, with n and m vertices and $m \leq n$. For all $\varepsilon \in (0, \frac{1}{2})$ we can $(1+\varepsilon)$-approximate $F_G(P,Q)$ in time* $\tilde{O}\left(m\,(n)^{1-\frac{2}{d}}\,\varepsilon^{-3}\right)$.

References

1 Alessandro Bombelli, Lluis Soler, Eric Trumbauer, and Kenneth D Mease. Strategic air traffic planning with Fréchet distance aggregation and rerouting. *Journal of Guidance, Control, and Dynamics*, 40(5):1117–1129, 2017. doi:10.2514/1.G002308.

2 Sotiris Brakatsoulas, Dieter Pfoser, Randall Salas, and Carola Wenk. On map-matching vehicle tracking data. In *International Conference on Very Large Data Bases (VLDB)*, pages 853–864, 2005. doi:10.5555/1083592.1083691.

3 Kevin Buchin, Maike Buchin, David Duran, Brittany Terese Fasy, Roel Jacobs, Vera Sacristan, Rodrigo I Silveira, Frank Staals, and Carola Wenk. Clustering trajectories for map construction. In *ACM International Conference on Advances in Geographic Information Systems (SIGSPATIAL)*, pages 1–10, 2017. doi:10.1145/3139958.3139964.

4 Kevin Buchin, Tim Ophelders, and Bettina Speckmann. SETH says: Weak Fréchet distance is faster, but only if it is continuous and in one dimension. In *ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 2887–2901, 2019. doi:10.5555/3310435.3310614.

5 Anne Driemel, Ivor van der Hoog, and Eva Rotenberg. On the Discrete Fréchet Distance in a Graph. In *Symposium on Computational Geometry (SoCG 2022)*, volume 224, pages 36:1–36:18, 2022. doi:10.4230/LIPIcs.SoCG.2022.36.

6 Thomas Eiter, Heikki Mannila, et al. Computing discrete fréchet distance. 1994.

7 Minghui Jiang, Ying Xu, and Binhai Zhu. Protein structure–structure alignment with discrete Fréchet distance. *Journal of bioinformatics and computational biology*, 6(01):51–64, 2008. doi:10.1142/s0219720008003278.

8 Maximilian Konzack, Thomas McKetterick, Tim Ophelders, Maike Buchin, Luca Giuggioli, Jed Long, Trisalyn Nelson, Michel A Westenberg, and Kevin Buchin. Visual analytics of delays and interaction in movement data. *International Journal of Geographical Information Science*, 31(2):320–345, 2017. doi:10.5555/3048386.3048392.

9 Ariane Mascret, Thomas Devogele, Iwan Le Berre, and Alain Hénaff. Coastline matching process based on the discrete Fréchet distance. In *Progress in Spatial Data Handling*, pages 383–400. Springer, 2006. doi:10.1007/3-540-35589-8_25.

10 M.E. Munich and P. Perona. Continuous dynamic time warping for translation-invariant curve alignment with applications to signature verification. In *IEEE International Conference on Computer Vision (ICCV)*, volume 1, pages 108–115 vol.1, 1999. doi:10.1109/ICCV.1999.791205.

11 Han Su, Shuncheng Liu, Bolong Zheng, Xiaofang Zhou, and Kai Zheng. A survey of trajectory distance measures and performance evaluation. *Proceedings of the VLDB Endowment*, 29(1):3–32, 2020. doi:10.1007/s00778-019-00574-9.

12 Ivor van der Hoog, Thijs van der Horst, Eva Rotenberg, and Lasse Wulf. Fréchet Distance in Unweighted Planar Graphs. In *European Symposium on Algorithms (ESA 2025)*, volume 351, pages 24:1–24:16, 2025. doi:10.4230/LIPIcs.ESA.2025.24.

The Euclidean Minimum Spanning Tree Extension Problem (and its Approximation)

Emilio Di Giacomo[1], Giuseppe Liotta[1], Daniel Perz[1], and Morteza Saghafian[2]

1 **University of Perugia, Italy**
emilio.digiacomo|giuseppe.liotta|daniel.perz@unipg.it
2 **TU Wien, Vienna, Austria**
msaghafi@ac.tuwien.ac.at

Abstract

Let T be a tree and let Γ' be a drawing of a subtree of T such that Γ' is a Euclidean minimum spanning tree (EMST) of its vertex set. We study the problem of extending Γ' to a drawing of T that is also an EMST of the full vertex set. We prove the problem to be NP-hard even for trees having maximum vertex degree three. We also show that for any $\varepsilon > 0$ the drawing Γ' can be extended to a $(1+\varepsilon)$-EMST drawing of T, i.e., to a planar straight-line drawing in which the distance between any two vertices is at least $\frac{1}{1+\varepsilon}$ times the length of the longest edge on the path connecting them. The computed drawings have polynomial edge-length ratio for trees with bounded vertex degree and can be constructed in linear time assuming the real RAM model of computation.

1 Introduction

An *EMST-drawing* of a tree T is a straight-line drawing Γ of T such that Γ is a Euclidean minimum spanning tree (EMST) of the set of points representing the vertices of T. Let T be a tree, let $T' \subset T$ be a subtree of T and let Γ' be an EMST-drawing of T'. We say that Γ' is a *partial EMST-drawing* of T. An *EMST-extension* of T with respect to Γ' is an EMST-drawing Γ of T such that Γ' is a subdrawing of Γ.

A seminal paper by Monma and Suri [6] proves that every tree with vertex degree at most five admits an EMST-drawing. Monma and Suri [6] also show that no trees with vertex degree larger than six admit such a drawing. Eades and Whitesides [2] prove that, in fact, recognizing whether a tree with maximum degree-six vertices has an EMST-drawing is NP-hard. The proof in [6] for the EMST-drawings of trees with maximum degree five is constructive and it gives rise to realizations whose edge-length ratio (i.e., the ratio of the longest to the shortest edge in the drawing) is exponential in the size of the tree. The works of Kaufmann [5] and of Frati and Kaufmann [3] show how to compute EMST drawings with polynomial edge-length ratio for trees with vertex degree at most four.

In this paper we study the following generalization of the recognition problem considered by Eades and Whitesides [2], which we call the EMST-Extension problem.

EMST-Extension

Instance: A tree T and a partial EMST-drawing Γ' of a subtree of T.
Question: Does there exist an EMST-extension of T with respect to Γ'?

While the work of Eades and Whitesides [2] already implies that the EMST-Extension problem is NP-hard for trees having vertices of degree six, our first contribution is to extend the NP-hardness to trees with smaller vertex degree.

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

▶ **Theorem 1.1.** *The EMST-Extension problem is NP-hard even for trees having maximum vertex degree three.*

Motivated by Theorem 1.1, we investigate whether Γ' can be extended to some straight-line drawing Γ of the full tree T such that Γ is a good geometric approximation of the Euclidean minimum spanning tree of its vertex set. Our second contribution is to show that for any tree T, for any EMST-drawing Γ' of a subtree of T, and for any $\varepsilon > 0$, it is always possible to extend Γ' to a $(1+\varepsilon)$-EMST drawing of T. A $(1+\varepsilon)$-EMST drawing is a planar straight-line drawing of a tree such that for any pair of vertices their Euclidean distance is at least $\frac{1}{1+\varepsilon}$ times the length of the longest edge along the path connecting them. The proof of this second result is constructive, giving rise to $(1+\varepsilon)$-EMST drawings whose edge-length ratio is upper bounded by a polynomial in $O(n^{2+f(\Delta,\varepsilon)})$, where n denotes the number of vertices of the tree and Δ is its maximum vertex degree. The function $f(\Delta,\varepsilon)$ increases by either increasing Δ or decreasing ε, giving rise to a polynomial approximation scheme for the edge-length ratio of trees with degree Δ. More precisely, we prove the following.

▶ **Theorem 1.2.** *Let T be a tree whose maximum vertex degree is Δ, let $T' \subset T$ be a subtree of T and let Γ' be an EMST-drawing of T'. For any $\varepsilon > 0$, there exists a $(1+\varepsilon)$-EMST drawing Γ of T such that $\Gamma' \subset \Gamma$ and the edge-length ratio of Γ is $O(n^{2+\log_2 3c^{\Delta}})$, where c is a constant such that $c > \max\{\frac{\varepsilon+1}{\varepsilon}, \frac{1}{\sin\frac{\pi}{\Delta-1}}\}$. Also, this drawing can be computed in $O(n)$ time assuming the real RAM model of computation.*

Due to space constraints the full proofs of the theorems can be found in the full version.

2 NP-Hardness for maxdegree 3 trees

To prove Theorem 1.1 we reduce from Planar-Monotone-3-Sat [1]. In our reduction, we construct a partial EMST-drawing Γ' whose vertices are at grid points of a triangular grid and whose edges have length one. We call such a (partial) EMST-drawing a *unit-length (partial) EMST-drawing*. Given a unit-length partial EMST-drawing Γ', a point p of the triangular grid is said to be *used* if p represents a vertex in Γ', *unused* otherwise. In our reduction the part of the tree T that is not drawn consists of a set of paths each sharing an end-vertex u with Γ'. We call each such path π a *dangling path* and we say that a vertex u is the *attachment of* π. We say that π is the *dangling path at* u if u is a vertex in Γ', π is not a path in Γ' but it must be drawn by placing one of its end-vertices at u.

Assume we are given a unit-length partial EMST-drawing Γ' and a used point p of the triangular grid. We want to add a point p' with edge pp' to the drawing such that the extended drawing is still an EMST-drawing. Consider the hexagon S whose corners are the six points adjacent to p on the grid and assume that at least one of them is unused. If no two unused points of these six points are adjacent, then it is easy to see that p' is at one of the unused points. Namely, if p' was strictly inside S, then its distance from one of the used points of S would be smaller than one, and therefore shorter than any other edge in the drawing, implying that the drawing is not an EMST-drawing; if otherwise, p' was outside S, then its distance from one of the used points of S would be smaller than the distance from p, and therefore shorter than the edge pp', implying again that the drawing is not an EMST-drawing. Hence, p' must be at one of the unused point of S and the drawing extended with edge pp' is still a unit-length EMST-drawing.

In the following we describe the different gadgets used in the construction. Edges in the given drawing are black, the unused points inside the respective gadget are depicted with crosses and the attachments are depicted with large disks.

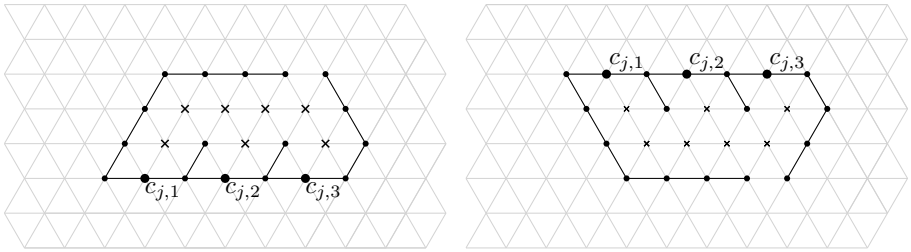


Figure 1 A positive 3-clause gadget (left) and a negative 3-clause gadget (right).

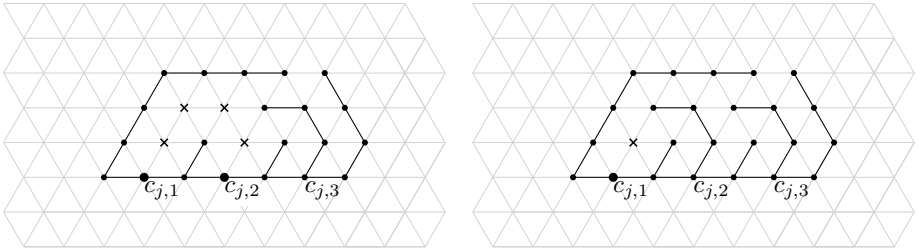


Figure 2 A positive 2-clause gadget (left) and a positive 1-clause gadget (right).

Clause gadget. An *h-clause gadget* is a subtree of Γ' used to model a clause of φ with h literals for $h = 1, 2, 3$. Figure 1 (left) shows a positive 3-clause and Figure 1 (right) shows a negative 3-clause. The three distinguished points $c_{j,1}, c_{j,2}, c_{j,3}$ are the three attachments of $\mathcal{C}_j$. Observe that it is not possible to draw all three dangling paths of length three inside.

Variable gadget. A *variable gadget* $\mathcal{V}_i$ of size k_i is a subtree of Γ' used to model a variable of φ whose vertex degree in G_φ is k_i, see Figure 3. The middle path is a horizontal path of length $3k_i + 1$ (i.e., $3k_i + 2$ vertices). $\mathcal{V}_i$ has $2k_i + 1$ attachments, one in the middle path of $\mathcal{V}_i$ (depicted as u_i in Figure 3), and k_i in the upper and lower path. Further the dangling path incident to u_i has length $3k_i$, while all other dangling paths incident to an attachment of $\mathcal{V}_i$ have length 3. Observe, that if the dangling path at u_i is drawn above (below resp.) the middle path, then all dangling paths incident to upper (lower resp.) attachments are forced to be above (below resp.) the upper path.

Sector gadget and Connector gadget. A *connector gadget* $\mathcal{K} \subset \Gamma'$ is used to model the connection between variable and clause gadgets, see Figure 4. They consist of a series of *sector gadgets* $\mathcal{S}_h \subset \Gamma'$, $1 \leq h \leq k$, where each $\mathcal{S}_h$ consists of two parallel upward paths of

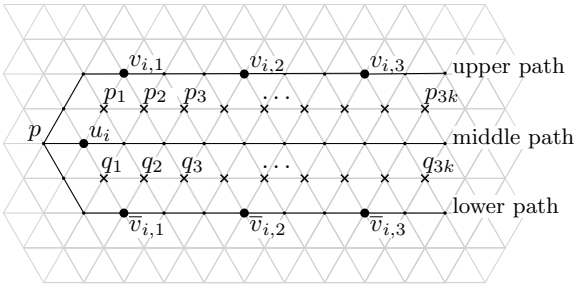


Figure 3 A variable gadget of size 3.

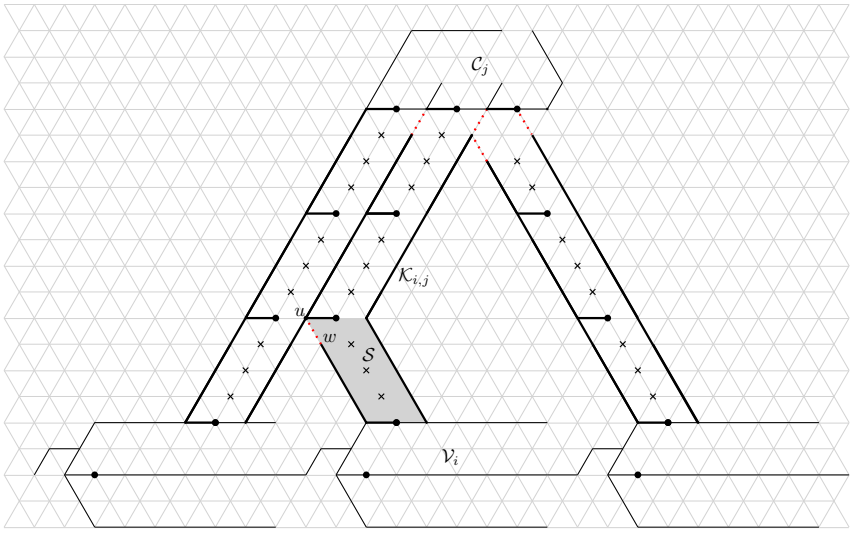


Figure 4 Three connectors incident to a common clause gadget. The fat black edges are edges of the connector gadgets in Γ' while the red dotted edges (for example uw) are not in Γ'.

length 4 and two horizontal edges. Note that the exact shape of the left and right boundary is not fixed. The two attachments of $\mathcal{S}_h$ are incident to two dangling paths of length 3. Observe that only one of these dangling paths can be inside $\mathcal{S}$ and that the resulting drawing uses the unused points of $\mathcal{S}_h$. This means that at least one dangling path of $\mathcal{K}$ has to be drawn inside an incident variable or a clause gadget.

The idea to see that extending this constuction of Γ' to Γ is at least as hard as solving an instance of Planar-Monotone-3-Sat, is the following. Starting with a truth assignment of the boolean formula we draw the dangling path at u_i in $\mathcal{V}_i$ below the middle path if and only if the corresponding variable is assigned with true. We draw the remaining dangling paths as close to the middle as possible.

Starting with a completed drawing Γ_φ we focus on a clause gadget $\mathcal{C}_j$. By construction one incident dangling path π is not drawn inside. This dangling path is also incident to a connector gadget $\mathcal{K}_{i,j}$ that connects a variable gadget $\mathcal{V}_i$ with $\mathcal{C}_j$. Hence, we assign the corresponding variable as true if $\mathcal{C}_j$ corresponds to a positive literal, and false otherwise.

3 $(1+\varepsilon)$-EMST-drawings

Given two vertices u and v and an edge e of a straight-line drawing Γ, we denote by $d(u,v)$ the Euclidean distance between u and v in Γ and by $|e|$ the length of the segment representing e in Γ. A straight-line drawing Γ of a tree $T=(V,E)$ is an EMST-drawing if and only if the following *EMST-property* holds: $\forall u,v \in V, d(u,v) \geq |e_\Gamma(u,v)|$, where $e_\Gamma(u,v)$ is the longest edge of the path from u to v in Γ. Let $\varepsilon > 0$ be a given constant and let T be a tree. A $(1+\varepsilon)$-EMST drawing of T is a planar straight-line drawing Γ of T such that for any two vertices u and v the following $(1+\varepsilon)$-*EMST property* is satisfied: $\forall u,v \in V, d(u,v) \geq \frac{1}{1+\varepsilon}|e_\Gamma(u,v)|$, where $|e_\Gamma(u,v)|$ is the length of the longest edge in the path from u to v in Γ [4]. The *edge-length ratio* $\rho(\Gamma)$ of Γ is the ratio between the length of the longest edge in Γ and the length of the shortest edge in Γ.

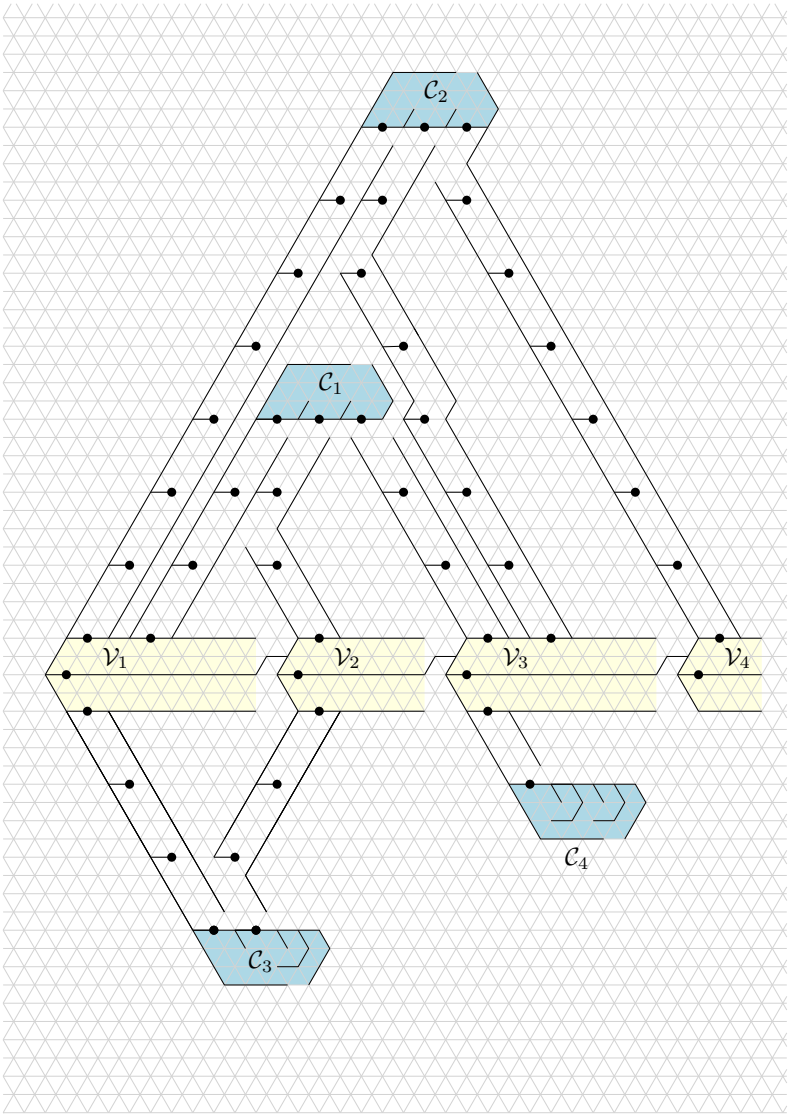


Figure 5 The unit-length partial EMST-drawing Γ'_φ for $\varphi = (v_1 \vee v_2 \vee v_3) \wedge (v_1 \vee v_3 \vee v_4) \wedge (\bar{v}_1 \vee \bar{v}_2) \wedge (\bar{v}_3)$. The clause gadgets are shaded blue, the variable gadgets are yellow.

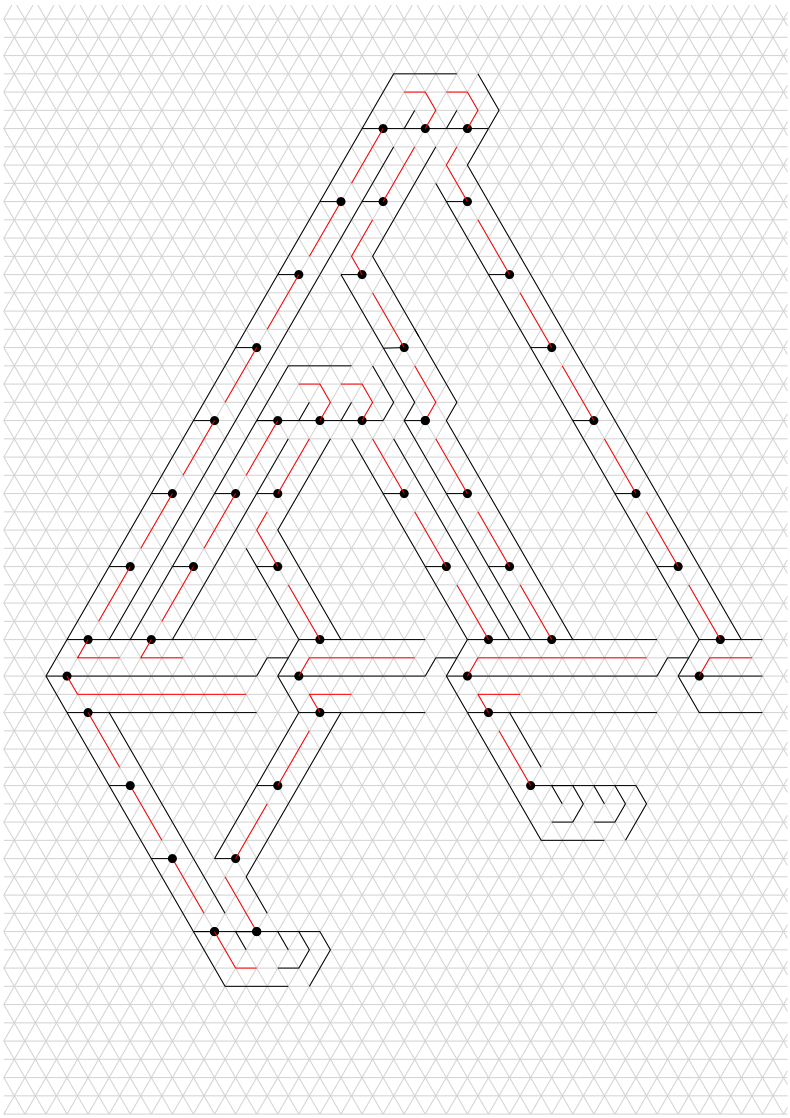


Figure 6 An EMST-extension (in red) of T_φ with respect to Γ'_φ of Figure 5 (in black).

▶ **Lemma 3.1.** *Let Γ' be a partial EMST-drawing of a tree T. For any vertex v of Γ' and for any $\varepsilon > 0$ there exists a disk $D(v)$ centered at v such that: (a) For any point $p \in D(v)$ and for any vertex $u \in \Gamma'$, we have $d(p,u) \geq \frac{1}{1+\varepsilon}|e_{\Gamma'}(u,v)|$. (b) For any two vertices u and v of Γ' and any two points $p \in D(u)$ and $q \in D(v)$, we have $d(p,q) \geq \frac{1}{1+\varepsilon}|e_{\Gamma'}(u,v)|$.*

We prove our result by describing an algorithm that, given a partial EMST-drawing of a tree T, computes an EMST-extension of T; we will use as a subroutine an algorithm from [4]. We recall in the next lemma the properties of the drawings computed by this algorithm that are relevant for our purposes.

▶ **Lemma 3.2** ([4], Theorem 3)**.** *Let T be a tree of maximum vertex degree $\Delta > 2$; there exists an algorithm that, for any $\varepsilon > 0$, computes a $(1+\varepsilon)$-EMST drawing Γ of T such that: (i) Γ is completely contained in a quadrant Q of a disk of radius less than $n^{2+\log_2 3c^\Delta}$, where c is a constant such that $c > \max\{\frac{\varepsilon+1}{\varepsilon}, \frac{1}{\sin\frac{\pi}{\Delta-1}}\}$; (ii) the shortest edge of Γ has length at least one; (iii) the sub-drawing Γ_k of Γ consisting of the first k children of the root of T, for $1 \leq k \leq \Delta$ is completely contained in a sub-sector of Q of internal angle $\frac{k\pi}{2(\Delta-1)}$. Also, this drawing can be computed in $O(n)$ time assuming the real RAM model of computation.*

▶ **Theorem 1.2.** *Let T be a tree whose maximum vertex degree is Δ, let $T' \subset T$ be a subtree of T and let Γ' be an EMST-drawing of T'. For any $\varepsilon > 0$, there exists a $(1+\varepsilon)$-EMST drawing Γ of T such that $\Gamma' \subset \Gamma$ and the edge-length ratio of Γ is $O(n^{2+\log_2 3c^\Delta})$, where c is a constant such that $c > \max\{\frac{\varepsilon+1}{\varepsilon}, \frac{1}{\sin\frac{\pi}{\Delta-1}}\}$. Also, this drawing can be computed in $O(n)$ time assuming the real RAM model of computation.*

Sketch of proof. Choose as the root of T any vertex that is already drawn in Γ' and let w be a vertex of T whose descendants are not all drawn in Γ'. Let T_w be the subtree of T rooted at w consisting of all descendants of w not yet drawn. Notice that, w has degree at most $\Delta - 1$ in T_w. We add dummy children to vertex w so that it has degree 2Δ in T_w.

Let $D(w)$ be the disk centered at w that satisfies Lemma 3.1. We construct a $(1+\varepsilon)$-drawing Γ_w of T_w by using the algorithm of Lemma 3.2. We scale down this drawing so that the quadrant containing it has radius that is at most the radius r_ε of $D(w)$. To this aim, we define the scaling factor $\sigma = \frac{r_\varepsilon}{n^{2+\log_2 3c^\Delta}}$. Since, by Lemma 3.2 the drawing Γ_w is contained in a disk of radius less than $n_w^{2+\log_2 3c^\Delta}$, where n_w is the number of vertices of Γ_w, when scaled by the factor σ, it will be contained in a disk of radius less than r_ε. Further, if we remove the dummy vertices that we added to increase the degree of w in T_w we obtain a drawing that is completely contained in a circular sector of radius r_ε and angle at most $\frac{(\Delta-1)\pi}{2(2\Delta-1)} \leq \frac{\pi}{4}$ by property (iii) of Lemma 3.1 (recall that the degree of w in T_w is at most $\Delta - 1$ and we computed the drawing for a tree of vertex degree 2Δ). Let e_1 be an edge incident to w in Γ' and let e_2 the edge incident on w in Γ' that follows e_1 in the clockwise order around w (notice that if w has degree one in Γ', then $e_2 = e_1$). Let α be the clockwise angle between e_1 and e_2 ($\alpha = 2\pi$ if $e_1 = e_2$). Since Γ' is an EMST-drawing, then $\alpha \geq \frac{\pi}{3}$. It follows that Γ_w can be attached to Γ so that it is completely contained in $D(w)$ inside the angle α. Hence, the intersection between Γ' and Γ_w only consists of the vertex w. By applying the drawing procedure above to every vertex of Γ with at least one descendant not yet drawn, we obtain a drawing Γ of T. Notice that all sub-drawings are scaled by the same scaling factor σ.

It is possible to prove that for every pair of vertices the $(1+\varepsilon)$-EMST property holds.

It remains to prove that the edge-length ratio of Γ is $\rho(\Gamma) \in O(n^{2+\log_2 3c^\Delta})$. Denote by L_{max} (resp. L_{min}) the length of the longest (resp. shortest) edge of Γ'. Analogously, denote by ℓ_{max} (resp. ℓ_{min}) the length of the longest (resp. shortest) edge in the sub-drawing $\Gamma \setminus \Gamma'$. We have $\rho(\Gamma) = \frac{L_{max}}{\ell_{min}}$, which cam be rewritten as $\rho(\Gamma) = \frac{L_{max}}{L_{min}} \frac{L_{min}}{\ell_{max}} \frac{\ell_{max}}{\ell_{min}}$. The ratio $\frac{L_{max}}{L_{min}}$ is

the edge-length ratio $\rho(\Gamma')$ of Γ'; it is a given value not depending on the algorithm. The ratio $\frac{L_{min}}{\ell_{max}}$ is at most $\frac{L_{min}}{r_\varepsilon} = \frac{3(1+\varepsilon)}{\varepsilon}$ because ℓ_{max} is at most r_ε once the computed sub-drawings are scaled down to fit inside the disks of radius r_ε. Let e_{max} be the edge of length ℓ_{max} and assume it belongs to a subtree T_w with n_w vertices; let e_{min} be the edge of length ℓ_{min}. Denote by ℓ'_{max} and ℓ'_{min} the lengths of e_{max} and of e_{min}, respectively, before scaling down the sub-drawings. By Lemma 3.2 $\ell'_{min} \geq 1$ and $\ell'_{max} \leq n_w^{2+\log_2 3c^\Delta} \leq n^{2+\log_2 3c^\Delta}$. Since all sub-drawings are scaled down by the same factor α, we have $\frac{\ell_{max}}{\ell_{min}} = \frac{\ell'_{max}}{\ell'_{min}} \leq n^{2+\log_2 3c^\Delta}$. In conclusion, we have $\rho(\Gamma) \leq \rho(\Gamma')\frac{3(1+\varepsilon)}{\varepsilon} n^{2+\log_2 3c^\Delta}$. ◀

4 Open problems

We conclude the paper with three open problems that in our opinion are worth investigating.

- Study the complexity of the EMST-Extension problem for paths.
- Study a variant of the EMST-Extension problem in which Γ' is not connected.
- Study extension problems for other families of proximity drawings other than EMSTs.

References

1 Mark de Berg and Amirali Khosravi. Optimal binary space partitions in the plane. In My T. Thai and Sartaj Sahni, editors, *Computing and Combinatorics, 16th Annual International Conference, COCOON 2010, Nha Trang, Vietnam, July 19-21, 2010. Proceedings*, volume 6196 of *Lecture Notes in Computer Science*, pages 216–225. Springer, 2010. doi:10.1007/978-3-642-14031-0_25.

2 Peter Eades and Sue Whitesides. The realization problem for euclidean minimum spanning trees in np-hard. *Algorithmica*, 16(1):60–82, 1996. doi:10.1007/BF02086608.

3 Fabrizio Frati and Michael Kaufmann. Polynomial area bounds for MST embeddings of trees. *Comput. Geom.*, 44(9):529–543, 2011. URL: https://doi.org/10.1016/j.comgeo.2011.05.005, doi:10.1016/J.COMGEO.2011.05.005.

4 Emilio Di Giacomo, Walter Didimo, Giuseppe Liotta, and Henk Meijer. Drawing a tree as a minimum spanning tree approximation. *J. Comput. Syst. Sci.*, 78(2):491–503, 2012. URL: https://doi.org/10.1016/j.jcss.2011.06.001, doi:10.1016/J.JCSS.2011.06.001.

5 Michael Kaufmann. Polynomial area bounds for MST embeddings of trees. In Seok-Hee Hong, Takao Nishizeki, and Wu Quan, editors, *Graph Drawing, 15th International Symposium, GD 2007, Sydney, Australia, September 24-26, 2007. Revised Papers*, volume 4875 of *Lecture Notes in Computer Science*, pages 88–100. Springer, 2007. doi:10.1007/978-3-540-77537-9_12.

6 Clyde L. Monma and Subhash Suri. Transitions in geometric minimum spanning trees. *Discret. Comput. Geom.*, 8:265–293, 1992. doi:10.1007/BF02293049.

Graph Tile Connectability with Turn Tiles

Maarten Löffler[1] **and Ids de Vlas**[1]

1 **Utrecht University, The Netherlands**
{m.loffler|i.l.devlas}@uu.nl

Abstract

In a graph tiling problem, we are given a small set of tile types with multiplicities, and are asked to place them in a container such that their sides match. Each tile has a partial drawing of a graph on it. We consider a variant in which the resulting graph is required to be connected.

1 Introduction

Aichholzer *et al.* [1] investigate the *graph tiling problem*: given a set of simple *tiles* on which a piece of a graph is drawn and a square container, is it possible to put the tiles into the container so that all edges match? Specifically, they consider only six different tile layouts (Figure 1), and the input consists of six integers indicating the multiplicity of each tile, as well as an integer indicating the size of the container. They motivate their investigation from material engineering [7, 8].

In this note, we begin extending the classification in [1] to *connected* layouts, in which not only all tile sides need to be compatible, but the resulting graph must also be connected. This additional criterion makes the problem significantly more complex. The problem is related to edge tile matching problems studied by Demaine *et al.* in the context of self-assembly [2, 3], and specific variants have been considered and partially written down in the past [4, 5]. The problem is also related to Wang tiles [10], but with exact multiplicities, as well as to several games and puzzles in recreational mathematics [6, 9].

Contribution. We provide a partial classification of multiplicities for specific subsets of tile types used in [1] for which it is possible to tile a rectangular container such that the resulting graph is connected. All the tiles are square. One is empty and five have a single vertex in the middle, with half-edges extending to each of the four sides, or not. Taking rotations in account, this results in six different types; see Figure 1 for the complete set with identifiers. We focus especially on instances involving tile type Ⅳ: the *turn tile*.

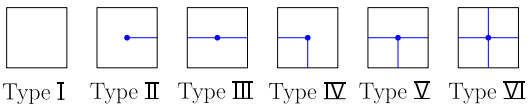


Figure 1 The tileset considered in this work, with identifiers.

An instance of the problem is characterized by six integers $(N_{\text{Ⅰ}}, N_{\text{Ⅱ}}, N_{\text{Ⅲ}}, N_{\text{Ⅳ}}, N_{\text{Ⅴ}}, N_{\text{Ⅵ}})$, the multiplicities of each tile, and two more integers (m, n), the dimensions of the desired

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

output region. We will not take into account any instances where the total multiplicities of the tiles does not exactly cover the output region; the following condition must hold: $N_{\text{I}} + N_{\text{II}} + N_{\text{III}} + N_{\text{IV}} + N_{\text{V}} + N_{\text{VI}} = m \cdot n$. An instance is a YES-instance if it is possible to tile the $m \times n$ area using the multiplicities of each tile such that every tile placement is valid, there may be no dangling half-edges (but half-edges extending to the boundary of the container are allowed). Furthermore, the resulting graph must be connected.

Following [1], we divide the problem in *cases*, where every case corresponds to a subset of the tile types: those types for which the multiplicity is nonzero. So, for example, the instance $(0, 3, 1, 0, 0, 5)$ belongs to the case $\langle \text{II}, \text{III}, \text{VI} \rangle$. We further group cases into *classes*.

The YES-class. The YES-class is a class of cases that only contain YES-instances.
The NO-class. The NO-class is the impossible counterpart of the YES-class; the cases in this class are either never possible or only possible in instances with trivial dimensions.
The PL-class. The PL-class is a class of cases that are possible if and only if the tiles are placable in the grid (regardless of connectivity). Here we inherit the algorithms from [1].
The P-class. The P-class contains cases for which there exists a polynomial-time algorithm to distinguish YES-instances from NO-instances.
The NP-class. Finally, the NP-class contains cases for which no polynomial-time algorithm to distinguish YES-instances from NO-instances exists, unless P=NP.

Results. We identify all cases in the YES-, NO-, and PL-classes. The YES-class is small, with only the following three cases: $\{\langle \text{VI} \rangle, \langle \text{V} \rangle, \langle \text{V}, \text{VI} \rangle\}$. The class is exhaustive, these are the only three cases in this class, including any bigger subsets using the same tiles.

The NO-class has the following seven cases and is once again exhaustive.
$\{\langle \text{I} \rangle, \langle \text{II} \rangle, \langle \text{III} \rangle, \langle \text{IV} \rangle, \langle \text{I}, \text{VI} \rangle, \langle \text{II}, \text{III} \rangle, \langle \text{II}, \text{IV} \rangle\}$

Finally, we argue that the following cases are part of the PL-class: $\{\langle \text{I}, \text{V} \rangle, \langle \text{III}, \text{VI} \rangle, \langle \text{I}, \text{III}, \text{VI} \rangle, \langle \text{I}, \text{V}, \text{VI} \rangle\}$

For the remaining cases, we provide polynomial-time algoririthms for a number of cases; see Figure 2. It remains open whether any cases belong to the NP-class.

$\langle \text{I} \rangle$	NO	$\langle \text{I}, \text{IV} \rangle$	P	$\langle \text{III}, \text{VI} \rangle$	PL	$\langle \text{I}, \text{III}, \text{IV} \rangle$	?	$\langle \text{II}, \text{IV}, \text{V} \rangle$	?
$\langle \text{II} \rangle$	NO	$\langle \text{I}, \text{V} \rangle$	PL	$\langle \text{IV}, \text{V} \rangle$	?	$\langle \text{I}, \text{III}, \text{V} \rangle$	?	$\langle \text{II}, \text{IV}, \text{VI} \rangle$	?
$\langle \text{III} \rangle$	NO	$\langle \text{I}, \text{VI} \rangle$	NO	$\langle \text{IV}, \text{VI} \rangle$	?	$\langle \text{I}, \text{III}, \text{VI} \rangle$	PL	$\langle \text{II}, \text{V}, \text{VI} \rangle$	?
$\langle \text{IV} \rangle$	NO	$\langle \text{II}, \text{III} \rangle$	NO	$\langle \text{V}, \text{VI} \rangle$	YES	$\langle \text{I}, \text{IV}, \text{V} \rangle$	?	$\langle \text{III}, \text{IV}, \text{V} \rangle$	?
$\langle \text{V} \rangle$	YES	$\langle \text{II}, \text{IV} \rangle$	NO			$\langle \text{I}, \text{IV}, \text{VI} \rangle$	?	$\langle \text{III}, \text{IV}, \text{VI} \rangle$	?
$\langle \text{VI} \rangle$	YES	$\langle \text{II}, \text{V} \rangle$	?	$\langle \text{I}, \text{II}, \text{III} \rangle$	P	$\langle \text{I}, \text{V}, \text{VI} \rangle$	PL	$\langle \text{III}, \text{V}, \text{VI} \rangle$	P
		$\langle \text{II}, \text{VI} \rangle$	P	$\langle \text{I}, \text{II}, \text{IV} \rangle$	P	$\langle \text{II}, \text{III}, \text{IV} \rangle$	?	$\langle \text{IV}, \text{V}, \text{VI} \rangle$	?
$\langle \text{I}, \text{II} \rangle$	P	$\langle \text{III}, \text{IV} \rangle$	?	$\langle \text{I}, \text{II}, \text{V} \rangle$	?	$\langle \text{II}, \text{III}, \text{V} \rangle$	?		
$\langle \text{I}, \text{III} \rangle$	P	$\langle \text{III}, \text{V} \rangle$	P	$\langle \text{I}, \text{II}, \text{VI} \rangle$	P	$\langle \text{II}, \text{III}, \text{VI} \rangle$	P		

Figure 2 All cases with up to three different tile types, and their classes.

Outline. Our most interesting results all involve cases that include tile type IV. As such, we devote the remainder of this note to our observations about this tile type in Section 2. In the full version we will list all of our results exhaustively, including arguments about several cases without IV - these arguments are all straightforward.

2 Type IV: the Turn Tile

The following cases have been grouped together since they all contain the type IV tile.

2.1 Cases $\langle \text{IV} \rangle$ and $\langle \text{II}, \text{IV} \rangle$

We start with taking a quick look at the two cases in the NO-class in this group: $\langle \text{IV} \rangle$, $\langle \text{II}, \text{IV} \rangle$.

A graph with only type IV tiles will only have vertices with degree 2. This means that the entire graph must be either a path with two half-edges connecting to the border of the grid, or a loop. The 2×2 squares at a corner of the grid can only be filled with at least one half-edge to the border, or with a four vertex disconnected cycle. Since there can be at most two half-edges to the border and there are four corners of the grid, it is impossible to make the entire graph connected if $m \geq 3 \vee n \geq 3$. Adding type II does not help. It is now possible to make a path that starts or ends in a type II tile instead of the border of the grid. The same argument still holds though, making this case just as impossible.

2.2 Case $\langle \text{I}, \text{IV} \rangle$

Adding type I tiles sometimes makes the problem possible. We are still limited to a single path or loop, but this path or loop does not have to cover the entire grid though. A key observation here can be made while following the path made by the type IV tiles. These type IV tiles can be grouped in groups of two consecutive tiles, where every group is in a single row. When colouring the rows in alternating colours, every group will have a different colour than its two neighbouring groups. We will refer to such a colouring as a *row colouring*. A similar setup can be used for the columns, creating a *column colouring*. A column colouring will have the same groups as the row colouring but shifted by one. When overlaying these two colourings, we get a colouring using four colours, that we refer to as a *full colouring*. Every four consecutive tiles will always be of the four different colours.

We will use the full colours as represented in Figure 3: In a row colouring, the bottom row is white and every white row has two adjacent blue rows. In a column colouring, the leftmost column is white and every white column has two adjacent red rows. In a full colouring, the bottom left tile will be white, every white square will have two red neighbours in the horizontal directions, two blue neighbours in the vertical directions, and 4 purple neighbours in the diagonal directions.

▶ **Lemma 2.1.** *Any layout of type I and IV tiles in an $m \times n$ grid where m and n are both odd and the type IV tiles are arranged to make a loop will use at least $m + n - 1$ type I tiles.*

Proof. Since m and n are odd, the outer ring of the grid in a full colouring cannot contain purple tiles: there are only $\lfloor \frac{m}{2} \rfloor \cdot \lfloor \frac{n}{2} \rfloor$ purple tiles. Since four consecutive tiles will be of four different colours, in a loop of type IV tiles, one in four tiles is purple. A loop of type IV tiles using all purple tiles will use $(m-1)(n-1)$ tiles. Since all types are either type I or type IV the remaining $m + n - 1$ tiles must be type I tiles. ◀

▶ **Corollary 2.2.** *Any connected layout of type I and IV tiles in an $m \times n$ grid where m and n are both odd will use at least $m + n - 4$ type I tiles.*

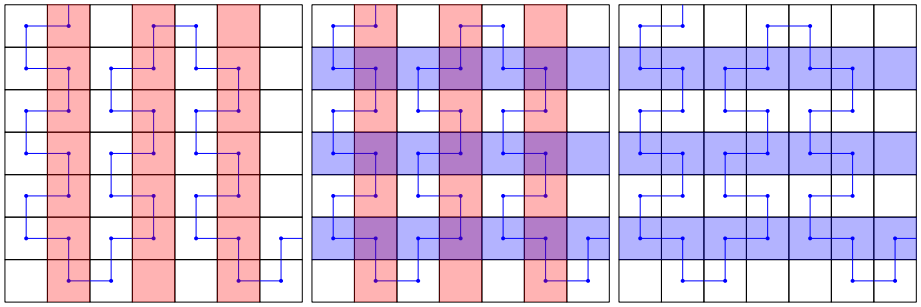


Figure 3 Three ways to colour the $\langle \text{I}, \text{IV} \rangle$ tileset. The middle colouring is created by overlaying the left and right colourings, forcing every four consecutive tiles in different colours.

Proof. This follows from Lemma 2.1. A cycle using $mn - m - n + 1$ type IV tiles can be broken up right after a purple tile, allowing three extra tiles of the other colours to be placed after it, for a total of $(m-1)(n-1)+3$ type IV tiles. This leaves $m+n-4$ type I tiles to fill up the rest of the space. ◀

▶ **Lemma 2.3.** *Any connected layout of type I and IV tiles in an $m \times n$ grid where m is odd and n is even will use at least $n-2$ type I tiles.*

Proof. Since m is odd, the top and bottom row in a row colouring are both white. There are $\lfloor \frac{m}{2} \rfloor \cdot n$ blue tiles and $\lceil \frac{m}{2} \rceil \cdot n$ white tiles. Every type IV tile can be grouped in groups of two consecutive tiles, where every group is in a single row. Since consecutive groups will be of a different colour, any layout can have at most two more white than blue tiles. If all blue tiles are filled with type IV tiles, at most $(m-1) \cdot n + 2$ tiles can be filled by type IV tiles. The remaining $n-2$ tiles must be type I tiles. ◀

For the instances where m and n are both even, we need a little more work. Since all four colours in a full colouring would have the same number of tiles, we need another upper bound.

▶ **Lemma 2.4.** *Any connected layout of type I and IV tiles in an $m \times n$ grid where m and n are both even and $m \geq n$ will use at least $2n-6$ type I tiles.*

Proof. Omitted for space; see full version for proof. ◀

We now have lower bounds on the number of type I tiles in three cases, depending on the parity of n and m. For all three cases we can find a corresponding construction. The layout starts in the bottom left, then uses a zigzagging pattern in the leftmost two columns to the top of the grid, before turning around and using another zigzagging pattern in the next two columns. The pattern will continue until it reaches a corner on the right of the grid. If both m and n are even, an extra turn at the end is used. See Figure 4 for an example. See full version for details on layouts using more than the minimum number of I tiles.

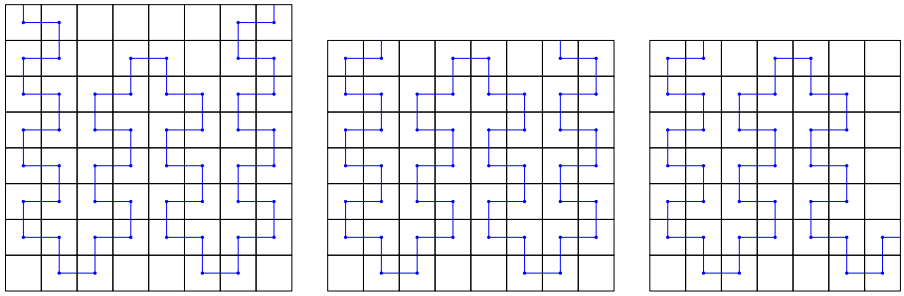


Figure 4 Example layouts using only I and IV tiles, using the minimum number of I tiles. The three subfigures represent the three variations with even m and n (left), odd m and even n (middle), and odd m and n (right).

When adding type II tiles to the mix most of the same lemmas still hold. There can be at most two type II tiles at the ends of the type IV path, replacing type IV tiles that were placed there in the layout using only type I and IV tiles. For the case where m and n are both even, the amount of turns in one direction is no longer limited by the rotation number. Allowing for a spiral pattern into the center of the grid.

2.3 Case $\langle$III, IV$\rangle$

Next we will take a look at the case using only type III and IV tiles. This case with the additional requirement that the resulting graph is a *cycle* has been studied before by Demaine *et al.* [4], and later by Iacono *et al.* [5], who give closed-form solutions for many values of n and m. Note, however, that a cycle is never possible when m and n are both odd; we focus on this case here.

We will first analyze the effect of type III tiles in a fragment of a type IV tile path over a full colouring, we will look at a segment of the path between two purple tiles. The type III tile will always flip the ordering of the tiles. Furthermore, it will add or remove a number of tiles before the next purple tile is reached. For details of this placement see Figure 5.

▶ **Lemma 2.5.** *Any connected layout of type III and IV tiles in an $m \times m$ grid where m is odd will use at least $m - 1$ type III tiles.*

Proof. Since m is odd, there are $\lfloor \frac{m}{2} \rfloor^2$ purple tiles and $\lceil \frac{m}{2} \rceil^2$ white tiles. Since every four consecutive tiles in a path of only type IV tiles will contain one of each colour, the highest number of tiles of these colours that is coverable with only type IV tiles is the following.

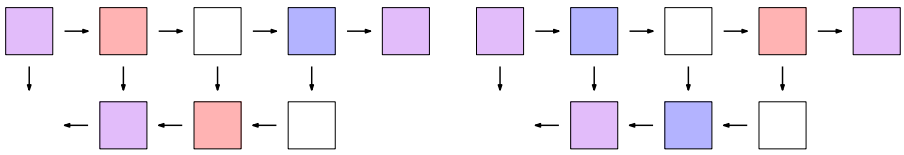


Figure 5 Various ways to insert a type III tile in a path of type IV tiles over a full colouring between two purple tiles. Horizontal arrows represent the colour transition if a type IV tile is placed at the base, vertical arrows represent the colour transition if a type III tile is placed at the base.

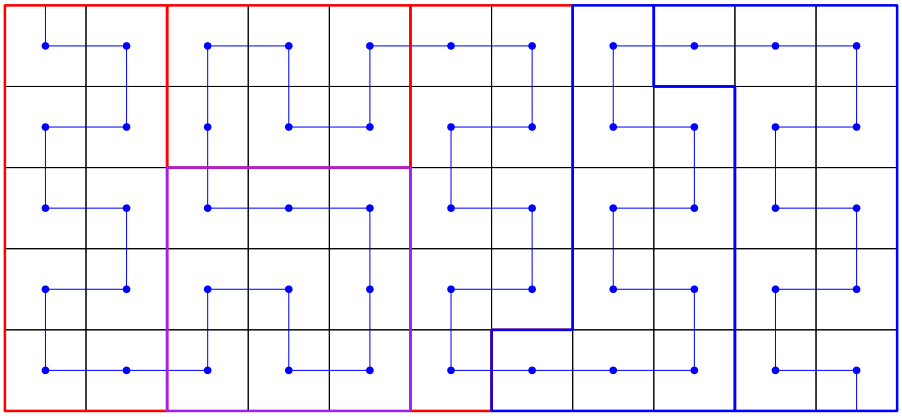


Figure 6 An extendable layout for type III and IV tiles maintaining the minimum number of type III tiles. The starting 3×3 area is shown in purple, containing two type III tiles. The extensions that maintain the spiral pattern are shown in red, containing one type III tile per red area. The extensions that diverge from the spiral pattern are shown in blue, containing two type III tiles. The blue extensions make a change in the previous smaller layout.

$\lfloor \frac{m}{2} \rfloor^2$ purple tiles, $\lfloor \frac{m}{2} \rfloor^2 + 1$ white tiles. There are $m - 1$ white tiles that still need to be covered. As seen in Figure 5, it is not possible to add more than one white tile between two purple tiles with one type III tile. The remaining white tiles thus each need a type III tile, resulting in at least $m - 1$ type III tiles. ◀

▶ **Lemma 2.6.** *Any connected layout of type III and IV tiles in an $m \times n$ grid where m and n are both odd will have a path that start and ends in a white tile.*

Proof. Grouping purple and white tiles into one colour and blue and red tiles into another colour, we get a checkerboard pattern. Since the dimensions are odd, there is one more purple/white tile than blue/red. Every tile along the path must alternate between these two groups, so the path must start and end in the purple/white group. There are no purple tiles adjacent to the edge of the grid, so the path must start and end with a white tile. ◀

▶ **Lemma 2.7.** *Any connected layout of type III and IV tiles in an $m \times n$ grid where m and n are both odd and $m \neq n$ will use at least $\max(m, n) - 2$ type III tiles.*

Proof. Omitted for space; see full version for proof. ◀

We now have two lower bounds on the number of type III tiles needed if m and n are both odd. A layout can be found for a 3×3 grid using the minimum number of type III tiles. This layout can be repeatedly extended by two rows or columns, maintaining the minimum number of type III tiles. The layout forms a double spiral pattern if m and n stay within two of each other. If one of m or n is a lot bigger than the other, a zigzagging pattern at one side will be added instead. The layout can be seen in Figure 6. While we have established a lower bound for the amount of type III tiles when m and n are both odd, the more general case for type III and IV tiles is still left open.

References

1 Oswin Aichholzer, Robert Ganian, Phillip Keldenich, Maarten Löffler, Gert G. T. Meijer, Alexandra Weinberger, and Carola Wenk. Graph tiles (poster abstract). In Vida Dujmovic and Fabrizio Montecchiani, editors, *33rd International Symposium on Graph Drawing and Network Visualization, GD 2025, Norrköping, Sweden, September 24-26, 2025*, volume 357 of *LIPIcs*, pages 51:1–51:5. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2025. URL: https://doi.org/10.4230/LIPIcs.GD.2025.51, doi:10.4230/LIPICS.GD.2025.51.

2 Erik D. Demaine and Martin L. Demaine. Jigsaw puzzles, edge matching, and polyomino packing: Connections and complexity. *Graphs and Combinatorics*, 23(1):195–208, 2007. doi:10.1007/s00373-007-0713-4.

3 Erik D. Demaine, Martin L. Demaine, Sandor P. Fekete, Mashhood Ishaque, Eynat Rafalin, Robert T. Schweller, and Diane Souvaine. Staged self-assembly: Nanomanufacture of arbitrary shapes with O(1) glues, 2008. URL: https://arxiv.org/abs/0803.0316.

4 Erik D. Demaine, Martin L. Demaine, Benjamin S. Horn, and Diane L. Souvaine. Assembling continuous paths with tiles. Unpublished, personal communication, 2010.

5 John Iacono, Maarten Löffler, and Birgit Vogtenhuber. Hamiltonian tilings with many bends. Unpublished, personal communication, 2016.

6 Jürgen Köller. Snake cube, 1999. Accessed: 12 January 2026. URL: https://www.mathematische-basteleien.de/snakecube.htm.

7 Ke Liu, Rachel Sun, and Chiara Daraio. Growth rules for irregular architected materials with programmable properties. *Science*, 377:975–981, 2022. doi:10.1126/science.abn1459.

8 Tommaso Magrini, Chelsea Fox, Adeline Wihardja, Athena Kolli, and Chiara Daraio. Control of mechanical and fracture properties in two-phase materials reinforced by continuous, irregular networks. *Advanced Materials*, 36(6):2305198, 2024. doi:10.1002/adma.202305198.

9 Torsten Sillke. Lege-wege: A domino puzzle, 1993. Accessed: 12 January 2026. URL: http://www.math.uni-bielefeld.de/~sillke/PUZZLES/lege-wege/domino-path.

10 Hao Wang. Proving theorems by pattern recognition — II. *The Bell System Technical Journal*, 40(1):1–41, 1961. doi:10.1002/j.1538-7305.1961.tb03975.x.

Dynamic $(1+\varepsilon)$-Spanner in Disk Intersection Graphs

Sarita de Berg[1], Ivor van der Hoog[1], Eva Rotenberg[1], Johanne M. Vistisen[1], and Sampson Wong[2]

1 **IT University of Copenhagen, Denmark**
debe@itu.dk, ivva@itu.dk, erot@itu.dk, jomy@itu.dk
2 **University of Copenhagen, Denmark**
sampson.wong123@gmail.com

Abstract

We study dynamic disk intersection graphs and show how to construct and dynamically maintain a $(1+\varepsilon)$-spanner over the disk intersection graph of a dynamic set, $\mathcal{S}$, of n disks. All disks in $\mathcal{S}$ have diameters $\in [4, \Psi]$ for some predefined and fixed Ψ. The $(1+\varepsilon)$-spanner has size $O(n\varepsilon^{-2}\log n \log \Psi \log(1/\varepsilon))$. The space used to maintain the spanner is $O(\Psi^2\varepsilon^{-2} n \log n \log^2 \Psi)$ and a disk can be inserted or deleted in $O(\Psi^2\varepsilon^{-2}\log^4 n \log^2 \Psi \log^2(1/\varepsilon))$ time.

Funding This work was supported by the Danmarks Frie Forskningsfond Case no. 10.46540/3160-00020B, and by the VILLUM Foundation grant (VIL37507) "Efficient Recomputations for Changeful Problems".

1 Introduction

Given a set of geometric shapes, e.g. disks or axis-aligned squares, its intersection graph has (i) a vertex for every element in the set of shapes, and (ii) an edge between a pair of vertices if and only if their corresponding elements intersect. We consider the weighted intersection graph where the edge weights are the Euclidean distance between the respective centres of the shapes (in our case disks, see Figure 1). Intersection graphs [9], especially disk intersection graphs [5,6], have attracted considerable attention.

Given a graph, a spanner is a subgraph that approximately preserves the distances of the original graph. Formally, a $(1+\varepsilon)$-spanner $G' \subseteq G$ has the property that for any pair of vertices $u, v \in G$, we have $d_{G'}(u,v) \leq (1+\varepsilon)\, d_G(u,v)$, where $d_G(\cdot,\cdot)$ denotes the shortest path metric in G. See [11] and [10] for surveys on graph spanners and geometric spanners.

The problem of computing a spanner of a disk intersection graph has previously been studied. Fürer and Kasiviswanatha [7] showed that, given a disc intersection graph with n vertices, one can construct a $(1+\varepsilon)$-spanner with $O(n\varepsilon^{-2})$ edge in $O(n^{4/3+\delta}\varepsilon^{-4/3}\log^{2/3}\Psi)$ time, where $\delta > 0$ and Ψ is the ratio between the largest and the smallest radius disk. Kaplan, Mulzer, Roditty, Seiferth and Sharir [8] improve the running time to expected $O((n\varepsilon^{-2})\log^9 n \lambda_6(\log n))$ time, where $\lambda_s(\cdot)$ denotes the Davenport-Shinzel bound of order s.

In this paper, we study the problem of dynamically maintaining a spanner of a disk intersection graph under insertions and deletions of disks. The problem is motivated by mobile wireless networks and robotics, where one may want to maintain approximate shortest path distance information as the network changes over time.

Our results. We provide the first algorithm for maintaining a $(1+\varepsilon)$-spanner of a disk intersection graph under disk insertions and deletions. Let n be the present number of

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

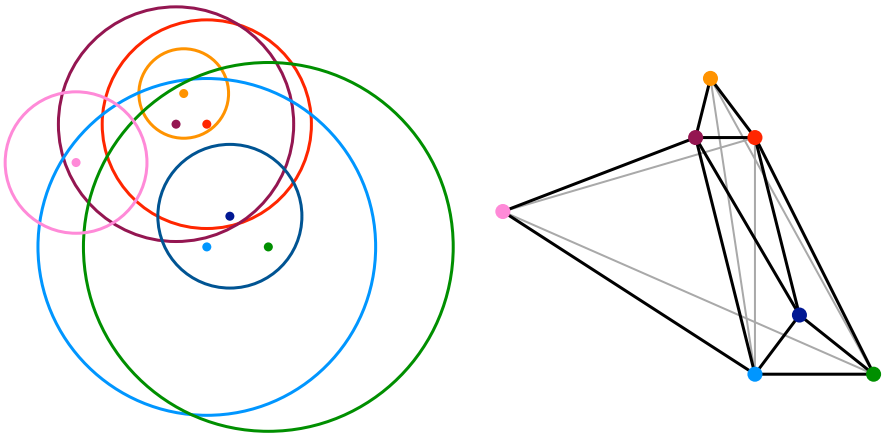


Figure 1 A set of disks (left), and the corresponding disk graph (right). Both the black and grey edges are edges of the disk graph, the fat black edges form a $(1+\varepsilon)$-spanner for some $\varepsilon > 0$.

disks. We assume that for a known input parameter Ψ, every disk has a radius between 4 and Ψ. Our dynamic spanner requires $O(\Psi^2\varepsilon^{-2}n\log n\log^2\Psi)$ space, and can be updated in $O(\Psi^2\varepsilon^{-2}\log^4 n\log^2\Psi\log^2(1/\varepsilon))$ expected amortised time under disk insertions and deletions.

Related work. A closely related problem is dynamic disk connectivity, which is to maintain the connected components of a disk intersection graph under disk insertions and deletions. Note that a dynamic spanner also maintains dynamic connectivity. Chan, Pătraşcu and Roditty [4] present a dynamic data structure with $O(n^{20/21+\varepsilon})$ update time, where $\varepsilon > 0$. Chan and Huang [3] improved the update time to $O(n^{7/8})$. If every disk has radius between 1 and Ψ, Kaplan, Mulzer, Roditty, Seiferth and Sharir [8] show that there is a data structure with expected amortised update time of $O(\Psi^2\log^{10} n)$. This was improved to $O(\Psi\log^7 n)$ time by Baumann, Kaplan, Klost, Knorr, Mulzer, Roddity and Seiferth [1].

2 Preliminaries

The input is a dynamic set $\mathcal{S} = (\sigma_1, \sigma_2, \ldots, \sigma_n)$ of n disks, where all disks are guaranteed to lie in some fixed bounding square B. To minimise numerical dependencies on $\sqrt{2}$, we define for any convex shape C the diameter $|C|$ of C as the length of the largest horizontal segment contained in C. We assume that, at all times, $\forall\sigma \in \mathcal{S}$, the diameter is at least 4 and at most Ψ, where Ψ is some fixed and known value. I.e., $|\sigma| \in [4, \Psi]$. Furthermore, the centres must be placed within a fixed bounding box of size $2^{(\Psi^2)} \times 2^{(\Psi^2)}$. We define the *disk intersection graph* $\mathcal{D}(\mathcal{S})$ as an edge-weighted graph where the vertex set is all disks in $\mathcal{S}$, and there is an edge (σ_1, σ_2) if and only if the two disks intersect. The weight of (σ_1, σ_2), denoted by $d(\sigma_1, \sigma_2)$, is the Euclidean distance between the two centre points. For any edge-weighted graph G over $\mathcal{S}$, we let $d_G(\sigma_1, \sigma_2)$ denote the length of the shortest path from σ_1 to σ_2.

Problem statement. An α-spanner of $\mathcal{D}(\mathcal{S})$ is any (edge-weighted) subgraph G of $\mathcal{D}(\mathcal{S})$ such that for any $\sigma_1, \sigma_2 \in S$, $d_G(\sigma_1, \sigma_2) \leq \alpha \cdot d_{\mathcal{D}(\mathcal{S})}(\sigma_1, \sigma_2)$. The *size* of an α-spanner is its number of edges. In this paper, we consider the problem of maintaining a $(1+\varepsilon)$-spanner of near-linear size subject to insertions and deletions in $\mathcal{S}$. The running time, space usage, and size of the spanner depend on n and the maximum diameter Ψ for disks in $\mathcal{S}$.

Dynamic Euclidean spanner. We use the following dynamic spanner in our construction:

▶ **Lemma 2.1** (Theorem 4.4 [2]). *For any planar point set P of size n there exists a dynamic $(1+\varepsilon)$-spanner of their pairwise Euclidean distances that has size $O(n\varepsilon^{-2}\log(1/\varepsilon))$. It can be maintained using $O(n\varepsilon^{-2}\log(1/\varepsilon))$ space and $O(\varepsilon^{-2}\log n\log^2(1/\varepsilon))$ update time.*

Quadtrees. We also use *quadtrees*, which are commonly used for disk intersection spanners [7]:

▶ **Definition 2.2.** Let B be an axis-aligned square whose side length is a positive power of two. A *split* operation selects a square and divides it into four equal squares. A *quadtree* is any tree obtained by recursively applying the split operation on B. The *complete quadtree* $\mathcal{T}_B$ is obtained by recursively applying the split operation until all leaves of the tree have size 1. Each node of $\mathcal{T}_B$ corresponds to a square in the plane, called a *cell*.

For any odd integer k, we denote by $k*C$ the region covered by a $k\times k$ grid of squares of size $|C|$ that has C as its centre. For any disk σ, we define its *storing cell* C_σ as the largest cell in $\mathcal{T}_B$ that contains the centre of σ and is contained in σ. The *storing family* $\mathcal{F}(\sigma)$ of σ are all descendants of C_σ in $\mathcal{T}_B$ that contain the centre of σ. Typically, one uses the minimum-size quadtree that contains all storing cells C_σ for $\sigma\in\mathcal{S}$ [1, 7]. We deviate from this standard by considering the minimum-size quadtree that contains all storing families:

▶ **Definition 2.3.** Given B and $\mathcal{S}$, we define $T(\mathcal{S})$ as the minimum-size (compressed) quadtree that contains the storing families $\mathcal{F}(\sigma)$ for all $\sigma\in\mathcal{S}$. For a cell $C\in T(\mathcal{S})$, its *population* $\pi(C)$ is all disks in $\mathcal{S}$ that have C in their storing family.

▶ **Observation 1.** *The (compressed) quadtree $T(\mathcal{S})$ has $O(n\log\Psi)$ cells. Moreover, $\forall C\in T(\mathcal{S})$, $\forall C'\in T(\mathcal{S})$ of size $|C|$ with $C'\subset 3*C$, all pairs $(\sigma,\sigma')\in\pi(C)\times\pi(C')$ intersect.*

▶ **Definition 2.4.** For a fixed $0<\varepsilon\leq 1$, we partition each cell $C\in T(\mathcal{S})$ into a grid of squares of side length $\varepsilon|C|$, called *ε-squares*. Let E be an ε-square of a cell C, then $\Gamma_\varepsilon(E)$ denotes all disks that have C as their storing cell and their centre in E. Analogously, $\pi_\varepsilon(E)$ denotes all disks in $\pi(C)$ that have their centre in E.

Separated intersection queries. Let $\mathcal{S}$ be a set of disks and k be a line where all centre points of $\mathcal{S}$ lie on one side of k. A *separated intersection query* (SIQ) consists of a disk ρ such that the centre point of ρ lies on the other side of k, and returns a disk in $\mathcal{S}$ that intersects ρ (if such a disk exists). We will use the data structure by [1]:

▶ **Lemma 2.5** (Lemma 2.4 [1]). *We can maintain $\mathcal{S}$ subject to $O(\log^2 n)$ time separated intersection queries (SIQ), with $O(n\log n)$ space and $O(\log^4 n)$ amortised expected update time.*

Proof. Lemma 2.4 in [1] presents a fully dynamic additively weighted nearest neighbour data structure with corresponding space and time bounds. We can insert all centres of $\mathcal{S}$ into this data structure, where the weight for each disk is the negative radius of the disk. For an intersection query with a disk ρ with centre point r, we query this data structure with r. Let the point corresponding to a disk σ be closest to r. If the distance is negative, then r is contained in σ. Otherwise, the returned distance d is the distance from r to the perimeter of $\mathcal{S}$ and ρ intersects a disk in $\mathcal{S}$ if and only if d is less than the radius of ρ. ◀

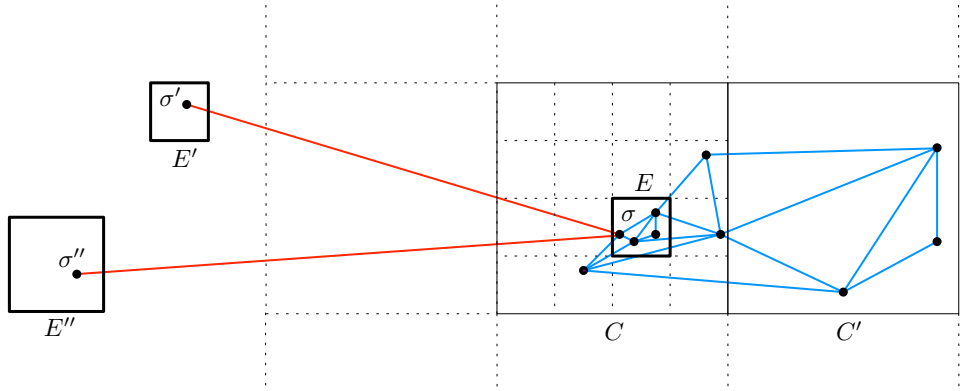


Figure 2 Our two types of edges where blue edges are Type i and red edges are Type ii.

3 Defining our spanner

We define a graph G that is a $(1+7\varepsilon)$-spanner of size $O(n\varepsilon^{-2}\log n \log \Psi \log(1/\varepsilon))$. By choosing $\varepsilon' = \varepsilon/7$, we obtain a $(1+\varepsilon)$-spanner of the same asymptotic size. Let $T(\mathcal{S})$ be the quadtree storing $\mathcal{S}$. Our spanner G has two types of edges (see Figure 2):

i. For each $C \in T(\mathcal{S})$, consider all cells $C' \in T(\mathcal{S})$ of size $|C|$ that are in $3 * C$. For each pair (C, C'), use Lemma 2.1 to construct a Euclidean $(1+\varepsilon)$-spanner on all centre points of disks in $\pi(C) \cup \pi(C')$. For each edge (s_1, s_2) in this spanner, add an edge to G between their corresponding disks σ_1 and σ_2. By Observation 1, (σ_1, σ_2) is in $\mathcal{D}(\mathcal{S})$.
ii. For each $C \in T(\mathcal{S})$ and $i \in [1, \lceil \log \Psi \rceil]$, consider the cells C' such that $|C'| = 2^{i-1}|C|$ and C' intersects $(2^{i+5}+1) * C$. For every pair of ε-squares (Def. 2.4) E of C and E' of C', if there exists a pair $(\sigma, \sigma') \in \Gamma_\varepsilon(E) \times \pi_\varepsilon(E')$ that intersect, then we select an arbitrary pair of intersecting disks $(\sigma_1, \sigma_2) \in \Gamma_\varepsilon(E) \times \pi_\varepsilon(E')$ and add (σ_1, σ_2) to G. We consider the edge (σ_1, σ_2) in G to be an ordered pair.

We will show that G not only preserves connectivity, but is a $(1+7\varepsilon)$-spanner of $\mathcal{D}(\mathcal{S})$. Specifically, we will show that for every edge $(\sigma_1, \sigma_2) \in \mathcal{D}(\mathcal{S})$ there exists a path in G from σ_1 to σ_2 of length at most $(1+7\varepsilon)\, d(\sigma_1, \sigma_2)$. To this end, consider the storing cells C_1 of σ_1, and C_2 of σ_2. Assume without loss of generality that $|C_1| \leq |C_2|$. We have two cases: either the centre of σ_2 lies inside $3 * C_1$ (Lemma 3.1) or not (Lemma 3.2).

▶ **Lemma 3.1.** *If the centre of σ_2 lies in $3 * C_1$ then there exists a path in G from σ_1 to σ_2 of length at most $(1+\varepsilon)\, d(\sigma_1, \sigma_2)$.*

Proof. Since the centre of σ_2 lies in $3 * C_1$, the cell C_2 must have a descendent C_2' of size $|C_1|$ that is contained in $3 * C_1$ where $\sigma_2 \in \pi(C_2')$. By construction of the Type i edges, G contains an Euclidean $(1+\varepsilon)$-spanner over all the disks in $\pi(C_1) \cup \pi(C_2')$. ◀

▶ **Lemma 3.2.** *If the centre of σ_2 does not lie in $3 * C_1$ then there exists a path in G from σ_1 to σ_2 of length at most $(1+7\varepsilon)\, d(\sigma_1, \sigma_2)$.*

Proof. Parts of the proof are illustrated by Figure 3. Let $\Delta = d(\sigma_1, \sigma_2)$. Recall that C_1 is the storing cell of σ_1 and consider the storing family $\mathcal{F}(\sigma_2)$ of σ_2. Since the centre of σ_2 is not in $3 * C_1$, $\Delta \geq |C_1|$ and so there exists an integer $j \geq 0$ such that $\Delta \in [2^j|C_1|, 2^{j+1}|C_1|)$.

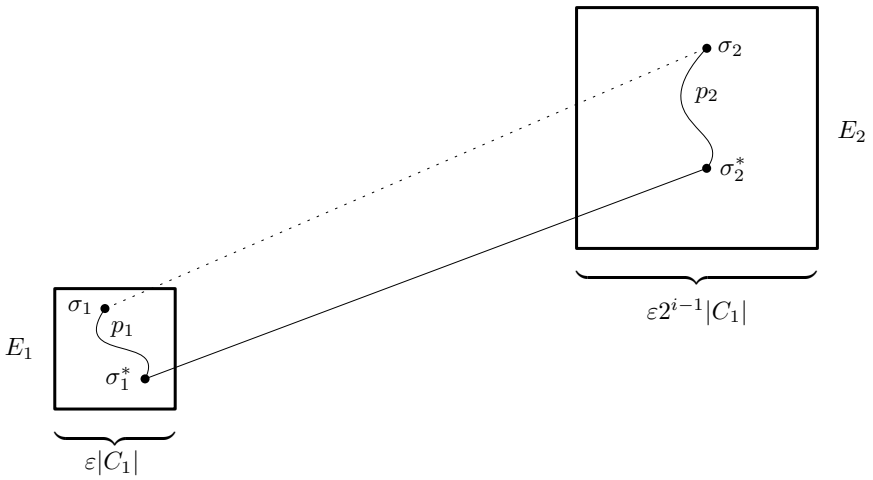


Figure 3 The path $\sigma_1 \to p_1 \to \sigma_1^* \to \sigma_2^* \to p_2 \to \sigma_2$ and the (dotted) edge (σ_1, σ_2).

Next, we find a cell C' such that there exists some Type ii edge between C_1 and C'. We consider two cases: either $0 \leq j \leq 3$ or $j > 3$.

If $0 \leq j \leq 3$ then $\Delta \in [|C_1|, 2^3|C_1|)$. As $|C_2| \geq |C_1|$, there exist a cell C' for which $|C'| = |C_1|$ and $C' \in \mathcal{F}(\sigma_2)$. We know that $|C'|$ lies in $(2^4+1) * C_1$ because $\Delta < 2^3|C_1|$. This implies that there exists a Type ii edge between C and C' (because for $i = 1$ we have $|C'| = 2^{i-1}|C_1|$ and C' intersects $(2^4+1) * C_1 \subset (2^{i+5}+1) * C_1$).

Now for the case of $j > 3$, we consider the positive integer $i := j - 3$, i.e. $\Delta \in [2^{i+3}|C_1|, 2^{i+4}|C_1|)$. The first observation is then that Δ is at most $\frac{1}{2}(|\sigma_1| + |\sigma_2|)$. Moreover, since $|C_2| \geq |C_1|$, $|\sigma_2| \geq \frac{1}{4}|\sigma_1|$ and so $\Delta \leq \frac{5}{2}|\sigma_2|$. It follows that $|C_2| \geq \frac{|\sigma_2|}{4\sqrt{2}} \geq \frac{2}{20\sqrt{2}}\Delta \geq \frac{2}{20\sqrt{2}}2^{i+3}|C_1| \geq 2^{i-1}|C_1|$. This means that there exists a $C' \in \mathcal{F}(\sigma_2)$ with $|C'| = 2^{i-1}|C_1|$. Because σ_1 and σ_2 intersect and their centres are contained in C_1 and C', the distance between C_1 and C' is at most Δ, which is bounded by $2^{i+4}|C_1|$. Thus, C' intersects $(2^{i+5}+1) * C_1$ (and $|C'| = 2^{i-1}|C_1|$) so we meet the conditions to have edges of Type ii between disks that have C_1 as their storing cell, and disks are in $\pi(C')$.

Next, denote by E_1 the ε-square of C_1 that contains the centre of σ_1 and by E_2 the ε-square of C' that contains the centre of σ_2. The pair (σ_1, σ_2) is a pair in $\Gamma_\varepsilon(C_1) \times \pi_\varepsilon(C')$ that intersect, and so there exists a Type ii edge between some $(\sigma_1^*, \sigma_2^*) \in \Gamma_\varepsilon(C_1) \times \pi_\varepsilon(C')$. Note that $d(\sigma_1, \sigma_1^*) \leq \sqrt{2}\varepsilon|C_1| \leq \sqrt{2}\varepsilon\Delta$. Similarly, $d(\sigma_2, \sigma_2^*) \leq \sqrt{2}\varepsilon|C'| \leq \sqrt{2}\varepsilon\Delta$ and so by triangle inequality, $d(\sigma_1^*, \sigma_2^*) \leq (1+3\varepsilon)\Delta$. We observe that σ_1 and σ_1^* both are in $\pi(C_1)$. Lemma 3.1 guarantees that there exists a path p_1 in G of Type i edges from σ_1 to σ_1^* whose length is at most $(1+\varepsilon)\, d(\sigma_1, \sigma_1^*) \leq (1+\varepsilon)\varepsilon|C_1| \leq 2\varepsilon\Delta$. Analogously, there exists a path p_2 of Type i edges from σ_2^* to σ_2 of length at most $(1+\varepsilon)\, d(\sigma_2, \sigma_2^*) \leq (1+\varepsilon)\varepsilon|C'| \leq 2\varepsilon\Delta$. We have now found a path $\sigma_1 \to p_1 \to \sigma_1^* \to \sigma_2^* \to p_2 \to \sigma_2$ in G of length at most $(1+7\varepsilon)\,\Delta$. ◀

The result is that G is a $(1+7\varepsilon)$-spanner of near-linear size.

▶ **Theorem 3.3.** *The graph G is a $(1+7\varepsilon)$-spanner of size $O(n\varepsilon^{-2}\log n \log\Psi \log(1/\varepsilon))$, which, given $\mathcal{D}(\mathcal{S})$, can be constructed in total time $O(n\varepsilon^{-2}(\log^4 n + \log n \log\Psi \log^2(1/\varepsilon)))$.*

4 Dynamically maintaining our $(1+7\varepsilon)$-spanner

To dynamically maintain G, we augment our data structure. Recall that Type ii edges (σ, σ') are ordered. We know that σ has a storing cell C, σ' has a storing cell C' and $|C'| = 2^{i-1}|C|$. The following intuitively states that for a fixed σ, there exist $O(\log \Psi)$ edges $(\sigma, \cdot)$. Yet, for a fixed σ', there exist $O(\Psi^2 \log \Psi)$ edges $(\cdot, \sigma')$. For the first case, consider for each $C \in T(\mathcal{S})$, for all $i \in [1, \lceil \log \Psi \rceil]$, all C' in the *complete quadtree* $\mathcal{T}_B$ with $|C'| = 2^{i-1}|C|$ where C' intersects $(2^{i+5}+1) * C$. We maintain an object that represents C'. Alternatively, one can insert C' into $T(\mathcal{S})$. For the second case, consider for each $C' \in T(\mathcal{S})$, for all $i \in [1, \lceil \log \Psi \rceil]$, all $O(\Psi^2)$ cells C in the *complete quadtree* $\mathcal{T}_B$ with $|C'| = 2^{i-1}|C|$ where C' intersects $(2^{i+5}+1) * C$. We also maintain an object that represents C. Alternatively, one can insert C into $T(\mathcal{S})$.

Using these objects, we augment the quadtree $T(\mathcal{S})$ with three data structures.

Maximal Bichromatic Matchings. Consider for each $C \in T(\mathcal{S})$, all $i \in [1, \lceil \log \Psi \rceil]$, all C' in the *complete quadtree* $\mathcal{T}_B$ with $|C'| = 2^{i-1}|C|$ that intersect $(2^{i+5}+1) * C$. For every pair (E, E') where E is an ε-square of C and E' is an ε-square of C', we consider the bipartite graph $\Gamma_\varepsilon(E) \times \pi_\varepsilon(E')$ where there is an edge $(\sigma_1, \sigma_2) \in \Gamma_\varepsilon(E) \times \pi_\varepsilon(E')$ if these two disks intersect. We maintain a Maximal Bichromatic Matching $M(E, E')$ of this graph.

Separated Intersection Query I (SIQ I). Consider for each $C \in T(\mathcal{S})$, all $i \in [1, \lceil \log \Psi \rceil]$ and all C' in the *complete quadtree* $\mathcal{T}_B$ with $|C'| = 2^{i-1}|C|$ and C' intersects $(2^{i+5}+1) * C$. Consider every pair (E, E') where E an ε-square of C and E' an ε-square of C'. We maintain the data structure $G(E, E')$ which stores all disks in $\Gamma_\varepsilon(E)$, minus the disks in $M(E, E')$, in the separated intersection query (SIQ) data structure of Lemma 2.5.

Separated Intersection Query II (SIQ II). Consider for each $C' \in T(\mathcal{S})$, all $i \in [1, \lceil \log \Psi \rceil]$. Consider the at most $O(\Psi^2)$ cells C in the *complete quadtree* $\mathcal{T}_B$ with $|C'| = 2^{i-1}|C|$ such that C' intersects $(2^{i+5}+1) * C$. We maintain the data structure $P(E, E')$ which stores all disks in $\pi_\varepsilon(E)$, minus the disks in $M(E, E')$, in the SIQ data structure of Lemma 2.5.

We can dynamically maintain the augmented data structure and, with it, maintain G.

▶ **Theorem 4.1.** *Let $\mathcal{S}$ be a dynamic set of n disks where at all times, their radii lie in $[4, \Psi]$ and all centres are placed in a fixed bounding box of size $2^{(\Psi^2)} \times 2^{(\Psi^2)}$. We can maintain G in $O(\Psi^2 \varepsilon^{-2} \log^4 n \log^2 \Psi \log^2(1/\varepsilon))$ expected amortised update time and using $O(\Psi^2 \varepsilon^{-2} n \log n \log^2 \Psi)$ total space.*

Proof sketch. When a disk σ is inserted, G is maintained by figuring out whether σ should be inserted into some MBM. This is done by querying SIQ II and SIQ I (in this order). After fixing the matchings and the SIQ data structures, G is updated. First, we update all relevant Type i edges using the "local" Euclidean spanners. Then edges of Type ii are updated using the newly updated matchings. Deletion is done very similarly to insertion with the notable difference that if the disk that is being removed was part of a matching, the other end of the edge needs to be rematched or inserted into one of the SIQ data structures. ◀

References

1 Alexander Baumann, Haim Kaplan, Katharina Klost, Kristin Knorr, Wolfgang Mulzer, Liam Roditty, and Paul Seiferth. Dynamic connectivity in disk graphs.

Discrete & Computational Geometry (DCG), 71(1):214–277, 2024. doi:10.1007/s00454-023-00621-x.

2 Timothy M Chan, Sariel Har-Peled, and Mitchell Jones. On locality-sensitive orderings and their applications. *SIAM Journal on Computing (SICOMP)*, 49(3):583–600, 2020. doi:10.1137/19M1246493.

3 Timothy M. Chan and Zhengcheng Huang. Dynamic geometric connectivity in the plane with constant query time. In Wolfgang Mulzer and Jeff M. Phillips, editors, *Symposium on Computational Geometry (SoCG)*, volume 293 of *LIPIcs*, pages 36:1–36:13. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2024. doi:10.4230/LIPICS.SOCG.2024.36.

4 Timothy M. Chan, Mihai Pătraşcu, and Liam Roditty. Dynamic connectivity: Connecting to networks and geometry. *SIAM J. Comput.*, 40(2):333–349, 2011. doi:10.1137/090751670.

5 Brent N Clark, Charles J Colbourn, and David S Johnson. Unit disk graphs. *Discrete mathematics*, 86(1-3):165–177, 1990. doi:10.1016/0012-365X(90)90358-O.

6 Aleksei V Fishkin. Disk graphs: A short survey. In *International Workshop on Approximation and Online Algorithms*, pages 260–264. Springer, 2003. doi:10.1007/978-3-540-24592-6_23.

7 Martin Fürer and Shiva Prasad Kasiviswanathan. Spanners for geometric intersection graphs with applications. *Journal of Computational Geometry (JoCG)*, 3(1):31–64, 2012. doi:10.20382/JOCG.V3I1A3.

8 Haim Kaplan, Wolfgang Mulzer, Liam Roditty, Paul Seiferth, and Micha Sharir. Dynamic planar voronoi diagrams for general distance functions and their algorithmic applications. *Discrete and Computational Geometry (DGC)*, 64(3):838–904, 2020. doi:10.1007/S00454-020-00243-7.

9 Terry A McKee and Fred R McMorris. *Topics in intersection graph theory*. SIAM, 1999. doi:10.1137/1.9780898719802.

10 Giri Narasimhan and Michiel Smid. *Geometric spanner networks*. Cambridge University Press, 2007. doi:10.1017/CBO9780511546884.

11 David Peleg and Alejandro A Schäffer. Graph spanners. *Journal of graph theory*, 13(1):99–116, 1989. doi:10.1002/jgt.3190130114.

Computing L_∞ Hausdorff Distances Under Translations: The Interplay of Dimensionality, Symmetry and Discreteness

Sebastian Angrick[1], Kevin Buchin[2], Geri Gokaj[1], and Marvin Künnemann[1]

1 **Karlsruhe Institute of Technology, Germany**
`firstname.lastname@kit.edu`
2 **TU Dortmund, Germany**
`firstname.lastname@tu-dortmund.de`

Abstract

We study the fine-grained complexity of the L_∞ *Hausdorff distance under translation* problem. Given two sets P and Q of n and m points, respectively, in $\mathbb{R}^d$, we ask: How is the running time influenced by the dimension d, the relationship between n and m, and whether we use the directed or undirected Hausdorff distance? We obtain, among other results, the following findings:

- The directed Hausdorff distance has an *intrinsically asymmetric* time complexity: While the $\tilde{O}((nm)^{d/2})$ combinatorial lower bound for $d \geq 3$ of (Chan, SoCG'23) is conditionally tight when $m \leq n$, we show that this is not the case for $n \ll m$ by providing a combinatorial, almost-linear-time algorithm for $d = 3$ and $n = m^{o(1)}$.
- We further prove novel, *general*, i.e., non-combinatorial, conditional lower bounds, in particular: (1) $m^{\lfloor d/2 \rfloor - o(1)}$ for $d \geq 3$ and small n and (2) $n^{d/2-o(1)}$ for $d = 3$ and small m.
- We observe that the directed and undirected case is closely related, in particular, most lower bounds for $d \geq 3$ hold for both the directed and undirected variant. A remarkable exception is the case of $d = 1$, for which we provide a conditional separation.

Related Version arXiv:2603.08890

1 Introduction

Consider the classic Hausdorff distance of given point sets P and Q, which comes in two flavors: The undirected (symmetric) as well as the directed (asymmetric) variant:

$$\delta_{\overrightarrow{H}}(P,Q) \coloneqq \max_{p\in P} \min_{q\in Q} \|p-q\|_\infty \qquad \delta_H(P,Q) \coloneqq \max\{\delta_{\overrightarrow{H}}(P,Q), \delta_{\overrightarrow{H}}(Q,P)\}.$$

Here and throughout the paper, we fix our metric space to be $(\mathbb{R}^d, L_\infty)$. For any constant $d \in \mathbb{N}$ and point sets P and Q of n and m points in $\mathbb{R}^d$, we can compute these distances in near-linear time[1] $\tilde{O}(n+m)$ using L_∞-nearest neighbor search, e.g., in a range tree [8].

A computationally more challenging task is to compute the Hausdorff distance under some transformation, i.e., to minimize the Hausdorff distance of P' and Q among all sets P' obtained from P using a set of admissible transformations. We study the *decision variant* which receives a test distance $\delta > 0$ as additional input[2]. Specifically, we study:

[1] Throughout the paper, we use $\tilde{O}(\cdot)$, to hide polylogarithmic terms in n and m. Since we assume d to be constant, this includes factors of the form $\log^{O(d)} nm$.

[2] Note that the case $\delta = 0$, i.e., finding an exact match, allows an almost-linear-time algorithm [7] and is thus uninteresting for our purposes.

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

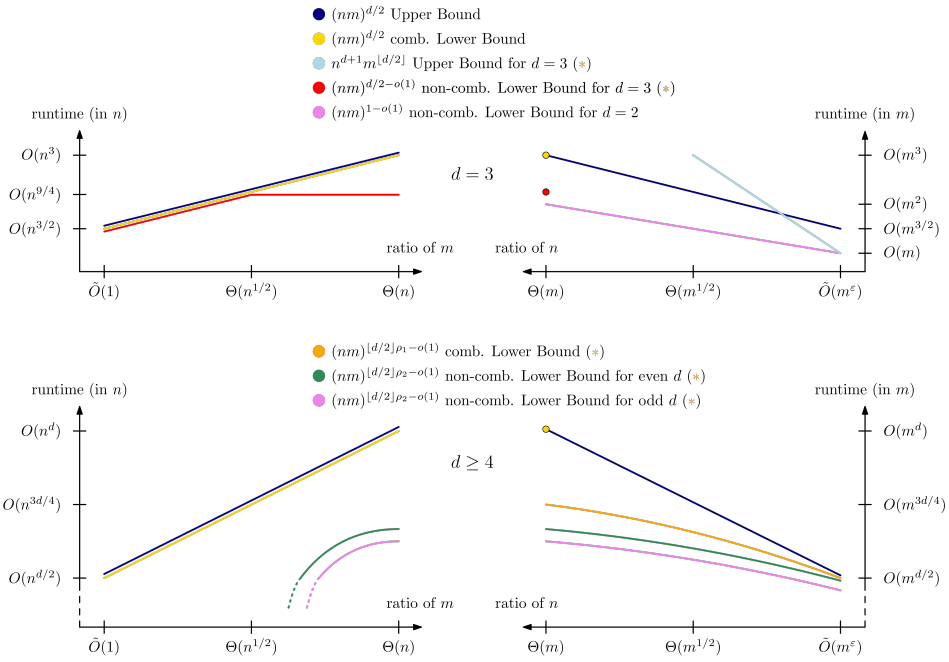


Figure 1 Schematic figure of upper and lower bounds for directed Hausdorff under Translation in 3-D (upper figure) and d-D for $d \geq 4$ (lower figure), novel results are marked with (∗) . ρ_1, ρ_2 are functions on the ratio of n, m which correspond to values described in the full version of the paper.

▶ **Definition 1.1** (Hausdorff under Translation (*HuT*)). Given sets $P, Q \subset \mathbb{R}^d$ of n and m points, respectively, and a real value $\delta > 0$, decide whether:

$$\exists \tau \in \mathbb{R}^d : \delta_{\overrightarrow{H}}(P + \tau, Q) \leq \delta \quad \equiv \quad \exists \tau \in \mathbb{R}^d \; \forall p \in P \; \exists q \in Q : \|(p + \tau) - q\|_\infty \leq \delta$$

To obtain the *undirected* versions of the above problems, we replace $\delta_{\overrightarrow{H}}(P + \tau, Q)$ by $\delta_H(P + \tau, Q)$, and denote the problem by *uHuT*. If not specified otherwise, we consider the directed versions. We call the case in which all input sets have roughly equal size, i.e., $Q = \Theta(|P|)$ the *balanced* parameter setting, otherwise we say the parameters are *lopsided*.

We may additionally limit the set of feasible translations T to be a set of at most $t \in \mathbb{N}$ possible translations. This variant is called the Discrete Hausdorff under Translation problem.

We give an overview of our results in Figure 1 and Table 1.

2 Preliminaries

A brief definition of relevant problems as well as assumed hardness hypotheses can be found in the full version.

For clarity of presentation, it is often useful to view the Hausdorff under Translation decision problem as a translation problem involving certain *shapes*.

▶ **Definition 2.1** (Translation Problem). Given a set of n points P and m shapes $\mathcal{Q}$ in $\mathbb{R}^d$, compute a translation $\tau \in T \subseteq \mathbb{R}^d$ such that

$$\forall p \in P \; \exists Q \in \mathcal{Q} : p + \tau \in Q.$$

$d=1$	Directed	Undirected
Continuous	$\tilde{O}(n^2)$ ([6]) (*) $n^{2-o(1)}$ ($)†	$\tilde{O}(n)$ ([9]) $\Omega(n)$
Discrete	$\tilde{O}(n^2)$ ($) (*) $n^{2-o(1)}$ ($)†	$\tilde{O}(n)$ ($) $\Omega(n)$

$d=2$	Directed	Undirected
Continuous	$\tilde{O}(n^2)$ ([6]) $n^{2-o(1)}$ ([3])	$\tilde{O}(n^2)$ ([6], $) $n^{2-o(1)}$ ([3])
Discrete	$\tilde{O}(n^2)$ ($) (*) $n^{2-o(1)}$ ($)†	$\tilde{O}(n^2)$ ($) (*) $n^{2-o(1)}$ ($)†

Table 1 An overview of known and novel results of the L_∞-Hausdorff distance under Translation in 1-D and 2-D for the balanced case, i.e., $n = m$ ($= t$ in the discrete setting). Upper Bounds are marked in blue, lower bounds in green, while results under non-standard hypothesis are additionally marked with (†). Novel results are denoted with (*) . Results referring to the full version are marked with $.

For an instance of HuT with point sets P, Q and $\delta > 0$, if we construct hypercubes $\mathcal{Q}$ with side length 2δ and centers $q \in Q$, the region $T^* = \bigcap_{p\in P} \bigcup_{\hat{Q}\in\mathcal{Q}} \hat{Q} - p$ is exactly the region of feasible translations, i.e., we have $\tau \in T^* \iff \forall p \in P\ \exists q \in Q : \delta_{\overrightarrow{H}}(P + \tau, Q) \leq \delta$ [1]. We call this case, where all shapes $\mathcal{Q}$ are hypercubes of the same size, the *Translation Problem with Hypercubes (TPwC)*. In case all shapes of $\mathcal{Q}$ are axis-aligned boxes (of arbitrary size), we call it the *Translation Problem with Boxes (TPwB)*. While the TPwC problem is equivalent to HuT, there is no reduction to HuT known within the same ambient space from the Translation Problem with Boxes or even from the translation problem with hypercubes of different sizes. We further use a third variant.

▶ **Definition 2.2** (Translation Problem with Orthants (TPwO))**.** Let B be a target box with a maximum side length δ_0. Given a set of k Translation Problem with Hypercube instances, each on a point set P_i and hypercubes $\mathcal{Q}_i$ with side length $\delta > \delta_0$, such that $\sum_{i\in[k]} |P_i| = n$, $\sum_{i\in[k]} |\mathcal{Q}_i| = m$, compute a translation $\tau \in \mathbb{R}^d$ such that $\forall i \in [k] : P_i + \tau \subset B$ and

$$\forall i \in [k]\ \forall p_i \in P_i\ \exists Q_i \in \mathcal{Q}_i : p_i + \tau \in Q_i.$$

We call the shapes of the TPwO instance, i.e., the shapes of its sub-instances, orthants as within the target box all hypercubes are indistinguishable from orthants. An important property of this problem, which we will use later, is that we may fix the side-length of all orthants to a value $\delta > \delta_0$. Similar to [4, Lemma 3], the TPwO problem reduces to the TPwC and thus has a reduction to HuT. Its proof is deferred to the full version. A converse reduction, from HuT to TPwO, is generally not known.

▶ **Lemma 2.3.** *The Translation Problem with Orthants on n points and m orthants reduces in linear time to the Translation Problem with Hypercubes on $\Theta(n)$ points and $\Theta(m)$ hypercubes.*

3 Technical Overview for Selected Results

In this section, we discuss the main ideas of our results. We present the general structure of the proofs and give sketches on the most important technical details. Because of spatial limitations, we defer further details as well as additional results such as

- the tight non-combinatorial lower bound in 3-D for $m = O(n^{1/2})$,
- the conditional separation of the directed and undirected variant in 1-D,

- all results related to the discrete HuT problem

to the full version.

3.1 (Non-)Combinatorial Lower Bounds for Arbitrary n, m

For our lower bounds in higher dimensions ($d \geq 4$), we give a reduction in three steps from the k-hyperclique problem to HuT, a formal definition and its hardness hypothesis can be found in the appendix of the full version.

In the first two steps, we reduce the k-hyperclique problem to the Translation Problem with Boxes. We generally follow the approach of Chan [4], who gave a reduction from k-clique to TPwC, by encoding hypercliques as an intersection of boxes inside a d-dimensional hypercube. Most importantly, our reduction allows near-arbitrary distributions of the parameters in the Translation Problem with Boxes, although our lower bounds deteriorate for growing n. We defer the first two proof steps and further details to the full version.

▶ **Theorem 3.1.** *Let $\lambda \in (0, 1], d \in \mathbb{N}_{\geq 2}$ be constant. If we can solve the Translation Problem with n points and m boxes in d-D in Time $T_d(n, m)$, we can solve the 3-uniform k-hyperclique problem on N vertices, for a sufficiently large but constant k, in time $T_d(N^{3\lambda k/d}, N^{(1-\lambda)k/d})$.*

However, there is no direct reduction from the Translation Problem with Boxes to HuT. In the third step of our overall reduction, we thus prove a reduction from the Translation Problem with Boxes in d-D to the Translation Problem with Orthants in $2d$-D. Lemma 2.3 gives a subsequent reduction to directed Hausdorff under Translation.

Step 3: From Boxes to Orthants

We reduce the Translation Problem with Boxes to the Translation Problem with Orthants, allowing a subsequent reduction to HuT. We implement the natural idea to cover boxes of arbitrary aspect ratios by orthants (large hypercubes), at the cost of doubling the underlying dimension. The basic idea of our translation gadget is to scale and rotate each dimension i by 45°, introducing a second dimension each. Any point is scaled and rotated accordingly. Each box, represented in a single dimension as an interval $[a, b]$, is split into two intervals $[a, \infty)$ and $(-\infty, b]$, which describes an orthant. For the proof, we see that a translation which stays on the newly created diagonal is equivalently a solution to the Translation Problem with Boxes as well as to the Translation Problem with Orthants. To make this transformation usable for our purposes, we need 2^d (mirrored) copies of the constructed orthants to constrict all possible translations to the desired diagonal. Figure 2 gives a visual intuition of the above description, the full proof is described in the full version.

▶ **Lemma 3.2.** *The Translation Problem with Boxes in d-D with input parameters n, m reduces in linear time to the Translation Problem with Orthants in $2d$-D with input parameters $n' = \Theta(n), m' = \Theta(m)$.*

Combining our results of Theorem 3.1 and Lemmas 3.2, 2.3, we get lower bounds for arbitrary distributions of n, m. However, as the bounds deteriorate with growing values of n, we only get tight lower bounds for $n = m^{o(1)}$. Other cases as well as the theorem's proof can be found in the full version.

▶ **Theorem 3.3.** *Let $\varepsilon > 0, d \in \mathbb{N}_{\geq 4}$ be constant. For $n = m^{o(1)}$, there is no $\tilde{O}((nm)^{\lfloor \frac{d}{2} \rfloor - \varepsilon})$ algorithm for directed Hausdorff under Translation in d-D under the k-hyperclique hypothesis.*

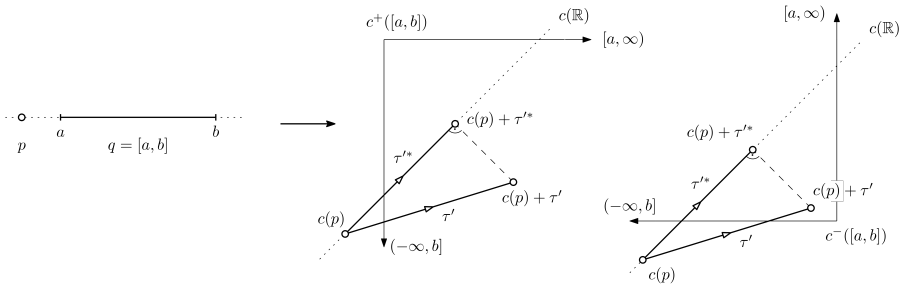


Figure 2 The reduction of a box in 1-D, i.e., an interval $[a, b]$, to orthants in 2-D. We prove that for each feasible translation τ' there exists a feasible translation τ'^* which lies on the diagonal $c(\mathbb{R})$.

3.2 A Faster Algorithm for $m \gg n$

For $n = m^{o(1)}$, Theorem 3.3 gives a $(nm)^{\lfloor d/2 \rfloor - o(1)}$ lower bound, which for even d matches the best known algorithm running in time $O((nm)^{d/2})$ [4], while for odd d it seems to have some slack. Intuitively, one might expect that the known $(nm)^{d/2-o(1)}$ combinatorial lower bound for $m = O(n)$ [4] also transfers to the case of $m \gg n$. Surprisingly, in 3-D the $(nm)^{1-o(1)}$ lower bound of [3] is tight for $n = m^{o(1)}$ as we give an $O(m^{1+o(1)})$ algorithm here.

The algorithm uses a structural insight from [5, Theorem 7]. Given a directed HuT instance with points P and hypercubes $\mathcal{Q}$ in d-dimensions, they already gave the combinatorial proof that the region of feasible translations, i.e., the region of translation vectors for which all points P are translated into the hypercubes of $\mathcal{Q}$, has $O(m^{\lfloor d/2 \rfloor} n^d)$ many vertices. However, their proof was not algorithmic yet, as it needed the explicit construction (of the vertices) of the union of $O(m)$ congruent hypercubes in time $O(m^{\lfloor d/2 \rfloor})$.

Such an algorithm for unit cubes in 3-D was recently introduced by Agarwal and Steiger [2], which enables our algorithm. Assuming an algorithm that computes the vertices of the union of m unit hypercubes in d-D in time $\tilde{O}(m^{\lfloor d/2 \rfloor})$, our algorithm directly generalizes to higher dimensions with a runtime of $\tilde{O}(m^{\lfloor d/2 \rfloor} n^{d+1})$. The full proof is deferred to the full version.

▶ **Theorem 3.4.** *Directed Hausdorff under Translation in 3-D can be solved in time* $\tilde{O}(n^4 m)$.

This result points to a discrepancy in complexity, at least for $d = 3$ but potentially also for all odd d. While we have a $O(m^{\lfloor d/2 \rfloor + o(1)})$ algorithm for $n = m^{o(1)}$, there is a $(nm)^{d/2-o(1)}$ tight (combinatorial) lower bound for $m \in O(n)$ [4]. It remains unknown where the cutoff point is located, i.e., for which values of n, m do we get faster than $(nm)^{d/2 \pm o(1)}$ algorithms?

3.3 Transferring Lower Bounds to the Undirected Setting

As of now, all here presented lower bounds only apply to the directed Hausdorff under Translation setting. Chan [4] already raised the question if these bounds transfer to the undirected setting, however there are no such lower bounds known so far for $d \geq 3$. Only for $d = 2$, lower bounds for the undirected setting are known [3], which match the bounds for the directed variant. For $d = 1$, we have an optimal $\tilde{O}(n + m)$ algorithm [9].

We partially resolve this question by providing a reduction from the Translation Problem with Orthants to undirected Hausdorff under Translation, maintaining all instance parameters. The main idea is that orthants are sufficiently large to strategically place $O(3^d)$ many auxiliary points such that each orthant is hit by at least one such point under any translation.

The proof is given in the full version.

▶ **Theorem 3.5.** *If we can decide the undirected Hausdorff under Translation problem in d-D with point sets of size n, m in time $T(n, m)$, we can decide the Translation Problem with Orthants in d-D with n points and m orthants in time $T(n, m) + O(n + m)$.*

All here discussed lower bounds for HuT in $d \geq 3$ dimensions, as well as the combinatorial $(nm)^{d/2-o(1)}$ lower bound for $m \in O(n)$ of Chan [4], can be directly applied to undirected Hausdorff under Translation as they use the Translation Problem with Orthants as an intermediate problem. At the cost of doubling the dimension using Lemma 3.2, we generally reduce any instance of directed Hausdorff under Translation to the undirected setting. We get the following excerpt of results, more results can be found in the full version.

▶ **Corollary 3.6.** *We get the following results:*

- *For arbitrary n, m, there is no $O((nm)^{d/2-\varepsilon})$ combinatorial algorithm for undirected Hausdorff under Translation in d-D under the k-clique hypothesis.*
- *For $\min(n, m) \in O(\max(n, m)^{1/2})$, there is no $O((nm)^{3/2-\varepsilon})$ algorithm for undirected Hausdorff under Translation in 3-D under the k-hyperclique hypothesis.*

References

1 P. K. Agarwal, M. Sharir, and S. Toledo. Applications of parametric searching in geometric optimization. *J. Algorithms*, 17(3):292–318, 1994. doi:10.1006/jagm.1994.1038.

2 Pankaj K. Agarwal and Alex Steiger. An output-sensitive algorithm for computing the union of cubes and fat boxes in 3D. *Proc. 48th Internat. Colloq. Automata Lang. Program. (ICALP)*, 198:10:1–10:20, 2021. doi:10.4230/LIPICS.ICALP.2021.10.

3 Karl Bringmann and André Nusser. Translating Hausdorff is hard: fine-grained lower bounds for Hausdorff distance under translation. *J. Comput. Geom.*, 13(2):30–50, 2021. doi:10.20382/JOCG.V13I2A3.

4 Timothy M. Chan. Minimum L_∞ Hausdorff distance of point sets under translation: Generalizing Klee's measure problem. In *Proc. 39th Internat. Sympos. Comput. Geom. (SoCG)*, volume 258, pages 24:1–24:13. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023. doi:10.4230/LIPICS.SOCG.2023.24.

5 L. P. Chew, D. Dor, A. Efrat, and K. Kedem. Geometric pattern matching in d-dimensional space. *Discrete Comput. Geom.*, 21(2):257–274, 1999. doi:10.1007/PL00009420.

6 L. Paul Chew and Klara Kedem. Getting around a lower bound for the minimum Hausdorff distance. *Comput. Geom. Theory Appl.*, 10(3):197–202, June 1998. doi:10.1016/S0925-7721(97)00032-1.

7 Nick Fischer. Deterministic sparse pattern matching via the Baur-Strassen theorem. In *Proc. 35th Annu. ACM-SIAM Sympos. Discrete Algorithms (SODA)*, pages 3333–3353. SIAM, 2024. doi:10.1137/1.9781611977912.119.

8 Sanjiv Kapoor and Michiel Smid. New techniques for exact and approximate dynamic closest-point problems. *SIAM Journal on Computing*, 25(4):775, 1996.

9 Günter Rote. Computing the minimum Hausdorff distance between two point sets on a line under translation. *Information Processing Letters*, 38(3):123–127, 1991. doi:10.1016/0020-0190(91)90233-8.

On Small Pair Decompositions for Point Sets: Hardness and the 1D case

Kevin Buchin[1], Jacobus Conradi[2,3], Sariel Har-Peled[4], Antonia Kalb[1], Abhiruk Lahiri[*] [5], Lukas Plätz[6], Carolin Rehs[†] [1,7], and Sampson Wong[3]

1 **Department of Computer Science, TU Dortmund University, Germany**
2 **Institute of Computer Science, Universität Bonn, Germany**
3 **Department of Computer Science, University of Copenhagen, Denmark**
4 **Department of Computer Science, University of Illinois, USA**
5 **Heinrich Heine University, Düsseldorf and CISPA Helmholtz Centre for Information Security, Saarbrücken, Germany**
6 **Faculty of Computer Science, Ruhr University Bochum, Germany**
7 **Department of Mathematics and Computer Science, Eindhoven University of Technology, The Netherlands**

Abstract

We study the problem of computing for a point set P the well-separated pair decomposition with the minimum number of pairs. In this extended abstract, we present approximation algorithms for one-dimensional point sets, show that the problem is NP-hard for points in the plane, and also prove the hardness of approximation in general.

1 Introduction

Given a set P of n points in a metric space, the *Well-Separated Pair Decomposition* (WSPD), introduced by Callahan and Kosaraju [4], is an approximate compact representation of the metric induced by P. It has been used to solve numerous problems, such as the all-nearest-neighbors [4], approximate MST [4], spanners [14], Delaunay triangulations [3], distance oracles [6], and others [9]. See Smid [16] for a survey.

For a prespecified $\varepsilon \in (0,1)$, the $\frac{1}{\varepsilon}$-WSPD represents the $\binom{n}{2}$ pairwise distances of a set $P \subseteq \mathbb{R}^d$ of n points, as a union of $O(n/\varepsilon^d)$ (edge-disjoint) bicliques, where all the distances in a single biclique are the same up to a multiplicative factor in $1 \pm O(\varepsilon)$. More precisely, we require that $\max\big(\nabla(X), \nabla(Y)\big) \leq \varepsilon \boldsymbol{d}(X,Y)$ for every pair $\{X,Y\}$ in a $\frac{1}{\varepsilon}$-WSPD (where $\nabla(\cdot)$ denotes the diameter of a set and $\boldsymbol{d}(\cdot,\cdot)$ the minimum distance between two sets). This biclique cover can be computed in $O(n \log n + n/\varepsilon^d)$ time, where each biclique is represented as a pair of nodes in the tree[1] [7, Ch. 3]. The *size* of the WSPD is the number of bicliques.

This raises a natural question: how do we construct a small (or even the smallest) representation of the pairwise distances in P?

[*] Partially supported by the Strategic Research Fund, Heinrich Heine University, Düsseldorf. The author would also like to thank the ALGO lab for providing one of Germany's cleanest coffee machines to stimulate research.
[†] Supported by the PRIME programme of the German Academic Exchange Service (DAAD) with funds from the German Federal Ministry of Research, Technology and Space (BMFTR)

[1] In WSPD constructions, this refers to a hierarchical spatial decomposition data structure, such as a compressed quadtree or a fair split tree, where each node compactly represents a subset of points.

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

Minimum-size WSPDs. The bound on the size of a $\frac{1}{\varepsilon}$-WSPD in $\mathbb{R}^d$ is $O(n/\varepsilon^d)$ [4], which can be far from optimal. In $\mathbb{R}^2$, the smallest $\frac{1}{\varepsilon}$-WSPD for the point set $P = \{p_i = (i, (4/\varepsilon)^i) \mid i = 1, \ldots, n\}$ is formed by the pairs $X_i = \{p_1, \ldots, p_i\}$ and $Y_i = \{p_{i+1}\}$, for $i = 2, \ldots, n$. Its size $(n-1)$ is significantly smaller than $\Theta(n/\varepsilon^2)$. Computing small WSPDs may have downstream effects on its applications, such as the sizes of WSPD-based spanners or distance oracles. This motivates us to study the minWSPD problem.

▶ **Problem 1.1** (minWSPD). *Given a set P of n points in a metric space $(\mathcal{X}, d)$, and a parameter ε, compute the minimum-size $\frac{1}{\varepsilon}$-WSPD ($\mathcal{W}$) of P. That is, minimize the number of pairs in $\mathcal{W}$, while guaranteeing that the resulting WSPD provides the desired separation.*

Our contributions. We show that: (1) Computing the exact minimum-size $\frac{1}{\varepsilon}$-WSPD is NP-Hard in $\mathbb{R}^2$. (2) In 1D, we provide a PTAS and a practical 3-approximation algorithm with runtime $O(n \log n)$ for the minimum-size $\frac{1}{\varepsilon}$-WSPD. This assumes that the bicliques mentioned above are not required to be disjoint[2]. Dropping this assumption increases the approximation factor by a constant.

Comparison to previous work. Computing a WSPD with instance-specific guarantees has previously been considered by Har-Peled *et al.* [10]. Our result differs in that our $\frac{1}{\varepsilon}$-WSPD is a constant-factor approximation compared to the minimum $\frac{1}{\varepsilon}$-WSPD, whereas the $\frac{1}{\varepsilon}$-WSPD in [10] only compares to the minimum $\frac{33}{\varepsilon}$-WSPD.

2 Preliminaries

Metrics, distances and well-separated A ***metric space*** $\mathcal{X}$ is a pair $(\mathcal{X}, d)$, where $\mathcal{X}$ is the ground set and $d : \mathcal{X} \times \mathcal{X} \to [0, \infty)$ is a ***metric***. For $P \subseteq \mathcal{X}$, let $\nabla(P)$ be its ***diameter***. We denote the ***distance*** between sets $X, Y \subseteq \mathcal{X}$ as $d(X, Y) = \min_{p \in X, q \in Y} d(p, q)$. For two sets $X, Y \subseteq \mathcal{X}$, let $X \otimes Y = \{\{x, y\} \mid x \in X, y \in Y, x \neq y\}$. A pair of sets $\{X, Y\}$ of points in a metric space $(\mathcal{X}, d)$ is ***$\frac{1}{\varepsilon}$-separated*** if $\max(\nabla(X), \nabla(Y)) \leq \varepsilon d(X, Y)$.

Pair decompositions. For a point set $P \subseteq \mathcal{X}$, a ***pair decomposition*** of P is a set of pairs $\mathcal{W} = \{\{X_1, Y_1\}, \ldots, \{X_s, Y_s\}\}$, such that (I) $X_i, Y_i \subset P$ for every i, (II) $X_i \cap Y_i = \emptyset$ for every i, and (III) $\bigcup_{i=1}^{s} X_i \otimes Y_i = P \otimes P$. For a point set P, a ***well-separated pair decomposition*** of P with parameter $1/\varepsilon$, denoted by ***$\frac{1}{\varepsilon}$-WSPD***, is a set of pairs $\mathcal{W} = \{\{X_1, Y_1\}, \ldots, \{X_s, Y_s\}\}$ such that $X_i, Y_i \subset P$, $X_i \cap Y_i = \emptyset$, their union covers $P \otimes P$, and each pair is separated such that $\max(\nabla(X_i), \nabla(Y_i)) \leq \varepsilon d(X_i, Y_i)$. The ***size*** of the WSPD $\mathcal{W}$ is the number of pairs in the decomposition. The WSPD $\mathcal{W}$ is a PartitionWSPD if every pair of $P \otimes P$ is covered exactly once by the pairs of $\mathcal{W}$. Otherwise, it is a CoverWSPD.

Constructing WSPDs via SetCover. Given a set P of n points, one possible approach is to compute a set $\mathcal{F}$ of all maximal well-separated pairs in $P \otimes P$. A pair $p = \{X, Y\} \in \mathcal{F}$ covers all the pairs of points in $E_p = X \otimes Y$. Thus, the problem of finding a minWSPD is equivalent to solving the minimum SetCover problem for the instance $(P \otimes P, \{E_p \mid p \in \mathcal{F}\})$.

We can compute greedily a $\Theta(\log |\mathcal{F}|)$-approximation (in polynomial time) by adding at each step the pair in $\mathcal{F}$ that covers the maximum number of uncovered point pairs in Q [15]. It is known that SetCover cannot be efficiently approximated within a factor of $(1 - o(1)) \ln n$,

[2] We call such a $\frac{1}{\varepsilon}$-WSPD CoverWSPD, see Section 2 for definitions.

unless P = NP [12]. If one can limit the size of $\mathcal{F}$ to be polynomial in n, this readily leads to an $O(\log n)$-approximation of minWSPD.

Since $|\mathcal{F}| = O(n^2)$ for $d = 1$ (see Section 4 for details), greedy SetCover yields an $O(\log n)$-approximation for computing a minimum $\frac{1}{\varepsilon}$-WSPD for a point set on the real line for any $\varepsilon > 0$. For higher dimensions, no polynomial bound on $|\mathcal{F}|$ is known.

Naturally, one can restrict the sets used to build the pairs to a certain (smaller) family of sets. For example, in two dimensions, consider all the subsets of P of the form $P \cap \square$ where $\square$ is some square. Then, one can solve/approximate the resulting SetCover problem. However, even in two dimensions and squares, the resulting discrete SetCover problem corresponds to a set of points in four dimensions, which must be covered using a set of axis-aligned boxes. To the best of our knowledge, no better than $O(\log n)$-approximation is known in this case.

3 Computational Hardness of the Minimum-Size WSPD

In this section, we prove that computing the minWSPD is NP-hard in the plane, and NP-hard to approximate in general metric spaces within a factor of n^{δ}, for sufficiently small $\delta \in (0, 1)$.

Arbitrary metrics hardness We show the hardness by reduction from ECliqueCover. In ECliqueCover, an instance is $(\mathcal{G}, k)$, where $\mathcal{G}$ is a graph, and one has to decide if one can cover the edges of $\mathcal{G}$ by k cliques that are contained in $\mathcal{G}$. (The CliqueCover problem asks for a disjoint cover of the vertices of $\mathcal{G}$ by cliques.) It is known that not only is the problem NP-Hard, but it is NP-Hard to approximate within a factor of $n^{1-\delta}$, for some $\delta \in (0, 1)$, see [11]. Essentially, either version of the clique cover is as hard as the minimum-cardinality vertex coloring problem.

Note that the following hardness results apply to both the CoverWSPD and PartitionWSPD variants.

▶ **Theorem 3.1.** *Computing the (exact) minimum size 3-WSPD in a metric space is NP-Hard.*

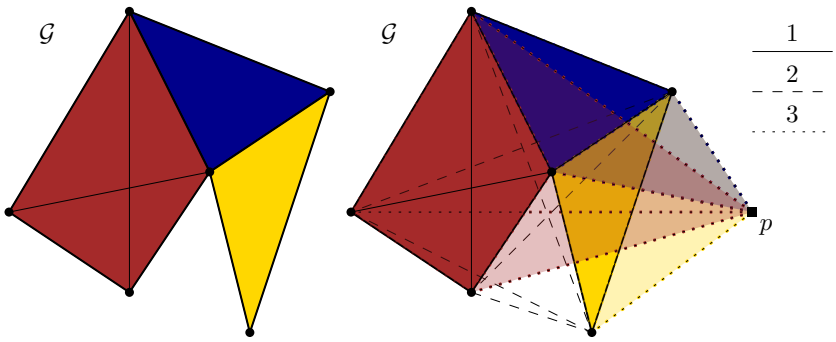


Figure 1 Reduction to ECliqueCover. The graph $\mathcal{G}$ with circles as nodes. Solid edges have a distance of 1; dashed, 2; and dotted, 3. Every solid or dashed edge needs a pair. Every clique C_i (solid colors) allows us to cover the dotted edges with the pair $\{\{p\}, C_i\}$ (polyhedra).

Proof. We reduce ECliqueCover to the problem of finding a minimum 3-WSPD. Given an instance $(\mathcal{G}, k)$ with n vertices, construct a metric space on $\mathcal{X} = \mathsf{V}(\mathcal{G}) \cup \{p\}$. Define distances as: $d(u, v) = 1$ if $(u, v) \in \mathsf{E}(\mathcal{G})$, else 2; and $d(p, v) = 3$ for all $v \in \mathsf{V}(\mathcal{G})$.

If $\mathcal{G}$ has a clique cover $\{C_1, \ldots, C_k\}$, construct

$$\mathcal{W} = \left\{ \{\{p\}, C_i\} \;\middle|\; i = 1, \ldots, k \right\} \cup \left\{ \{\{u\}, \{v\}\} \;\middle|\; u, v \in \mathsf{V}(\mathcal{G}), u \neq v \right\}.$$

Since $\nabla(C_i) = 1$ and $d(\{p\}, C_i) = 3$, the pairs $\{\{p\}, C_i\}$ are 3-separated. The singleton pairs cover all internal edges of $\mathcal{G}$. The total size is $\binom{n}{2} + k$.

Conversely, let $\mathcal{W}$ be a 3-WSPD of $\mathcal{X}$. For any distinct $u, v \in \mathsf{V}(\mathcal{G})$, separation requires the covering sets to have diameter $< d(u, v)/3 \leq 2/3$. Thus, all pairs in $\mathsf{V}(\mathcal{G}) \times \mathsf{V}(\mathcal{G})$ must be covered by singleton pairs $\{\{u\}, \{v\}\}$, contributing exactly $\binom{n}{2}$ to the size. Remaining pairs must cover $\{p\} \times \mathsf{V}(\mathcal{G})$ and take the form $\{\{p\}, \mathsf{V}_i\}$. To be 3-separated, we require $3\nabla(\mathsf{V}_i) \leq d(\{p\}, \mathsf{V}_i) \leq 3$. This forces $\nabla(\mathsf{V}_i) \leq 1$, meaning each V_i induces a clique in $\mathcal{G}$.

Thus, $\mathcal{X}$ has a 3-WSPD of size $\binom{n}{2} + k \iff \mathcal{G}$ has a clique cover of size k. ◀

▶ **Lemma 3.2** (Proof in the Full Version [2]). *For a finite metric space $\mathcal{X}$ defined over a set of n points, for $\delta \in (0, 1)$ sufficiently small, there is no n^δ-approximation to the size of the optimal 3-WSPD of $\mathcal{X}$, unless $P = NP$.*

It is also hard in $\mathbb{R}^2$. More details on this can be found in the full version [2].

▶ **Theorem 3.3** (Proof in the Full Version [2]). *Computing the minimum size WSPD is NP-Hard for point sets in the Euclidean plane. Specifically, given a point set Q in the plane, and parameters $t \in \mathbb{N}$ and $\varepsilon \in (0, 1)$, deciding if Q has a $\frac{1}{\varepsilon}$-WSPD of size at most t is NP-Hard.*

4 The One-dimensional Problem

Approximating minWSPD via SetCover. Let $P = \{p_1, \ldots, p_n\} \subset \mathbb{R}$ be a set of n points with $p_i < p_{i+1}$. We reduce CoverWSPD to a geometric SetCover instance. The universe is the set $Q = \{(p_i, p_j) \mid p_i, p_j \in P, i < j\}$, the grid points strictly above the diagonal $P \times P$. To each maximal well-separated pair corresponds an axis-aligned ***shadow square*** $\mathsf{S}_r := [r_x - \tau, r_x] \times [r_y, r_y + \tau]$ where $\tau = \varepsilon(r_y - r_x)$ in the plane, which geometrically represents the set of point pairs covered by this maximal well-separated pair anchored at a point $r = (r_x, r_y) \in \mathbb{R}^2$. This transforms minWSPD into covering Q with a minimal set of such squares. This admits a PTAS in $n^{O(1/\delta^2)}$ time using local search [5, 13], giving the following.

▶ **Lemma 4.1.** *Given $P \subset \mathbb{R}$ and $\varepsilon, \delta \in (0, 1)$, a $\frac{1}{\varepsilon}$-CoverWSPD of size at most $(1 + \delta) \cdot \mathrm{opt}$ is computable in $n^{O(1/\delta^2)}$ time.*

A 3-Approximation for CoverWSPD. We interpret Q as the set of all points above the diagonal in the non-uniform grid $P \times P$, and search for a set of squares that covers Q. The algorithm uses the "One for the Price of Two" approach of Bar-Yehuda [1]. We compute a set of three squares that provides at least as much coverage as a single square in the optimal solution.

The ***vicinity*** $V(p)$ of a point $p \in Q$ is the set of all points $p' \in Q$ such that $p, p' \in \mathsf{S}_r$ for some anchor $r \in \mathbb{R}^2$. See Figure 2 for an example. We can restrict the set of anchors to $r \in Q$, because if there exists a S_r with $p = (p_x, p_y)$ and $p' = (p'_x, p'_y)$ in S_r, then the point $r' := (\min(p_x, p'_x), \min(p_y, p'_y))$ is a valid anchor in Q such that $p, p' \in \mathsf{S}_{r'}$.

Now, consider the optimal solution $\mathcal{O} \subseteq \mathcal{F}$, and consider a single point $p \in Q$ not yet covered. The point p is covered by $\mathcal{O}$, and for all squares $\square \in \mathcal{O}$ that contain p, we have $\square \subseteq V(p)$ by definition. Our main idea is to sweep the points Q from right to left and cover the left half of the vicinity of a point with three squares (compare to Figure 2).

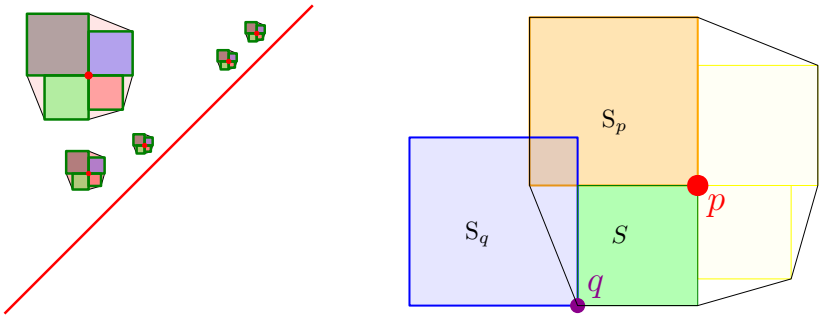


Figure 2 Left: A few points and their vicinities, for $\varepsilon = 0.4$. Right: The left half of the vicinity of a point p can be covered by three squares.

Algorithm. We use a sweep-line algorithm to process points from right to left. We maintain a segment tree to track the status of coverage in the current column. For each uncovered point p, we greedily add three specific squares that cover the left half of its vicinity. We use priority queues to manage interval insertion and deletion events as the sweep progresses. A visualization of the sweep and event status is shown in Figure 3.

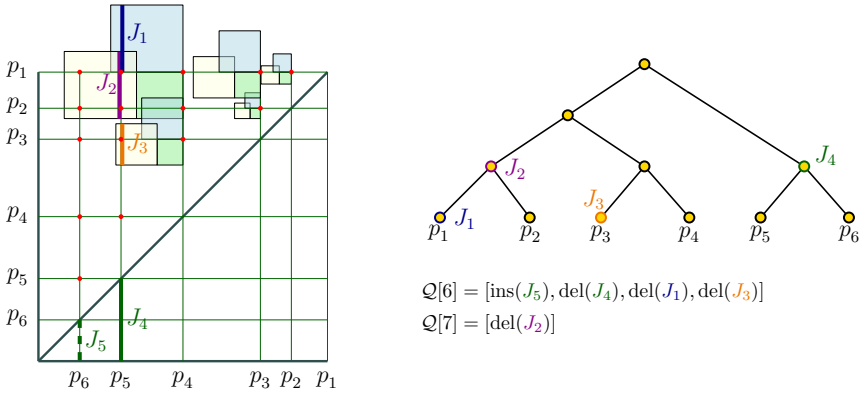


Figure 3 Visualization of the sweep and event status at the beginning of column p_5. A search would find p_4. When we place an interval over it on the backtrack, we check whether the node's children are covered; if so, we mark the node as covered. So, covering p_4 propagates to the root node, and the next search would terminate at the root.

Approximation factor. As we cover the intersection of the vicinity and the sweeping halfplane of a point $V(p)$, our solution covers the optimal square(s) that cover p. With this, we can be sure to never charge those squares again. By reversing this logic, we get that if a point is not covered, we also have never charged any optimal square(s) that cover it. Therefore, the approximation factor is 3, which is the minimum number of squares to cover the vicinity of a point in any sweep direction.

▶ **Lemma 4.2.** *For any $\varepsilon > 0$, there exists a halfplane such that only three squares are needed to cover the intersection of the vicinity of a point p and the halfplane.*

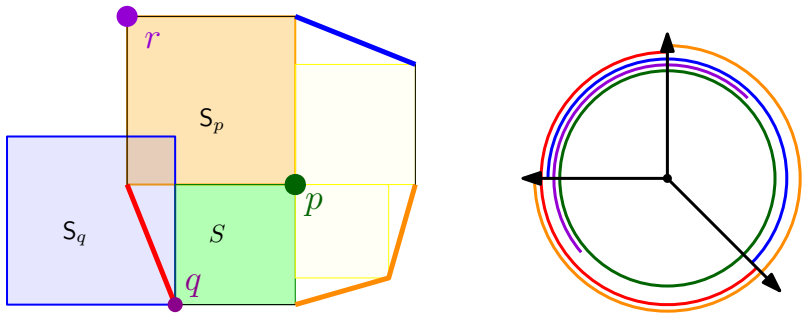


Figure 4 The five features in different colors and an angle plot with intervals show which direction includes which features. The three arrows highlight directions of the normal vector, which are the only ones with fewer than four features.

Proof. To account for every sweep's direction, consider all halfplanes through p. First, observe that one can cover the two left quadrants with three squares. This argument gives us the lower bound. We highlight five features also shown in Figure 4. The upper left point r, the neighborhood of p, the diagonals in the upper right and lower left quadrant, and the two diagonals (as a single feature) in the lower right quadrant. All halfplanes, except for three, contain an open neighborhood of at least four of them. Every square that contains such an open neighborhood cannot cover any other square. One direction is the solution; the other is a symmetrical version. The last one contains only two features. However, it contains the lower right quadrant, which can not be covered with three squares. So, we need at least three squares, which provide a tight upper bound. ◀

Finally, we analyze the running time.

▶ **Lemma 4.3.** *Given $P \subset \mathbb{R}$ and $\varepsilon \in (0,1)$, a CoverWSPD of size at most $3 \cdot \mathrm{opt}$ is computable in $O(n \log n)$ time.*

Proof. Assuming the optimal solution has size $O(n)$, the total runtime is $O(n \log n)$. Each square is computed in $O(1)$ time and triggers one insertion, deletion, and a point lookup. By augmenting the tree nodes with coverage flags, we support insertion, deletion, and locating the next uncovered point in $O(\log n)$ time. Summing these costs over $O(n)$ squares, along with the priority queue maintenance, yields the result. ◀

From a CoverWSPD to a PartitionWSPD. If one requires the separated pairs to be disjoint, we give a constant approximation.

▶ **Lemma 4.4** (Proof in the Full Version [2])**.** *The union of n axis-aligned squares $\mathcal{S}$ can be decomposed in polynomial time into a set $\mathcal{R}$ of at most $9n$ interior-disjoint rectangles, such that every $r \in \mathcal{R}$ is contained in some square of $\mathcal{S}$.*

▶ Remark. A similar result to Lemma 4.4 holds for any (convex) pseudo-disks, but the constant is somewhat worse. A direct proof follows by randomized incremental construction, drawing the objects from "back to front", see [8, Sec. 2].

Combining Lemma 4.1 and Lemma 4.4 implies the following result.

▶ **Lemma 4.5.** *Given $P \subset \mathbb{R}$ and $\varepsilon \in (0,1)$, a PartitionWSPD of size at most $10 \cdot \mathrm{opt}$ is computable in $n^{O(1)}$ time.*

5 Conclusion

We provided several results for approximating the minimum WSPD. Some open problems are: Is the minimum WSPD problem NP-Hard in $\mathbb{R}$? Is there a $(1+\varepsilon)$-approximation algorithm, or a simple $O(1)$-approximation algorithm, in $\mathbb{R}^d$, for $d \geq 2$?

References

1 Reuven Bar-Yehuda. One for the price of two: A unified approach for approximating covering problems. *Algorithmica*, 27(2):131–144, 2000. doi:10.1007/S004530010009.

2 Kevin Buchin, Jacobus Conradi, Sariel Har-Peled, Antonia Kalb, Abhiruk Lahiri, Lukas Plätz, and Carolin Rehs. On small pair decompositions for point sets. *CoRR*, abs/2601.22728, 2026. arXiv:2601.22728, doi:10.48550/ARXIV.2601.22728.

3 Kevin Buchin and Wolfgang Mulzer. Delaunay triangulations in $O(\text{sort}(n))$ time and more. *J. ACM*, 58(2):6:1–6:27, 2011. doi:10.1145/1944345.1944347.

4 P. B. Callahan and S. R. Kosaraju. A decomposition of multidimensional point sets with applications to k-nearest-neighbors and n-body potential fields. *J. Assoc. Comput. Mach.*, 42(1):67–90, 1995. doi:10.1145/200836.200853.

5 Minati De and Abhiruk Lahiri. Geometric dominating-set and set-cover via local-search. *Comput. Geom.*, 113:102007, 2023. doi:10.1016/J.COMGEO.2023.102007.

6 Jie Gao and Li Zhang. Well-separated pair decomposition for the unit-disk graph metric and its applications. *SIAM J. Comput.*, 35(1):151–169, 2005. doi:10.1137/S0097539703436357.

7 S. Har-Peled. *Geometric Approximation Algorithms*, volume 173 of *Math. Surveys & Monographs*. Amer. Math. Soc., Boston, MA, USA, 2011. URL: http://sarielhp.org/book/, doi:10.1090/surv/173.

8 S. Har-Peled, H. Kaplan, and M. Sharir. Approximating the k-level in three-dimensional plane arrangements. In M. Loebl, J. Nešetřil, and R. Thomas, editors, *A Journey Through Discrete Mathematics: A Tribute to Jiří Matoušek*, pages 467–503. Springer, 2017. doi:10.1007/978-3-319-44479-6.

9 S. Har-Peled and M. Mendel. Fast construction of nets in low dimensional metrics, and their applications. *SIAM J. Comput.*, 35(5):1148–1184, 2006. doi:10.1137/S0097539704446281.

10 Sariel Har-Peled, Benjamin Raichel, and Eliot W. Robson. Well-separated pairs decomposition revisited. *CoRR*, 2509.05997, 2025. arXiv:2509.05997, doi:10.48550/arXiv.2509.05997.

11 Carsten Lund and Mihalis Yannakakis. On the hardness of approximating minimization problems. *J. Assoc. Comput. Mach.*, 41(5):960–981, 1994. doi:10.1145/185675.306789.

12 Dana Moshkovitz. The projection games conjecture and the NP-Hardness of $\ln n$-approximating set-cover. *Theory of Computing*, 11(7):221–235, 2015. doi:10.4086/toc.2015.v011a007.

13 N. H. Mustafa and S. Ray. PTAS for geometric hitting set problems via local search. In *Proc. 25th Annu. Sympos. Comput. Geom.* (SoCG), pages 17–22, 2009.

14 Giri Narasimhan and Michiel H. Smid. *Geometric spanner networks*. Cambridge University Press, 2007. doi:10.1017/CBO9780511546884.

15 Peter Slavík. A tight analysis of the greedy algorithm for set cover. *J. Algorithms*, 25(2):237–254, 1997. doi:10.1006/jagm.1997.0887.

16 Michiel H. Smid. The well-separated pair decomposition and its applications. In *Handbook of Approximation Algorithms and Metaheuristics (2)*. Chapman and Hall/CRC, 2018.

On minimum Venn diagrams*

Sofia Brenner[1], Petr Gregor[2], Torsten Mütze[1], and Francesco Verciani[1]

1 **Institute of Mathematics, University of Kassel, Germany**
{sbrenner, tmuetze, verciani}@mathematik.uni-kassel.de
2 **Department of Theoretical Computer Science and Mathematical Logic, Charles University, Prague, Czech Republic**
gregor@ktiml.mff.cuni.cz

Abstract

An *n-Venn diagram* is a diagram in the plane consisting of n simple closed curves that intersect only finitely many times such that each of the 2^n possible intersections of their interiors is represented by a single connected region. An n-Venn diagram has at most $2^n - 2$ crossings, and if this maximum number of crossings is attained, then at each crossing exactly two curves intersect. To complement this, Bultena and Ruskey considered n-Venn diagrams that minimize the number of crossings, which implies that many curves intersect in every crossing. Specifically, they proved that the total number of crossings in any n-Venn diagram is at least $L_n := \lceil \frac{2^n-2}{n-1} \rceil$. Diagrams achieving this bound are called *minimum* Venn diagrams, and are known only for $n \leq 7$. Bultena and Ruskey conjectured that they exist for all $n \geq 8$. In this work, we establish an asymptotic version of their conjecture, obtaining n-Venn diagrams with $(1+o(1))L_n$ crossings when n is a power of 2, and n-Venn diagrams with $(2+o(1))L_n$ crossings in general. Our constructions are based on partitions of the hypercube into isometric paths and cycles, using a result of Ramras.

Related Version *Full Version*: https://arxiv.org/abs/2511.09230 [1]

1 Introduction

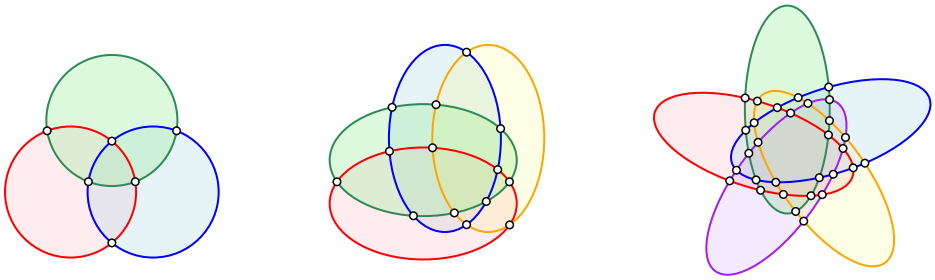
An *n-Venn diagram* is a collection of $n \geq 1$ simple closed curves in the plane that intersect in only finitely many points and that create exactly 2^n connected regions, one for every possible combination of being inside or outside with respect to each curve. It is easy to see that in any intersection point of at least two curves, at least two of them must cross, and thus we refer to any such intersection point as a *crossing*.

Venn diagrams are an appealing tool to visualize sets and their containment relations, and they are named after the English mathematician John Venn (1834–1923), who used them in the context of propositional logic [6]. Despite the simple definition, many questions about Venn diagrams lead to interesting and challenging mathematical and computational problems, which triggered a long and fruitful line of research devoted to them; see Ruskey and Weston's survey [5] and the many beautiful illustrations therein. These problems touch and connect various areas such as discrete geometry, graph drawing, graph theory, poset theory, coding theory, and enumerative combinatorics.

General constructions of n-Venn diagrams, valid for every $n \geq 1$, were provided already by Venn [6] and much later by Edwards [3]. In fact, these two constructions are connected via the well-known binary reflected Gray code, a listing of all 2^n binary strings of length n such that any two consecutive strings differ in a single bit.

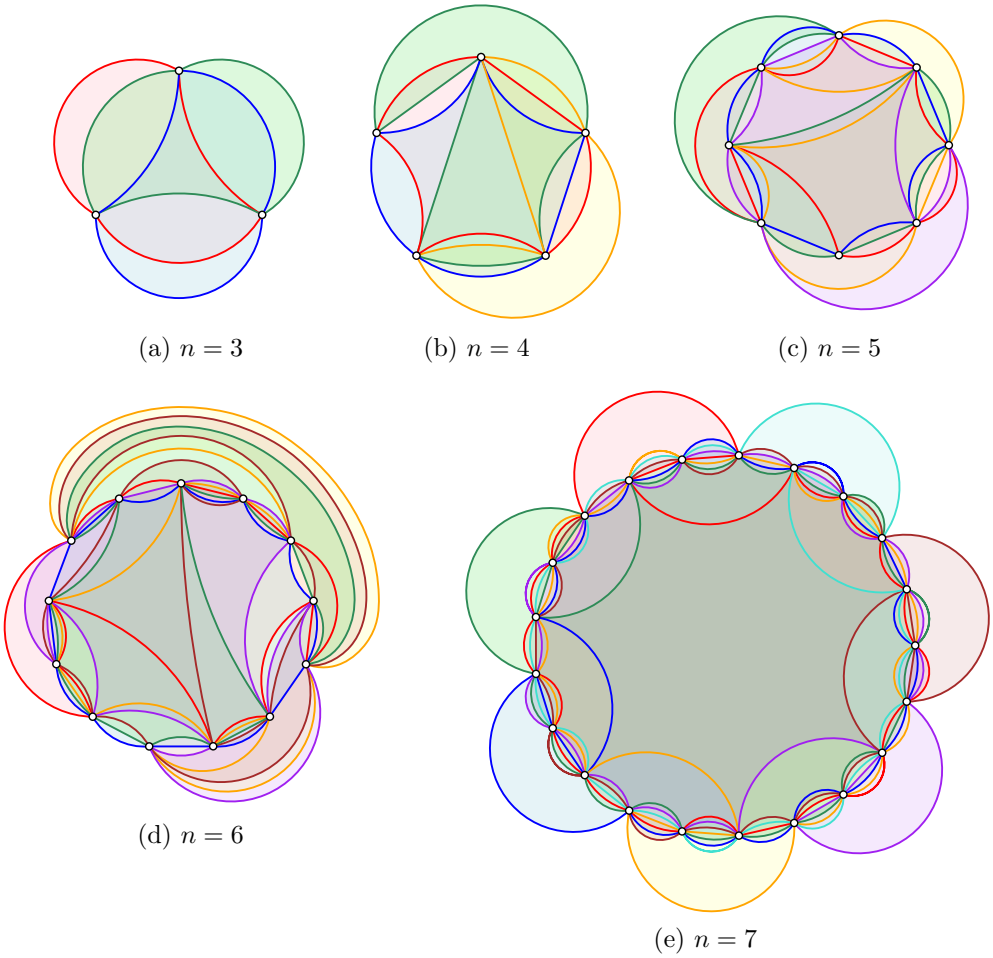
* Sofia Brenner, Torsten Mütze, and Francesco Verciani were supported by German Science Foundation grant 522790373.

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.



(a) $n = 3$ (b) $n = 4$ (c) $n = 5$

Figure 1 Simple Venn diagrams for $n = 3, 4, 5$, maximizing the number of crossings. For $n = 3, 4, 5$ the number of crossings $2^n - 2$ equals $6, 14, 30$.



(a) $n = 3$ (b) $n = 4$ (c) $n = 5$

(d) $n = 6$

(e) $n = 7$

Figure 2 Non-simple Venn diagrams for $n = 3, 4, 5, 6, 7$ that minimize the number of crossings. For those values of n, the number of crossings $L_n = \lceil \frac{2^n-2}{n-1} \rceil$ equals $3, 5, 8, 13, 21$. Only in the diagrams (a) and (e) every crossing involves all n curves.

1.1 Maximizing and minimizing the number of crossings

In this work, we are particularly interested in the number of crossings in an n-Venn diagram. A Venn diagram is *simple* if every crossing involves only two curves; see Figure 1. It is easy to see that simple n-Venn diagrams have exactly $2^n - 2$ many crossings, and this is the largest possible number of crossings among all n-Venn diagrams. To complement this, Bultena and Ruskey [2] considered Venn diagrams with the smallest possible number of crossings. They proved that any n-Venn diagram with $n \geq 2$ has at least

$$L_n := \left\lceil \frac{2^n - 2}{n - 1} \right\rceil$$

many crossings, and they called diagrams achieving this lower bound *minimum* Venn diagrams. If $(2^n - 2)/(n - 1)$ is integral, which happens for $n = 2, 3, 7, 19, 43, 55, 127, 163, \ldots$ (this is OEIS sequence A014741 incremented by 1), then this means that every crossing of the diagram involves all n curves. So far, minimum Venn diagrams are only known for $n \leq 7$; see Figure 2. Bultena and Ruskey [2] conjectured that minimum n-Venn diagrams exist for all $n \geq 8$, and this is also mentioned as an open problem in Ruskey and Weston's survey [5].

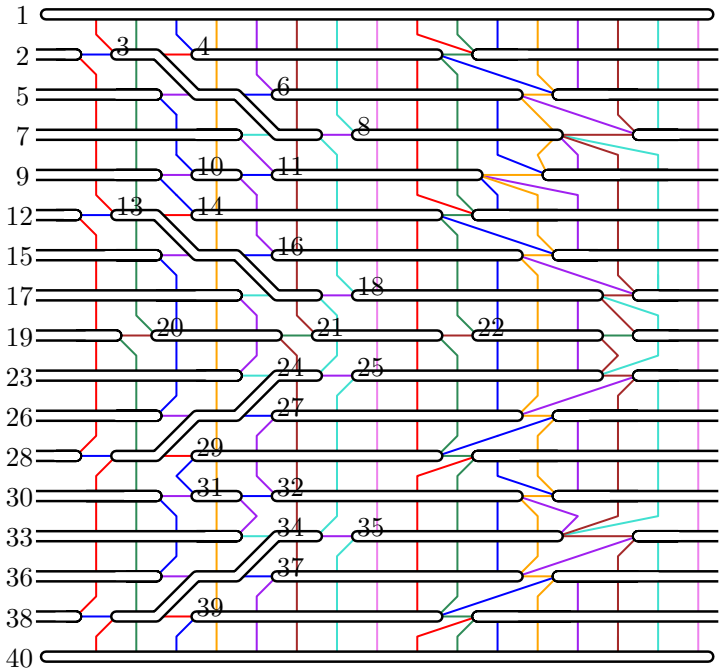


Figure 3 An 8-Venn diagram with 40 crossings obtained from Theorem 1.1. The 8 curves are drawn with colors. Each black/white bubble represents one crossing that has to be contracted to a single point, where the open bubbles at the left and right boundary wrap around at the bottom as if drawn on a cylinder.

1.2 Our results

Our first construction yields n-Venn diagrams for the case when n is a power of 2 in which the number of crossings is minimized up to a $(1 + o(1))$-factor (as $n \to \infty$).

▶ **Theorem 1.1.** *There is an 8-Venn diagram with 40 crossings, and for every $k \geq 4$ and $n \coloneqq 2^k$, there is an n-Venn diagram with exactly*

$$\left(1 + \frac{33}{8n} - \frac{2}{2^{n/2}} - \frac{2n}{2^n}\right)\frac{2^n}{n} \leq \left(1 + \frac{33}{8n}\right) L_n = (1 + o(1))L_n$$

many crossings.

Figure 3 shows the 8-Venn diagram with 40 crossings, which is only 3 more than the lower bound $L_8 = 37$.

For a general number of curves, i.e., when n is not necessarily a power of 2, we apply a doubling construction to obtain an n-Venn diagram in which the number of crossings is minimized up to a $(2 + o(1))$-factor.

▶ **Theorem 1.2.** *For every $0 \leq m \leq 7$ there is an $(8 + m)$-Venn diagram with $40 \cdot 2^m$ crossings, and for every $k \geq 4$, $n \coloneqq 2^k$ and $0 \leq m < n$, there is an $(n + m)$-Venn diagram with exactly*

$$\left(1 + \frac{33}{8n} - \frac{2}{2^{n/2}} - \frac{2n}{2^n}\right)\frac{2^{n+m}}{n} \leq \left(1 + \frac{33}{8n}\right)\frac{n+m}{n} L_{n+m} \leq (2 + o(1))L_{n+m}$$

many crossings.

In particular, Theorem 1.2 gives n-Venn diagrams with the smallest known number of crossings for all $n \geq 8$; see Table 1.

Table 1 The number of crossings in the Venn diagrams obtained from Theorem 1.2 versus the lower bound, and the minimum numbers for monotone diagrams.

n	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
lower bound $L_n = \left\lceil \frac{2^n-2}{n-1} \right\rceil$ [2]	0	2	3	5	8	13	21	37	64	114	205	373	683	1261	2341	4369
Figure 2	0	2	3	5	8	13	21									
Theorem 1.2								40	80	160	320	640	1280	2560	5120	5118

2 Preliminaries

Venn diagrams are conveniently studied by considering their dual graph; see Figure 4. We consider the *n-dimensional hypercube Q_n*, or *n-cube* for short, the graph formed by all subsets of $[n] \coloneqq \{1, 2, \ldots, n\}$, with an edge between any two sets that differ in a single element. Specifically, the vertex in $Q(D)$ corresponding to a region of the diagram D is the set of all indices of curves that contain this region in their interior. For an edge $e = \{x, x \cup \{i\}\}$ of Q_n, we refer to i as the *direction* of e.

In this section, we briefly characterise the dual graphs of Venn diagrams, and describe a partition of the hypercube into isometric cycles, which will be the main components of our constructions.

2.1 Duals of Venn diagrams

The dual graph $Q(D)$ of an n-Venn diagram D satisfies the following properties:

① It is a spanning subgraph of Q_n, i.e., all 2^n vertices are present. Specifically, the vertex in $Q(D)$ corresponding to a region of the diagram D is the set of all indices of curves that contain this region in their interior.

② It is a plane graph, i.e., it is drawn in the plane without edge crossings, such that each face has even length 2ℓ for some integer $2 \le \ell \le n$ and contains exactly two edges of ℓ distinct directions. These ℓ directions correspond to the ℓ curves intersecting in the crossing corresponding to the face.

③ For every $j \in [n]$, the two subgraphs of $Q(D)$ induced by all vertices x with $j \in x$, and with $j \notin x$, respectively, are connected. This corresponds to the jth curve being simple.

Conversely, the dual of any subgraph of Q_n satisfying these properties is an n-Venn diagram.

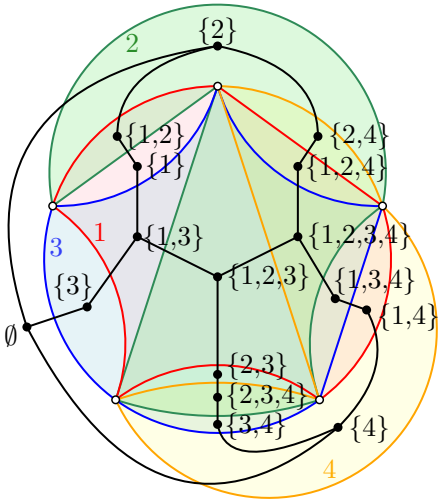


Figure 4 The dual graph of the 4-Venn diagram from Figure 2 (b) is a spanning subgraph of Q_4.

Clearly, the number of crossings of the diagram D equals the number of faces of the dual graph $Q(D)$. If we want to construct a minimum n-Venn diagram D, then as many crossings as possible (ideally almost all) should involve all n curves, i.e., in the dual graph $Q(D)$, we want almost all faces to have length $2n$.

2.2 Isometric partitions of hypercubes

For any vertex $x \subseteq [n]$ of the hypercube Q_n, we denote its *antipodal* vertex by $\overline{x} := [n] \setminus x$. Furthermore, given an integer $k \ge 1$ we define $k \cdot x := \{ki \mid i \in x\}$. Both of these operations extend to sets and sequences in the natural way. Finally, if $X \subset Q_{n-1}$, we define $X \cup \{n\} := \{x \cup \{n\} \mid x \in X\} \subset Q_n$.

A subgraph H of a graph G is *isometric* if it preserves distances, i.e., $d_H(u,v) = d_G(u,v)$ for any two vertices $u, v \in V(H)$. In particular, a path in the hypercube Q_n is isometric if and only if it contains no two edges of the same direction, and a cycle in Q_n is isometric if and only if the edges of the same direction come in pairs that lie oppositely on the cycle. Ramras [4] described a partition of the hypercube Q_{n-1}, for $n = 2^k$ and $k \ge 1$, into isometric paths. For this we consider the sequence of sets $(C_k)_{k \ge 1}$ defined recursively by

$$\begin{aligned} &C_1 := O_1 := \emptyset, \\ &O_k := \{\{1,3\},\{3,5\},\dots,\{2^k-3,2^k-1\}\} \text{ for } k \ge 2, \text{ and} \\ &C_k := O_k \cup 2C_{k-1} \text{ for } k \ge 2. \end{aligned} \tag{1}$$

For sets $x, y \subseteq [n]$ we define their symmetric difference as $x \oplus y := (x \setminus y) \cup (y \setminus x)$. The sets defined in (1) are a different basis than the one considered by Ramras, but the resulting

generated subspace $\langle C_k \rangle := \{x_1 \oplus x_2 \oplus \cdots \oplus x_t \mid x_i \in C_k \text{ for all } i \in [t]\}$ is the same, so we can apply his results. For any $x \in \langle C_k \rangle$, we define a path $P(x)$ of length $n-1$ in Q_{n-1} by

$$P(x) := (x, x \oplus \{1\}, x \oplus \{1,2\}, \ldots, x \oplus \{1,2,\ldots,n-1\}). \tag{2}$$

▶ **Theorem 2.1** ([4, Thm. 2.5]). *For any $k \geq 1$ and $n := 2^k$, the isometric paths $P(x)$ of length $n-1$ defined in (2) for all $x \in \langle C_k \rangle$ form a partition of the vertex set of Q_{n-1}.*

▶ **Corollary 2.2.** *For any $k \geq 1$ and $n := 2^k$, the isometric cycles $C(x)$ of length $2n$ defined by $C(x) := (P(x), \overline{P(x)} \cup \{n\})$ for all $x \in \langle C_k \rangle$ form a partition of the vertex set of Q_n.*

3 Construction of asymptotically minimum Venn diagrams

Both Theorem 1.1 and Theorem 1.2 have constructive proofs, which we sketch in this section. Detailed proofs can be found in the full version of this extended abstract [1].

3.1 Construction for $n = 2^k$ - proof sketch of Theorem 1.1

Using the characterization presented in Section 2.1, we build n-Venn diagrams by constructing planar spanning subgraphs of the hypercube Q_n satisfying conditions ①–③, with the goal of minimizing the number of faces. In other words, we want to have faces that are as large as possible.

In the first step of the construction, we embed the isometric cycles of the partition into the plane concentrically in an order that is specified by a certain Hamiltonian path P of Q_d, with $d = 2^k - k - 1 = |C_k|$. This path is shown on the left side of Figure 7. Each vertex of Q_d represents coefficients of a linear combination of the basis C_k, so consecutive concentric cycles in this order differ only by one element of the basis C_k.

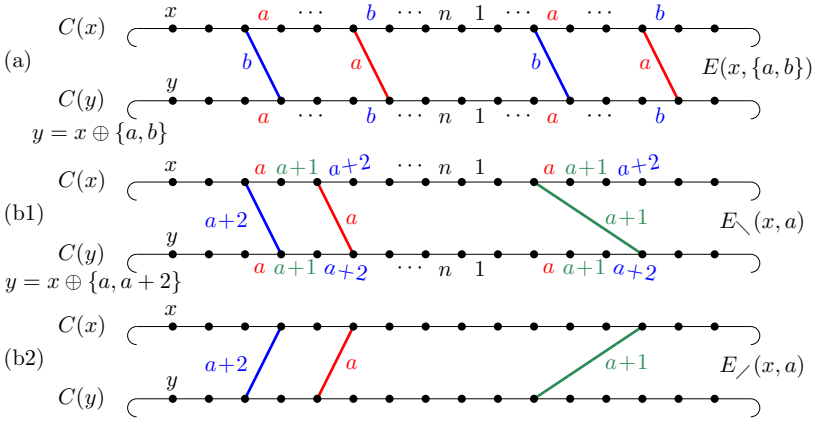


Figure 5 (a),(b1),(b2): Illustration of possible edges in the annulus between consecutive cycles $C(x), C(y)$, and definition of edge sets $E(x, \{a, b\}), E_{\diagdown}(x, a), E_{\diagup}(x, a)$.

We consider the annulus between two consecutive concentric cycles $C(x)$ and $C(y)$, where x and y differ by one element of C_k, i.e., $y = x \oplus \{a, b\}$ for some $\{a, b\} \in C_k$, see Figure 5.

- In general, it is possible to add edges to subdivide the annulus between cycles $C(x)$ and $C(y)$ into four faces, see Figure 5 (a). We refer to this set of four edges as $E(x, \{a, b\})$.
- If $\{a, b\} = \{a, a+2\}$, then it is possible to add edges to subdivide the annulus between cycles $C(x)$ and $C(y)$ into three faces, as shown in Figure 5 (b1),(b2), and we refer to the respective sets of three edges as $E_{\diagdown}(x, a)$ and $E_{\diagup}(x, a)$.

Recall that we want to minimize the number of faces, so we want to use the second option as often as possible. This is the reason for our definition of the basis C_k, instead of the one used by Ramras [4], since our basis maximizes elements of the form $\{a, a+2\}$.

We can make further improvements, as shown in Figure 6. Consider consecutive cycles $C(x), C(y), C(z)$ such that $y = x \oplus \{a, a+2\}$ and $z = y \oplus \{a+2, a+4\}$.

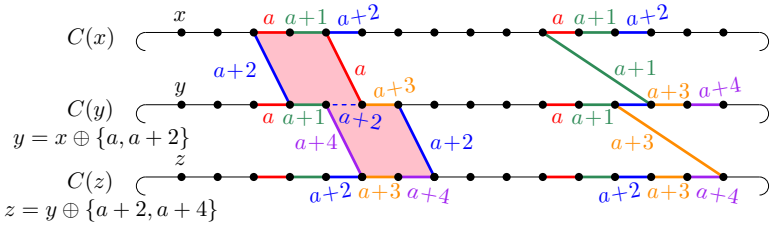


Figure 6 Illustration of the merging procedure between two 6-faces.

After inserting edges $E_\searrow(x, a)$ and $E_\searrow(y, a+2)$, it is possible to remove an edge of the cycle $C(y)$ in order to merge two 6-faces. Applying these techniques for $n = 2^3 = 8$ yields an 8-Venn diagram with only 40 crossings, as shown by its dual graph with 40 faces in Figure 7, with Figure 3 showing its corresponding primal graph.

In general, an appropriate choice of the Hamiltonian path P allows us to construct an n-Venn diagram with $(1 + o(1))L_n$ crossings, as stated in Theorem 1.1.

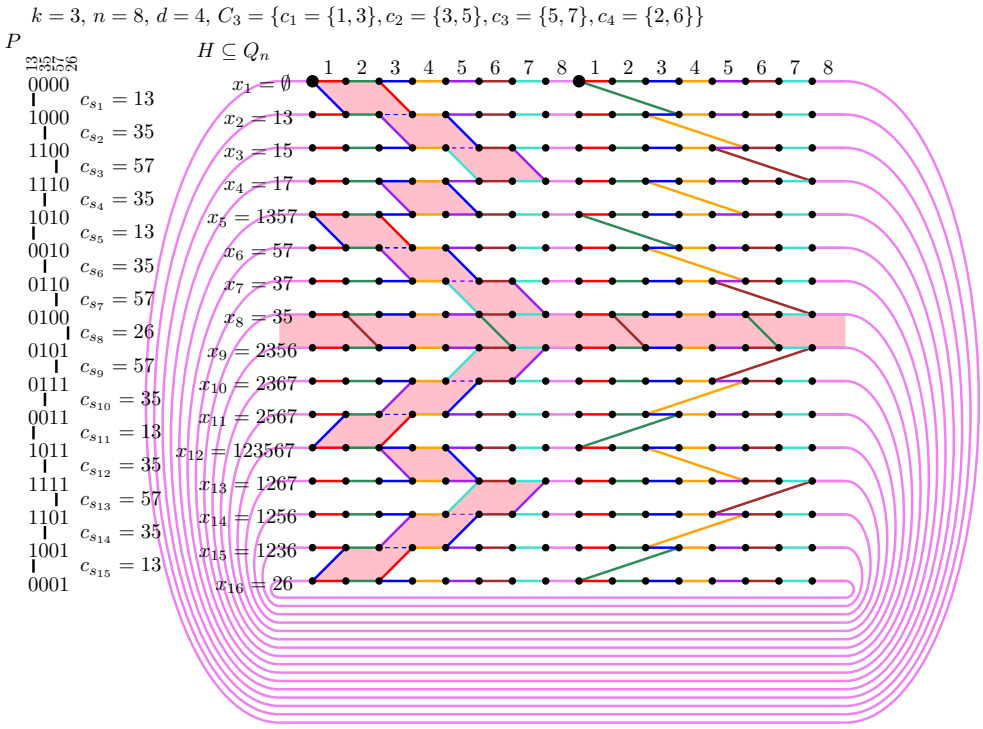


Figure 7 Dual graph of a Venn diagram for the case $k = 3$, i.e., $n = 8$. The dashed edges are deleted, so they are not part of H. The constructed dual graph H of an 8-Venn diagram has 40 faces, i.e., the corresponding Venn diagram has 40 crossings; see Figure 3, only 3 more than the lower bound $L_8 = 37$. All faces except the red ones have maximum possible length $2n$.

3.2 Doubling construction - proof sketch of Theorem 1.2

We now show how to lift the construction of n-Venn diagrams for the case when n is a power of 2 to the general case, thus proving Theorem 1.2. The lifting is achieved via the following straightforward doubling construction.

We say that a face of the dual graph of an n-Venn diagram is *colorful* if it has length $2n$ and contains two antipodal vertices of Q_n. In the primal Venn diagram, such a face corresponds to a crossing involving all n curves for which the cyclic ordering of curves around the crossing can be split into two halves such that every curve appears exactly once in each half. Consequently, there is a way to draw an additional curve through this crossing that crosses (not only intersects) each of the n existing curves.

The construction is illustrated in Figure 8. Starting from the dual graph $Q(D) \subseteq Q_n$ of an n-Venn diagram D in which the outer face is colourful with antipodal vertices x and $\overline{x}$, we construct the dual graph H of an $(n+1)$-Venn diagram as follows: We take two copies of $Q(D)$, namely $Q(D)$ and $Q(D) \cup \{n+1\}$ (where $Q(D) \cup \{n+1\}$ represents the graph $Q(D)$ with the element $\{n+1\}$ added to each vertex), embedded into the plane next to each other, and we add the two edges $\{x, x \cup \{n+1\}\}, \{\overline{x}, \overline{x} \cup \{n+1\}\}$ between them.

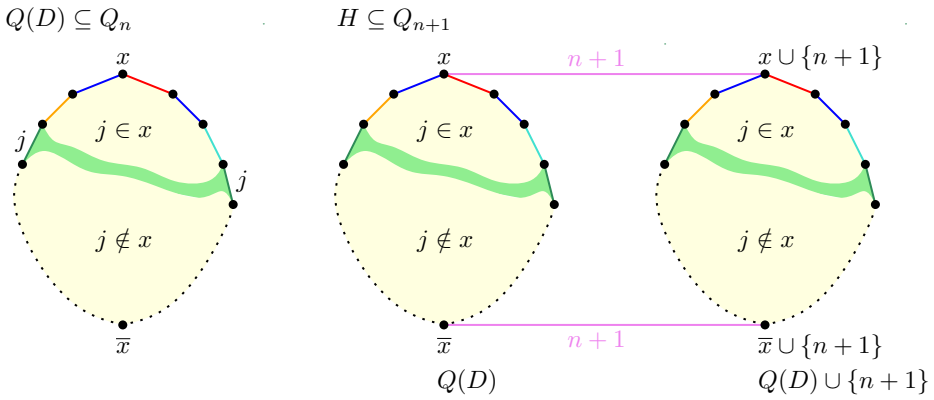


Figure 8 Illustration of the proof of Theorem 1.2.

The resulting graph H satisfies conditions ①–③ (in particular, note that Figure 8 emphasizes that the subgraphs respectively containing and not containing an arbitrary element $j \in [n+1]$ are connected), and thus is the dual graph of an $(n+1)$-Venn diagram. Moreover, H has twice as many faces as $Q(D)$, so the corresponding Venn diagram has twice as many crossings as D. Finally, the outer face of H is again colourful, which allows us to use H as the starting graph for a further doubling construction. We conclude by noting that that for $n = 2^k$, our construction in the previous section yields a diagram in which the outer face is colourful, as desired, so using that graph as starting point and applying the doubling construction m times yields Theorem 1.2.

References

1 Sofia Brenner, Petr Gregor, Torsten Mütze, and Francesco Verciani. On minimum Venn diagrams, 2025. https://arxiv.org/abs/2511.09230.

2 Bette Bultena and Frank Ruskey. Venn diagrams with few vertices. *Electron. J. Combin.*, 5:Research Paper 44, 21 pp., 1998. doi:10.37236/1382.

3 Anthony W. F. Edwards. Venn diagrams for many sets. In *Bulletin of the International Statistical Institute, 47th Session, Paris, Book 1*, pages 311–312. 1989.

4 Mark Ramras. Symmetric vertex partitions of hypercubes by isometric trees. *J. Combin. Theory Ser. B*, 54(2):239–248, 1992. doi:10.1016/0095-8956(92)90055-3.

5 Frank Ruskey and Mark Weston. A survey of Venn diagrams. *Electron. J. Combin.*, 4(1):Dynamic Survey 5, 10 HTML documents, 1997. doi:10.37236/26.

6 John Venn. On the diagrammatic and mechanical representation of propositions and reasonings. *Phil. Mag. S. 5.*, 9(59):1–18, 1880. doi:10.1080/14786448008626877.

Edge Densities of Drawings of Graphs with One Forbidden Cell*

Benedikt Hahn[1], Torsten Ueckerdt[2], and Birgit Vogtenhuber[1]

1 **Graz University of Technology, Austria**
{benedikt.hahn,birgit.vogtenhuber}@tugraz.at
2 **Karlsruhe Institute of Technology, Germany**
torsten.ueckerdt@kit.edu

Abstract

A connected topological drawing of a graph divides the plane into a number of cells. The *type* of a cell c is the cyclic sequence of crossings and vertices along the boundary walk of c. For example, all triangular cells with three incident crossings and no incident vertex share the same cell type. In this work, we initiate the in-depth study of drawings that do not contain one fixed cell type $\mathfrak{c}$, and investigate the edge density of the corresponding graph classes.

Related Version The results in this extended abstract were previously presented at GD [11] and a full version is available on arXiv [12].

1 Introduction

Many fundamental classes of graphs are defined by admitting a drawing without a forbidden configuration or pattern. This includes planar graphs (forbidding two crossing edges), as well as many beyond-planar graph classes such as k-planar graphs (forbidding an edge with $k+1$ crossings), quasiplanar graphs (forbidding three pairwise crossing edges), and many others. One of the most fundamental questions in this area concerns the *edge density* of a graph class $\mathcal{G}$, that is, the maximum possible number of edges of any n-vertex graph in $\mathcal{G}$. There is a long history of edge density results for a variety of beyond-planar graph classes (e.g. [1–4,7,15–17]) many of which were reproven with the recently introduced Density Formula [14].

We study the *cells* of a drawing, that is, the regions of $\mathbb{S}^2$ delimited by the vertices and edges of the drawing. The *type* of a cell c is the cyclic order of edge segments, vertices, and crossings along the boundary of c. For example, a -cell is incident to three edge segments, three crossings, and no vertex. Here, the type of a cell does not depend on whether some vertices, edge segments, or crossings on the boundary are identical. Distinguishing cells by their type is often useful when reasoning about drawings (see for example [13]). The recent Density Formula [14], whose various applications usually come down to counting and relating different cell types in drawings further increased the need for a good grasp on the relation between types of cells in a drawing and the structure of the underlying graph. Interestingly, Ackerman and Tardos [3] proved already in 2007, while considering quasiplanar drawings, that non-homotopic n-vertex drawings with no -cell contain at most $8n-20$ edges.

Our contribution. We consider for each cell type $\mathfrak{c}$ the corresponding beyond-planar graph class $\mathcal{G}_\mathfrak{c}$ consisting of all graphs that admit a drawing without cells of type $\mathfrak{c}$ (called $\mathfrak{c}$-*free*). We present upper and lower bounds on the edge densities of graphs in $\mathcal{G}_\mathfrak{c}$ when arbitrary

* This research was funded in part by the Austrian Science Fund (FWF) 10.55776/DOC183 and by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – 520708409.

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

drawings are admitted, and when drawings are required to be simple or non-homotopic. For each of these settings and each cell type (with one exception), we determine whether the asymptotic edge density is linear or superlinear. Moreover, in most cases, our bounds are tight up to an additive constant. Our results and some previous related work are summarized in Table 1. Our lower bounds entail various novel constructions while our upper bounds are proved via discharging arguments.

Forbidden cell	Graph type	Drawing type	Lower Bound	Upper Bound
Any	Simple	No restriction	$\binom{n}{2}$ Construction 3	$\binom{n}{2}$
	Multi	Non-homotopic	$8n-20$ [14]	$8n-20$ [3]
	Simple	Non-homotopic	$8n-28$ Construction 7	$8n-20$ [3]
	-	Simple	$7n-30$ Cons. 8 based on [3]	$7n-20$ [3]
	Multi	Non-homotopic	$9n-18$ Construction 6	∞
	Simple	Non-homotopic	$6n-12$ Construction 5	$\binom{n}{2}$
	-	Simple	$6n-12$ Construction 5	
	Simple	Non-homotopic	$\Omega(n^2)$ Construction 11	$\binom{n}{2}$
	-	Simple	$\Omega(n^{3/2})$ Construction 10	
	Multi	Non-homotopic	$6n-12$ Construction 5	$6n-12$ Theorem 4
	-	Simple	$6n-12$ Construction 5	$6n-12$ Theorem 4
Others	-	Simple	$\binom{n}{2}$ Construction 2	$\binom{n}{2}$

Table 1 Results on edge densities of graphs admitting a drawing without one fixed type of cell. Upper bounds hold for all $n \geq 4$, while lower bounds are proven for infinitely many values of n.

Related work. Apart from the aforementioned works of Ackerman and Tardos [3] and Kaufmann, Klemz, Knorr, Reddy, Schröder, and Ueckerdt [14] on -free drawings in the context of quasiplanar drawings, we are (as far as we know) the first to consider graph drawings with one forbidden cell type. However, some drawing styles are characterized by forbidding an infinite number of cells. These include planar and 1-planar drawings as well as k^+-real face drawings [5]. Further, some beyond-planar drawing styles are characterized by forbidding a configuration similar to a particular cell type. For example, a fan-crossing-free drawing [6] contains no edge crossed by two incident edges, and is thus -free. Quasiplanar drawings and -cells, and $(2,2)$-grid-free drawings and -cells have analogous correspondences. Thus, lower bound constructions for these beyond-planar graph classes carry over to our setting. For a collection of edge density results on beyond-planar graph classes, we refer to [8, Chapter 4]. Cell types have also been investigated for pseudoline arrangements (see [10] for an overview).

2 Preliminaries: Drawings, Cells, and Cell Types

We consider only finite (multi)graphs with no loops. To make the possible presence of parallel edges explicit or exclude them, we often write *multigraph* or *simple graph*. A drawing Γ of a multigraph maps vertices to pairwise distinct points on the sphere $\mathbb{S}^2$ and edges to curves connecting the corresponding vertices. As customary, we require that no edge passes through a vertex, any two edges have only finitely many points in common, each being a common endpoint or a proper crossing, and that no three edges cross in the same point. A drawing Γ is *connected* if the image of the map Γ is connected in $\mathbb{S}^2$.

A drawing is *simple* if any two edges share at most one point, that is, either a common endpoint or a crossing. Only simple graphs admit simple drawings. A *lens* is a region of $\mathbb{S}^2$ bounded by exactly two parts of edges. This could be two parallel edges, two edges crossing twice, or two crossing adjacent edges. We call a lens *empty* if its interior contains neither a crossing nor a vertex. A drawing is *non-homotopic* if it contains no empty lens.

Consider any fixed connected drawing Γ of some multigraph G. The crossings split each edge into *edge segments*. An *outer edge segment* is incident to a vertex, while an *inner edge segment* starts and ends with a crossing. The *cells* of Γ are the connected components of $\mathbb{S}^2 \setminus \Gamma(V \cup E)$. For a cell c, we denote by $v(c)$, $e(c)$, and $e_{\text{in}}(c)$ the number of vertices, edge segments, and inner edge segments on its boundary, counted with multiplicity. The *size* of c is the sum of the two: $\|c\| := e(c) + v(c)$. We denote by $\mathcal{C}$ the set of all cells, and set $\mathcal{C}_k := \{c \in \mathcal{C} \colon \|c\| = k\}$. The number inscribed in the pictogram we use for a small cell type is the size of that cell.

▶ **Observation 1** (Kaufmann, Klemz, Knorr, Reddy, Schröder, and Ueckerdt [14]). *Let Γ be any non-homotopic connected drawing of some multigraph G with at least three vertices. Then, $\mathcal{C}_1 = \mathcal{C}_2 = \emptyset$, $\mathcal{C}_3$ is the set of all $\triangle$-cells, $\mathcal{C}_4$ is the set of all $\square$-cells and all $\measuredangle$-cells, and $\mathcal{C}_5$ is the set of all $\pentagon$-cells, $\measuredangle$-cells, and $\measuredangle$-cells.*

3 Drawings of Complete Graphs

First, we show that for most cell types $\mathfrak{c}$, complete graphs admit simple $\mathfrak{c}$-free drawings.

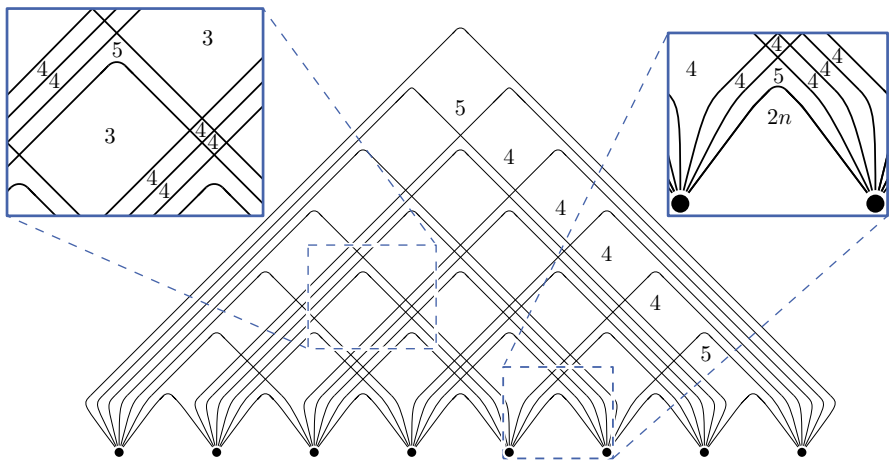


Figure 1 A simple drawing of K_n for $n = 8$ that contains only cells of size ≤ 5 and $2n$.

▶ **Construction 2.** *For every $n \geq 4$, the complete graph K_n admits a simple $\mathfrak{c}$-free drawing in which each cell c fulfills $\|c\| \leq 5$ or $\|c\| = 2n$* (see Figure 1).

By Observation 1, Construction 2 shows that for any cell type $\mathfrak{c} \notin \{$ 3-gon, 4-gon, 1-gon, 5-gon, 6-gon $\}$, every large enough complete graph admits a simple $\mathfrak{c}$-free drawing. To avoid 6-gon-cells in simple drawings of K_n, any drawing without uncrossed edges will do, see for example [13].

Thus, the edge density when forbidding most cells and restricting to simple graphs is $\binom{n}{2}$. When arbitrary drawings are permitted, complete graphs can also be drawn while avoiding the remaining four cell types 3-gon, 1-gon, 4-gon, and 5-gon.

▶ **Construction 3.** *For any cell type $\mathfrak{c}$ and every $n \geq 2$, the complete graph K_n admits a $\mathfrak{c}$-free drawing* (see Figure 2).

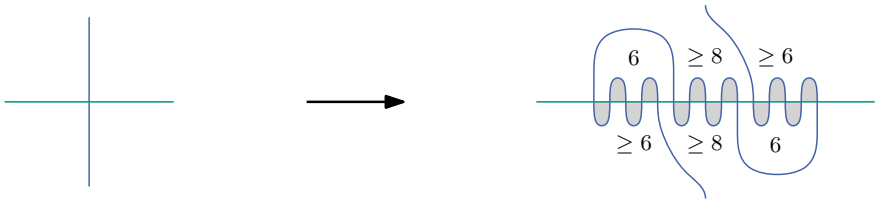


Figure 2 Any 3-gon-, 1-gon-, 4-gon-, or 5-gon-cell can be destroyed by redrawing crossings as depicted.

4 Drawings without 5-gon-Cells or 4-gon-Cells

In this section we give edge density bounds for non-homotopic 5-gon-free and 4-gon-free drawings. By relating the number of vertices and edges on the boundary of every cell type, a linear upper bound on the edge density of non-homotopic 5-gon-free drawings can be obtained. To prove the statement, we also show that in any non-homotopic drawing with at least three vertices, $3 \cdot$ 3-gon $+$ 1-gon $\leq \sum_{c \in \mathcal{C}_{\geq 5}} e_{\text{in}}(c)$, where, a pictogram denotes the number of times its depicted cell type appears in the drawing. As pointed out by a referee, the result can also be derived from the proof of a statement on straight-line drawings with crossing angle restrictions [9, Theorem 4.2].

▶ **Theorem 4.** *For $n \geq 3$, any connected n-vertex multigraph that admits a non-homotopic 5-gon-free drawing has at most $6n - 12$ edges.*

By Construction 5, the bound from Theorem 4 is tight, even for simple drawings. The construction starts with a plane triangulation and completes every $K_4 - e$ to a K_4.

▶ **Construction 5.** *For infinitely many values of n, there is a simple n-vertex graph with $6n - 12$ edges that admits a simple 4-gon-free and 5-gon-free drawing* (see Figure 3, Left).

For non-homotopic drawings without 4-gon-cells, a slight variation gives a better bound.

▶ **Construction 6.** *For infinitely many values of n, there is an n-vertex multigraph with $9n - 18$ edges that admits a non-homotopic 4-gon-free drawing* (see Figure 3, Right).

5 Drawings without 3-gon-Cells

3-gon-free drawings are closely related to *quasiplanar* drawings, that is, drawings in which no three edges cross pairwise. Ackerman and Tardos [3] showed that non-homotopic quasiplanar

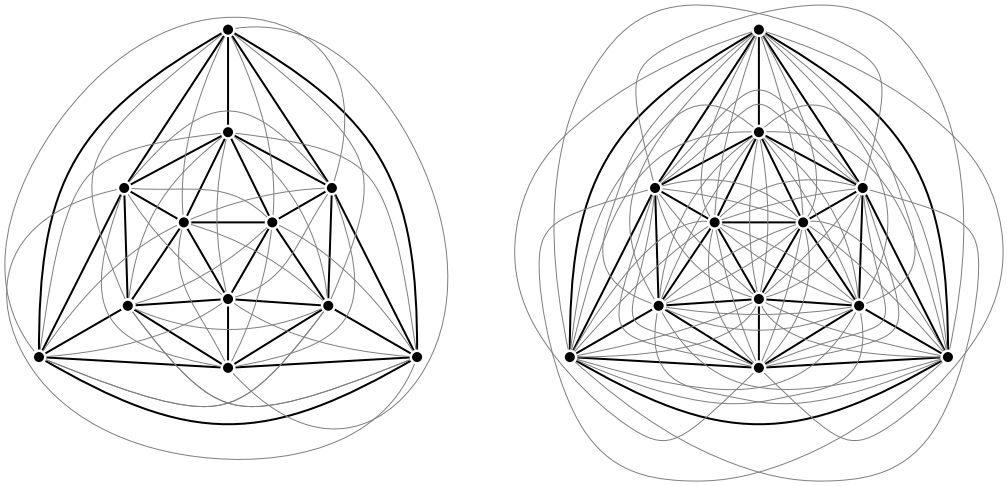


Figure 3 Left: A simple 4-free and 5-free drawing with $6n-12$ edges. Right: A non-homotopic 4-free drawing of a multigraph with $9n-18$ edges. On both sides, $n=12$.

drawings of multigraphs contain at most $8n-20$ edges and noted that their result carries over to the more relaxed setting of 3-free non-homotopic drawings of multigraphs. This bound was proven tight by a construction of non-homotopic quasiplanar (and therefore 3-free) drawings of multigraphs with $8n-20$ edges in [14].

For non-homotopic quasiplanar drawings of *simple* graphs, the best previous lower bound of $7.5n-\mathcal{O}(1)$ edges appears in [3]. We improve this bound to $8n-28$ in the 3-free setting, thus almost matching the upper bound of $8n-20$.

▶ **Construction 7.** *For infinitely many values of n, there is a simple graph on n vertices and $8n-28$ edges that admits a non-homotopic drawing without 3-cells* (see Figure 4).

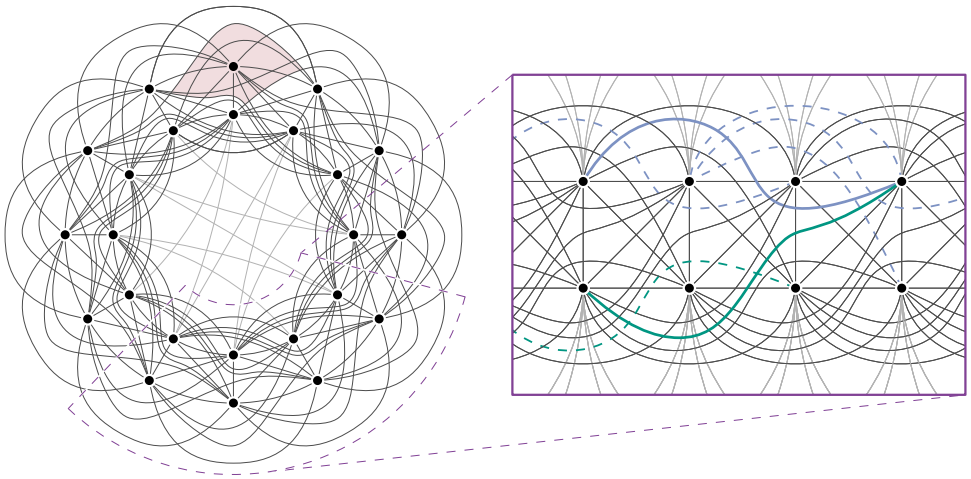


Figure 4 Construction 7 for $n=24$ (except some edges on the outer face), as well as a zoomed-in strip of the radial pattern. Colored elements highlight violations to quasiplanarity and simplicity.

For $\triangledown$-free simple drawings, Ackerman and Tardos [3] showed an upper bound of $7n-20$. They also mentioned that a $7n-\mathcal{O}(1)$ lower bound in this setting can be obtained by taking a subdrawing of the construction in [3, Theorem 4]. However, they did neither specify nor optimize the constant term. Doing so, we obtain a lower bound of $7n-30$.

▶ **Construction 8.** *For infinitely many values of n, there is a simple n-vertex graph with $7n-30$ edges that admits a $\triangledown$-free drawing.*

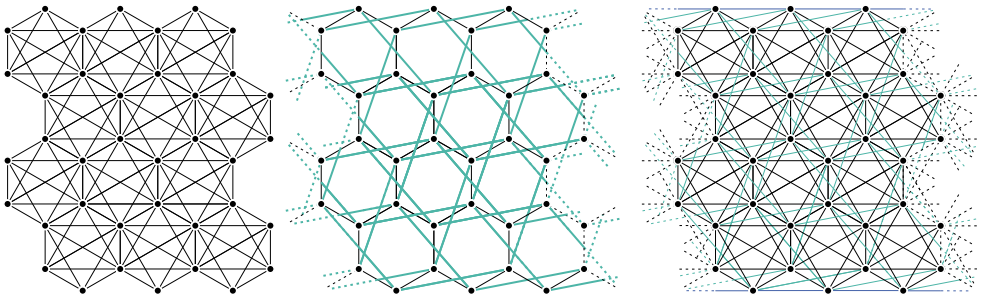


Figure 5 A simple $\triangledown$-free drawing obtained by a tiling of the cylinder with convex hexagons plus all but one of their diagonals and some long edges.

6 Drawings without $\triangle$-Cells

In this section, we consider the $\triangle$-cell. For simple drawings, so far no construction of $\triangle$-free drawings with a quadratic number of edges is known. However, if we allow drawings on the torus, we can achieve quadratic edge density.

▶ **Construction 9.** *For infinitely many n, there is a simple n-vertex graph with minimum degree $\geq \frac{n}{90}$ admitting a simple $\triangle$-free drawing on the torus* (see Figure 6).

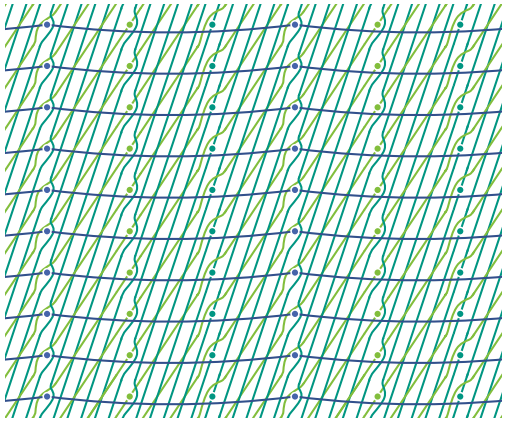


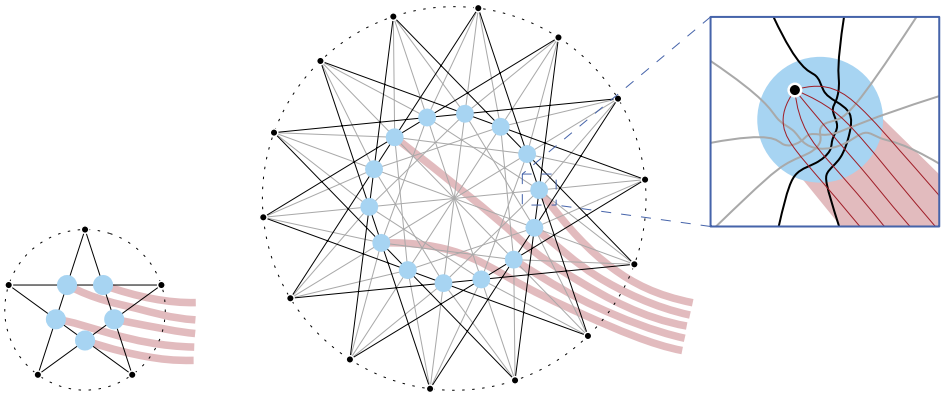
Figure 6 Part of a small simple $\triangle$-free drawing on the torus. Edges are almost straight and connect vertices that are three columns apart. Only in larger examples does the quadratic density become apparent.

Cutting edges in the toroidal construction and adding new vertices along the cut, we obtain simple $\triangle$-free drawings on the sphere with $\Omega(n^{3/2})$ edges.

▶ **Construction 10.** *For infinitely many n, there is a simple n-vertex graph with $> \frac{1}{29}n^{3/2}$ edges admitting a simple $\triangle$-free drawing.*

This leaves a gap to the trivial upper bound of $\mathcal{O}(n^2)$, which we are able to close in the relaxed setting of non-homotopic drawings, using a different construction.

▶ **Construction 11.** *For infinitely many values of n, there is a simple n-vertex graph of minimum degree $\geq \frac{n}{9}$ admitting a non-homotopic*[1] *$\triangle$-free drawing* (see Figures 7 and 8).



$n = 15$ $n = 42$

Figure 7 Building blocks used in Construction 11 for different values of n.

7 Concluding Remarks

In this work, we initiated the study of drawings of graphs with a forbidden type of cell. For arbitrary, non-homotopic, and simple drawings, and all cell types $\mathfrak{c}$ but $\square$, we determined whether graphs that admit a $\mathfrak{c}$-free drawing have at most linear or superlinear edge density. In many cases we obtain near-tight bounds. Being unable to control the $\square$-cell with our methods, we wonder in which regime the number of edges in $\square$-free drawings lies.

▶ **Open Problem 12.** *Is the edge density of simple graphs that admit a (simple, non-homotopic) $\square$-free drawing quadratic in n?*

For simple $\triangle$-free drawings of simple graphs, a gap remains between our $\Omega(n^{3/2})$ lower bound (Construction 10) and the trivial $\mathcal{O}(n^2)$ upper bound. In light of our quadratic lower bounds in the relaxed settings of toroidal simple drawings (Construction 9) and non-homotopic drawings (Construction 11), we believe the true edge density for simple $\triangle$-free drawings on the sphere should be quadratic.

▶ **Conjecture 13.** *The edge density of simple n-vertex graphs that admit a simple $\triangle$-free drawing is $\Omega(n^2)$.*

Acknowledgements. We thank an anonymous referee for their helpful comments, and in particular also for pointing us towards existing results [3, Theorem 6] and [9, Theorem 4.2] from the literature.

[1] In the GD version [11, Construction 9], we erroneously claimed that the resulting drawings are simple.

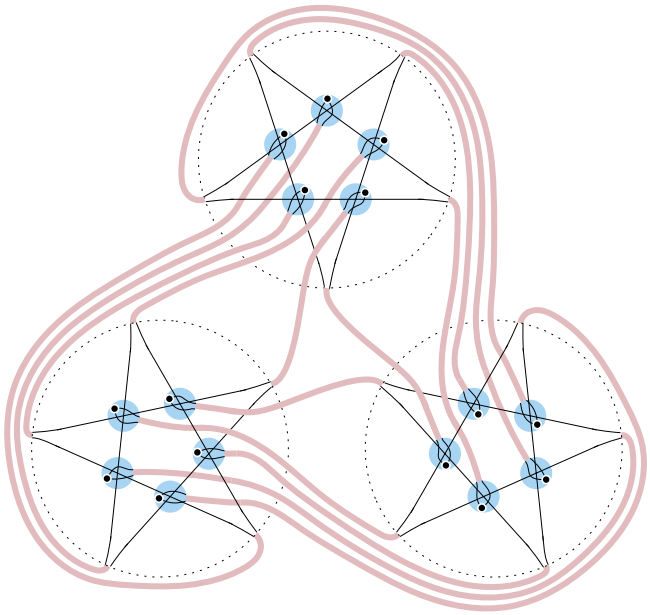


Figure 8 A non-homotopic -free drawing for $n = 15$, consisting of three interleaved copies of the component depicted in Figure 7. Pink corridors represent two edges each.

References

1. Eyal Ackerman. On the Maximum Number of Edges in Topological Graphs with no Four Pairwise Crossing Edges. *Discrete & Computational Geometry*, 41(3):365–375, 2009. doi:10.1007/s00454-009-9143-9.
2. Eyal Ackerman. On topological graphs with at most four crossings per edge. *Computational Geometry*, 85:101574, 2019. doi:10.1016/j.comgeo.2019.101574.
3. Eyal Ackerman and Gábor Tardos. On the maximum number of edges in quasi-planar graphs. *Journal of Combinatorial Theory Series A*, 114(3):563–571, 2007. doi:10.1016/J.JCTA.2006.08.002.
4. Patrizio Angelini, Michael A. Bekos, Henry Förster, and Michael Kaufmann. On RAC drawings of graphs with one bend per edge. *Theoretical Computer Science*, 828–829:42–54, 2020. doi:10.1016/j.tcs.2020.04.018.
5. Carla Binucci, Giuseppe Di Battista, Walter Didimo, Vida Dujmović, Seok-Hee Hong, Michael Kaufmann, Giuseppe Liotta, Pat Morin, and Alessandra Tappini. Graphs Drawn With Some Vertices per Face: Density and Relationships. *IEEE Access*, 12:68828–68846, 2024. doi:10.1109/ACCESS.2024.3401078.
6. Otfried Cheong, Sariel Har-Peled, Heuna Kim, and Hyo-Sil Kim. On the Number of Edges of Fan-Crossing Free Graphs. *Algorithmica*, 73:673–695, 2015. doi:10.1007/s00453-014-9935-z.
7. Walter Didimo, Peter Eades, and Giuseppe Liotta. Drawing graphs with right angle crossings. *Theoretical Computer Science*, 412(39):5156–5166, 2011. doi:10.1016/j.tcs.2011.05.025.
8. Walter Didimo, Giuseppe Liotta, and Fabrizio Montecchiani. A survey on graph drawing beyond planarity. *ACM Computing Surveys*, 52(1), 2019. doi:10.1145/3301281.
9. Vida Dujmovic, Joachim Gudmundsson, Pat Morin, and Thomas Wolle. Notes on Large Angle Crossing Graphs. *Chicago Journal of Theoretical Computer Science*, 17(1):1–14, 2011. doi:10.4086/cjtcs.2011.004.

10 Stefan Felsner and Jacob E Goodman. Pseudoline arrangements. In *Handbook of Discrete and Computational Geometry*, pages 125–157. Chapman and Hall/CRC, 2017.

11 Benedikt Hahn, Torsten Ueckerdt, and Birgit Vogtenhuber. Edge Densities of Drawings of Graphs with One Forbidden Cell. In *33nd International Symposium on Graph Drawing and Network Visualization (GD 2025)*, volume 357 of *Leibniz International Proceedings in Informatics (LIPIcs)*. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2025. to appear. doi:10.4230/LIPIcs.GD.2025.33.

12 Benedikt Hahn, Torsten Ueckerdt, and Birgit Vogtenhuber. Edge Densities of Drawings of Graphs with One Forbidden Cell, 2025. arXiv:2508.16368, doi:10.48550/arXiv.2508.16368.

13 Heiko Harborth and Ingrid Mengersen. Edges without crossings in drawings of complete graphs. *Journal of Combinatorial Theory, Series B*, 17(3):299–311, 1974. doi:10.1016/0095-8956(74)90035-5.

14 Michael Kaufmann, Boris Klemz, Kristin Knorr, Meghana M. Reddy, Felix Schröder, and Torsten Ueckerdt. The Density Formula: One Lemma to Bound Them All. In *32nd International Symposium on Graph Drawing and Network Visualization (GD 2024)*, volume 320 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 7:1–7:17. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024. doi:10.4230/LIPIcs.GD.2024.7.

15 János Pach, Rom Pinchasi, Micha Sharir, and Géza Tóth. Topological Graphs with No Large Grids. *Graphs and Combinatorics*, 21(3):355–364, 2005. doi:10.1007/s00373-005-0616-1.

16 Janos Pach, Rados Radoicic, Gabor Tardos, and Geza Toth. Improving the Crossing Lemma by Finding More Crossings in Sparse Graphs. *Discrete & Computational Geometry*, 36(4):527–552, 2006. doi:10.1007/s00454-006-1264-9.

17 János Pach and Géza Tóth. Graphs drawn with few crossings per edge. *Combinatorica*, 17(3):427–439, 1997. doi:10.1007/BF01215922.

Fully Scalable MPC Algorithms for WSPD in Euclidean Spaces

Eunjin Oh[1] and Hyeonjun Shin[2]

1 **Pohang University of Science and Technology, Korea**
`eunjin.oh@postech.ac.kr`
2 **Pohang University of Science and Technology, Korea**
`hyeonjun.shin@postech.ac.kr`

Abstract

In this paper, we study the problem of constructing a $(1/\varepsilon)$-*well-separated-pair-decomposition* (WSPD) for a point set of size n in d-dimensional Euclidean space for a constant d, in the Massively Parallel Computation (MPC) model, where multiple machines work in parallel and communicate in synchronous rounds. We present an $O(1)$-round MPC algorithm that constructs a $O(1/\varepsilon)$-WSPD of size $(1/\varepsilon)^{O(d)}n$. This improves the best-known algorithm [FOCS'93] for computing a WSPD which requires $O(\log n)$ rounds. As a consequence, the following problems can be solved in $O(1)$ rounds in the MPC model: computing a $(1+\varepsilon)$-spanner, a $(1-\varepsilon)$-approximation of the diameter, the closest pair.

1 Introduction

In this paper, we study geometric proximity problems in the *Massively Parallel Computation* (MPC) model, now a standard framework for large-scale data processing in distributed systems such as MapReduce [16], Dryad [25], Hadoop [31], and Spark [32]. Here, the input data is distributed across multiple machines, and computation proceeds in synchronous rounds of local computation and communication. A central algorithmic goal is to minimize the number of communication rounds and to use small local memory while keeping the total work nearly linear. Over the past decade, significant progress has been made on fundamental MPC problems such as sorting, connectivity, and matching [5, 6, 8, 11, 14, 18, 20, 23, 28, 30]. While some geometric problems such as clustering, Euclidean MST, and spanners have been studied [3, 4, 7, 12, 15, 19, 26], they represent only a limited portion of the rich landscape of computational geometry. In particular, only a few prior works consider fundamental proximity structures such as well-separated pair decompositions (WSPD), which are central to many classic geometric algorithms.

Our contribution. In this paper, we present the first $O(1)$-round, fully scalable MPC algorithm for computing a $(1/\varepsilon)$-WSPD of size $(1/\varepsilon)^{O(d)}n$ in d-dimensional Euclidean spaces for a constant $d \geq 1$. This algorithm uses $(1/\varepsilon)^{O(d)}n$ total space and $O(n^\delta)$ local space per machine for any $\delta \in (0,1)$. Furthermore, as applications, we present fully scalable MPC algorithms for constructing a $(1+\varepsilon)$-spanner of size $(1/\varepsilon)^{O(d)}$, computing $(1-\varepsilon)$-approximation of the diameter, the closest pair, which take $O(1)$ rounds, and uses the same local space and total space.

Comparison with prior work. Due to the importance of the WSPD, computing a WSPD was among the earliest topics studied in parallel computational geometry. Callahan and Kosaraju [9] gave an optimal PRAM algorithm that computes an $O((1/\varepsilon)^d n)$-sized $(1/\varepsilon)$-WSPD of n points in d-dimensional Euclidean space in $O(\log n)$ depth using $O(n)$ processors.

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

Algorithm 1 genWSPD(u, v)

1: Assume $\mathsf{diam}(\square_u) \leq \mathsf{diam}(\square_v)$. Otherwise, exchange the roles of u and v.
2: **if** $\max\{\mathsf{diam}(\square_u), \mathsf{diam}(\square_v)\} \leq (\varepsilon/8) \cdot \mathsf{dist}(p_u, p_v)$ **then**
3: **Return** $\{u, v\}$
4: **else**
5: Denote the children of v by $v_1, \ldots v_i$.
6: **Return** $\bigcup_{j=1}^{i} \mathsf{genWSPD}(u, v_j)$

This yields an $O(\log n)$-round MPC algorithm via standard simulation [23, 27]. Our result improves upon this by reducing the round complexity from $O(\log n)$ to $O(1)$.

As the applications are fundamental in computational geometry, several results in the MPC model are known.

- **Spanner:** t-spanners with $O(n^{1+1/t^2})$ edges can be constructed in $O(1)$ rounds [13, 19], as explicitly stated in [26]. In high-dimensional Euclidean spaces, they outperform ours in terms of round complexity, whereas our spanner has fewer edges in low dimensions.
- **Diameter:** A $(1-\varepsilon)$-approximation of the diameter follows from evaluating the maximum directional width over a $(1/\varepsilon)^{O(d)}$ net on the unit sphere [2], which can be implemented in $O(1)$ rounds in the MPC model.
- **Closest pair:** The closest pair problem can be solved in $\mathbb{R}^d$ with $d \leq 2$ in $O(1)$ rounds by constructing the Delaunay triangulation [21, 29]. Our result improves upon this by presenting a more general algorithm that applies to arbitrary dimensions.

2 Preliminaries

For any two points p and q, we denote their Euclidean distance by $\mathsf{dist}(p, q)$. We assume that one can compute $\mathsf{dist}(p, q)$ in $O(1)$ time. For any two point set X, Y, we define their distance as the minimum distance between any pair of points $x \in X$ and $y \in Y$. That is, $\mathsf{dist}(X, Y) = \min_{x \in X, y \in Y} \mathsf{dist}(x, y)$. When $X = \{x\}$ is a singleton, we alternatively use $\mathsf{dist}(x, Y)$ to denote $\mathsf{dist}(X, Y)$. We denote the *diameter* of P, defined as the maximum pairwise distance between the points in P by $\mathsf{diam}(P) = \max_{p,q \in P} \mathsf{dist}(p, q)$.

2.1 Well-Separated Pair Decomposition.

We say that two sets A, B of points are $(1/\varepsilon)$-*separated* if $\max\{\mathsf{diam}(A), \mathsf{diam}(B)\} \leq \varepsilon \cdot \mathsf{dist}(A, B)$ for $\varepsilon > 0$. A collection $\mathcal{W} = \{\{A_1, B_1\}, \ldots, \{A_k, B_k\}\}$ is called a $(1/\varepsilon)$-*well-separated pair decomposition* ($(1/\varepsilon)$-WSPD) of a point set P if

- $A_i, B_i \subset P$, and A_i and B_i are $(1/\varepsilon)$-separated for every i,
- $A_i \cap B_i = \emptyset$ for every i, and
- $\bigcup_{i=1}^{s} A_i \otimes B_i = P \otimes P$, where $A \otimes B = \{(x, y) \mid x \in A, y \in B, x \neq y\}$.

In $\mathbb{R}^d$, any point set of size n has a $(1/\varepsilon)$-well-separated pair decomposition consisting of $(1/\varepsilon)^{O(d)} n$ pairs [10]. We define the *size* of the WSPD as the number of pairs it contains. While Euclidean spaces have a WSPD of linear size, the sum of the sizes of all pairs of points in the WSPD can be quadratic. To avoid explicitly listing all points in each pair, we will use a compact representation for a point set: a representative point and a bounding region.

Construction. One can compute a $(1/\varepsilon)$-WSPD efficiently using a *compressed quadtree* [24], where each node u corresponds to a cell (hypercube) $\square_u$. Let P be a set of n points. Consider the axis-aligned unit hypercube $\square$ containing P. The unit hypercube can be hierarchically partitioned into smaller axis-aligned hypercubes, where each cell is recursively subdivided into 2^d cells of half the side length until every cell contains at most one point in P. The *(regular) quadtree* of P is a tree T whose nodes correspond to a cell of the hierarchical partition of $\square$, and whose non-leaves have 2^d children, satisfying the following properties. Let $\square_v$ denote the corresponding cell of a node v of T. The cells corresponding to the children of a non-leaf u form a subdivision of $\square_u$ into cells of half the side length. Each leaf of T corresponds to a cell containing at most one point in P. Throughout this paper, if $\square_v$ of a leaf v contains a point $p \in P$, we consider $\square_v$ as p itself. To represent the points of P contained in $\square_u$, we associate each node u with a representative point, denoted by p_u, defined as an arbitrary point of P contained in $\square_u$.

Note that the height of the quadtree can be unbounded in the worst case if the spread of P is not bounded. Thus, instead of a quadtree, we need to use a *compressed quadtree*, which is a compact representation of the quadtree. Intuitively, the compressed quadtree of P is the tree whose nodes are the nodes in a quadtree of P whose corresponding cells contain at least one point in P, and no child is corresponding to a cell containing the same subset of P. More specifically, the compressed quadtree of P is obtained from the (regular) quadtree by removing all nodes whose cells are empty (i.e., contain no point of P) and contracting any node whose cell contains the same point set as its only child. Observe that each leaf of the compressed quadtree corresponds to a unique point in P.

Algorithm 1 describes how to compute a WSPD using a compressed quadtree T. In particular, we run Algorithm 1 with a pair consisting of two copies of the root of T. This algorithm always returns a $(1/\varepsilon)$-WSPD of size $(1/\varepsilon)^{O(d)}n$. More specifically, the algorithm constructs a $(1/\varepsilon)$-WSPD by recursively checking whether two nodes satisfy the following, starting from the root node paired with itself.

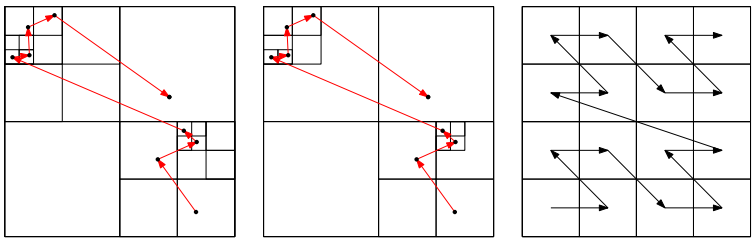
$$\max\{\mathsf{diam}(\square_u), \mathsf{diam}(\square_v)\} \leq (\varepsilon/8) \cdot \mathsf{dist}(p_u, p_v). \tag{1}$$

If this holds, returns the pair $\{u, v\}$ as a pair of the WSPD; otherwise, assuming without loss of generality that $\mathsf{diam}(\square_u) \leq \mathsf{diam}(\square_v)$, it recursively refines the pair by replacing v with its children: for each child v' of v, the algorithm recurses on the pair $\{u, v'\}$.

2.2 Massively Parallel Computation (MPC) Model

In the MPC model, an input of size n is arbitrarily distributed over m machines, each with a local memory of $O(s)$ words. We assume that a word can store a real number with arbitrary precision as in [1], which is a standard assumption for geometric problems. The number of machines is typically $m = O(n/s)$ or slightly larger. We say that an MPC algorithm is *fully scalable* if it works for $s = n^\delta$ with an arbitrary constant $\delta \in (0, 1)$. We assume that each machine has a unique ID from $[m]$. The computation of an MPC algorithm proceeds in *synchronous rounds*, each consisting of a *local computation phase* followed by a *communication phase*. In the computation phase, each machine performs local computations on its local data. In the communication phase, each machine sends messages to other machines. Each machine is allowed to send and receive a total of $O(s)$ words per round.

Throughout this paper, we use the following subroutines that can be implemented in $O(1)$ MPC rounds: sorting, predecessor, indexing, minimum/maximum, broadcast [5, 17, 22, 23]. In particular, we can *sort* the elements stored across consecutive machines in $O(1)$ rounds.



(a) (b) (c)

Figure 1 (a) Illustration of the quadtree. (b) Illustration of the compressed quadtree. In (a) and (b), the ordering of the points colored red is the $\mathcal{Z}$-order. (c) Illustration of the $\mathcal{Q}$-order among the cells of the same side length.

We can *broadcast* a message of size $O(s^\alpha)$ to a set of consecutive machines in $O(1)$ rounds for a constant $\alpha < 1$.

3 WSPD Construction in the MPC model

Let P be a set of n points in $\mathbb{R}^d$. We compute a WSPD in two steps: first, by constructing a compressed quadtree, and then by constructing a WSPD using this tree.

Key idea: Using $\mathcal{Q}$-order and $\mathcal{Z}$-order. To construct the compressed quadtree, we make use of a total ordering of cells of T and points of P; see [24]. Consider a (regular) quadtree T of P, and its depth-first search (DFS) traversal, where the children of each node are always visited in the same pre-defined order. The *$\mathcal{Q}$-order* of T is a total ordering of the cells corresponding to nodes in T according to the DFS order of T. See Figure 1(c). Notice that the $\mathcal{Q}$-order on the compressed quadtree is inherited directly from its underlying (regular) quadtree. Moreover, the total ordering of P obtained by restricting the $\mathcal{Q}$-order to the points of P is known as the *$\mathcal{Z}$-order* of P. See Figure 1(a-b).

As in [24], we assume that both the floor operation and the operation on two real numbers that computes the index of the first bit at which they differ in their binary representations can be performed in constant time. Under this assumption, both $\mathcal{Q}$- and $\mathcal{Z}$-order comparisons can be performed in $O(1)$ time, using only the coordinates of the centers of cells for the $\mathcal{Q}$-order and point coordinates for the $\mathcal{Z}$-order. Similarly, for any two points in P, the smallest cell in the compressed quadtree that contains both can be computed in constant time using only their coordinates. Furthermore, given a point and a side length, we can compute in constant time the cell in the quadtree of that side length containing the point.

MPC construction the compressed quadtree. We first generate all nodes of the compressed quadtree T, and then identify their parents and children. We begin by sorting all points in P according to the $\mathcal{Z}$-order. This can be done efficiently in parallel, as $\mathcal{Z}$-order comparisons rely only on the coordinates of the points. Next, for each pair of consecutive points under the $\mathcal{Z}$-order, we construct the smallest cell containing both points. Each such cell corresponds to a non-leaf of T. Then, we have all nodes in T since each point corresponds to a unique leaf. Subsequently, we compute the parent and children for each node in T. In particular, for each $1 \leq i \leq 2^d$, we sort the cells, so that the cell corresponding to a node and that of its i-th child under the DFS order appear consecutively. This can be done by modifying the pre-defined order among siblings, which defines the $\mathcal{Q}$-order. Consequently, this allows us

to associate each node with its i-th child in parallel. We repeat this for all $1 \le i \le 2^d$ to associate every node with its children.

Constructing a WSPD. To construct a WSPD, we use the following observation from [24]: For a pair $(u, v) \in \mathcal{W}$ constructed by Algorithm 1 such that the recursion on $\{u, v\}$ is invoked from the recursion on $\{u, \mathsf{par}(v)\}$, $\mathsf{dist}(\square_{\mathsf{par}(u)}, \square_{\mathsf{par}(v)}) \le (1/\varepsilon) \cdot \mathsf{diam}(\square_{\mathsf{par}(v)})$, and $\mathsf{diam}(\square_{\mathsf{par}(v)}) \le \mathsf{diam}(\square_{\mathsf{par}(u)})$. Here, we refer to a node $u \in T$ as a *pairing candidate* of a node $v \in T$ if u satisfies the above bounds. Then, we identify all pairing candidates for each node in T. To do this, we construct auxiliary cells for each node v whose side length equals that of $\square_{\mathsf{par}(v)}$, which align with the hierarchical partition defining T and lie within distance $(1/\varepsilon) \cdot \mathsf{diam}(\square_{\mathsf{par}(v)})$ from $\square_{\mathsf{par}(v)}$. We then sort all auxiliary cells and all nodes in the $\mathcal{Q}$-order of their corresponding cells, breaking ties by letting the auxiliary cell precede the node corresponding to the identical cell. Observe that the successor node of an auxiliary cell of v or its children may be a pairing candidate for the children of v. Thus, we can identify all pairing candidates for each node in parallel. It concludes the following theorem.

▶ **Theorem 3.1.** *Given a point set P of size n in $\mathbb{R}^d$ for any constant $d \ge 1$ and for any $\varepsilon > 0$, one can construct a $(1/\varepsilon)$-WSPD of P of size $(1/\varepsilon)^{O(d)}n$ in $O(1)$ rounds, using $(1/\varepsilon)^{O(d)}n$ total space and $O(n^\delta)$ local space per machine for any $\delta \in (0, 1)$.*

4 Applications of WSPD

In this section, we present three applications: fully scalable MPC algorithms for constructing a spanner, computing the closest pair, approximating the diameter. These results are immediate consequences of our MPC construction of WSPD [24].

To construct a $(1+\varepsilon)$-spanner, we first construct a (c/ε)-WSPD $\mathcal{W}$ with $c \ge 16$. Then, a $(1+\varepsilon)$-spanner can be obtained by forming a graph by adding edges between the representative points of each pair of $\mathcal{W}$. Therefore, we only need to compute the distance between the representative points of each pair of $\mathcal{W}$. This step requires only local computation, as all pairs in the $\mathcal{W}$ are explicitly stored together with their representative points.

▶ **Theorem 4.1.** *Let P be a set of n points in $\mathbb{R}^d$ for constant d, and let $\varepsilon > 0$. One can construct a $(1+\varepsilon)$-spanner of P with $(1/\varepsilon)^{O(d)}n$ edges in $O(1)$ rounds, using $(1/\varepsilon)^{O(d)}n$ total space and $O(n^\delta)$ local space per machine for any $\delta \in (0, 1)$.*

For $(1-\varepsilon)$-approximate diameter, we only need to compute the maximum distance between the representative points in each pair of a $(4/\varepsilon)$-WSPD. Recall that this maximum can be computed in $O(1)$ rounds.

▶ **Theorem 4.2.** *Let P be a set of n points in $\mathbb{R}^d$ for constant d, and let $\varepsilon > 0$. One can compute $(1-\varepsilon)$-approximate diameter of P in $O(1)$ rounds, using $(1/\varepsilon)^{O(d)}n$ total space and $O(n^\delta)$ local space per machine for any $\delta \in (0, 1)$.*

Similarly, we can find the exact closest pair. We first construct a c-WSPD $\mathcal{W}$ for a fixed constant $c \ge 1$, and compute the minimum distance between the representative points among all pairs of $\mathcal{W}$.

▶ **Theorem 4.3.** *Let P be a set of n points in $\mathbb{R}^d$ for constant d. One can compute the closest pair in $O(1)$ rounds, using $O(n)$ total space and $O(n^\delta)$ local space per machine for any $\delta \in (0, 1)$.*

References

1 Pankaj K. Agarwal, Kyle Fox, Kamesh Munagala, and Abhinandan Nath. Parallel algorithms for constructing range and nearest-neighbor searching data structures. In *Proceedings of the 35th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems (PODS 2016)*, pages 429–440, 2016.

2 Pankaj K. Agarwal, Jiří Matoušek, and Subhash Suri. Farthest neighbors, maximum spanning trees and related problems in higher dimensions. *Computational Geometry*, 1(4):189–201, 1992.

3 AmirMohsen Ahanchi, Alexandr Andoni, MohammadTaghi Hajiaghayi, Marina Knittel, and Peilin Zhong. Massively parallel tree embeddings for high dimensional spaces. In *Proceedings of the 35th ACM Symposium on Parallelism in Algorithms and Architectures (SPAA 2023)*, pages 77–88, 2023.

4 Alexandr Andoni, Aleksandar Nikolov, Krzysztof Onak, and Grigory Yaroslavtsev. Parallel algorithms for geometric graph problems. In *Proceedings of the 46th Annual ACM Symposium on Theory of Computing (STOC 2014)*, pages 574–583, 2014.

5 Alexandr Andoni, Zhao Song, Clifford Stein, Zhengyu Wang, and Peilin Zhong. Parallel graph connectivity in log diameter rounds. In *Proceedings of the 59th Annual Symposium on Foundations of Computer Science (FOCS 2018)*, pages 674–685, 2018.

6 Sepehr Assadi, MohammadHossein Bateni, Aaron Bernstein, Vahab Mirrokni, and Cliff Stein. Coresets meet EDCS: algorithms for matching and vertex cover on massive graphs. In *Proceedings of the 30th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2019)*, pages 1616–1635, 2019.

7 Amir Azarmehr, Soheil Behnezhad, Rajesh Jayaram, Jakub Łącki, Vahab Mirrokni, and Peilin Zhong. Massively parallel minimum spanning tree in general metric spaces. In *Proceedings of the 36th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2025)*, pages 143–174, 2025.

8 Alkida Balliu, Rustam Latypov, Yannic Maus, Dennis Olivetti, and Jara Uitto. Optimal deterministic massively parallel connectivity on forests. In *Proceedings of the 34th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2023)*, pages 2589–2631, 2023.

9 Paul B. Callahan. Optimal parallel all-nearest-neighbors using the well-separated pair decomposition. In *Proceedings of the 34th Annual Symposium on Foundations of Computer Science (FOCS 1993)*, pages 332–340, 1993.

10 Paul B. Callahan and S. Rao Kosaraju. A decomposition of multidimensional point sets with applications to k-nearest-neighbors and n-body potential fields. *Journal of the ACM (JACM)*, 42(1):67–90, 1995.

11 Yi-Jun Chang and Da Wei Zheng. Fully scalable massively parallel algorithms for embedded planar graphs. In *Proceedings of the 35th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2024)*, pages 4410–4450, 2024.

12 Xi Chen, Vincent Cohen-Addad, Rajesh Jayaram, Amit Levi, and Erik Waingarten. Streaming Euclidean MST to a constant factor. In *Proceedings of the 55th Annual ACM Symposium on Theory of Computing (STOC 2023)*, pages 156–169, 2023.

13 Vincent Cohen-Addad, Vahab Mirrokni, and Peilin Zhong. Massively parallel k-means clustering for perturbation resilient instances. In *Proceedings of the 39th International Conference on Machine Learning (ICML 2022)*, pages 4180–4201, 2022.

14 Sam Coy and Artur Czumaj. Deterministic massively parallel connectivity. In *Proceedings of the 54th Annual ACM Symposium on Theory of Computing (STOC 2022)*, pages 162–175, 2022.

15 Artur Czumaj, Guichen Gao, Shaofeng H.-C. Jiang, Robert Krauthgamer, and Pavel Veselỳ. Fully-scalable MPC algorithms for clustering in high dimension. In *Proceedings of the*

51st International Colloquium on Automata, Languages, and Programming (ICALP 2024), page 50, 2024.

16 Jeffrey Dean and Sanjay Ghemawat. Mapreduce: simplified data processing on large clusters. *Communications of the ACM*, 51(1):107–113, 2008.

17 Michael Dinitz and Yasamin Nazari. Massively parallel approximate distance sketches. In *Proceedings of the 23rd International Conference on Principles of Distributed Systems (OPODIS 2019)*, pages 35:1–35:17, 2019.

18 Michal Dory, Orr Fischer, Seri Khoury, and Dean Leitersdorf. Constant-round spanners and shortest paths in congested clique and MPC. In *Proceedings of the 40th ACM Symposium on Principles of Distributed Computing (PODC 2021)*, pages 223–233, 2021.

19 Alessandro Epasto, Mohammad Mahdian, Vahab Mirrokni, and Peilin Zhong. Massively parallel and dynamic algorithms for minimum size clustering. In *Proceedings of the 33rd Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2022)*, pages 1613–1660, 2022.

20 Mohsen Ghaffari, Themis Gouleakis, Christian Konrad, Slobodan Mitrović, and Ronitt Rubinfeld. Improved massively parallel computation algorithms for MIS, matching, and vertex cover. In *Proceedings of the 37th ACM Symposium on Principles of Distributed Computing (PODC 2018)*, pages 129–138, 2018.

21 Michael T. Goodrich. Randomized fully-scalable bsp techniques for multi-searching and convex hull construction. In *Proceedings of the 8th annual ACM-SIAM symposium on Discrete algorithms (SODA 1997)*, pages 767–776, 1997.

22 Michael T. Goodrich. Communication-efficient parallel sorting. *SIAM Journal on Computing*, 29(2):416–432, 1999.

23 Michael T. Goodrich, Nodari Sitchinava, and Qin Zhang. Sorting, searching, and simulation in the Mapreduce framework. In *Proceedings of the 22nd International Symposium on Algorithms and Computation (ISAAC 2011)*, pages 374–383, 2011.

24 Sariel Har-Peled. *Geometric approximation algorithms*. American Mathematical Soc., 2011.

25 Michael Isard, Mihai Budiu, Yuan Yu, Andrew Birrell, and Dennis Fetterly. Dryad: distributed data-parallel programs from sequential building blocks. In *Proceedings of the 2nd ACM SIGOPS/EuroSys European Conference on Computer Systems 2007*, pages 59–72, 2007.

26 Rajesh Jayaram, Vahab Mirrokni, Shyam Narayanan, and Peilin Zhong. Massively parallel algorithms for high-dimensional Euclidean minimum spanning tree. In *Proceedings of the 35th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2024)*, pages 3960–3996, 2024.

27 Howard Karloff, Siddharth Suri, and Sergei Vassilvitskii. A model of computation for Mapreduce. In *Proceedings of the 21st Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2010)*, pages 938–948, 2010.

28 Nathaniel Lahn, Sharath Raghvendra, and Kaiyi Zhang. A combinatorial algorithm for approximating the optimal transport in the parallel and MPC settings. In *Proceedings of the 37th Conference on Neural Information Processing Systems (NeurIPS 2023)*, pages 21675–21686, 2023.

29 Abhinandan Nath, Kyle Fox, Pankaj K. Agarwal, and Kamesh Munagala. Massively parallel algorithms for computing tin dems and contour trees for large terrains. In *Proceedings of the 24th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 1–10, 2016.

30 Krzysztof Nowicki and Krzysztof Onak. Dynamic graph algorithms with batch updates in the massively parallel computation model. In *Proceedings of the 32th ACM-SIAM Symposium on Discrete Algorithms (SODA 2021)*, pages 2939–2958, 2021.

31 Tom White. *Hadoop: The definitive guide*. O'Reilly Media, Inc., 2012.

32 Matei Zaharia, Mosharaf Chowdhury, Michael J. Franklin, Scott Shenker, and Ion Stoica. Spark: Cluster computing with working sets. In *Proceedings of the 2nd USENIX Workshop on Hot Topics in Cloud Computing (HotCloud 2010)*, pages 10:1–10:7, 2010.

Computing the Fréchet Distance When Just One Curve is c-Packed: A Simple Almost-Tight Algorithm

Jacobus Conradi[*1], Ivor van der Hoog[†2], Thijs van der Horst[3,4], and Tim Ophelders[‡4]

1 **BARC, University of Copenhagen, Denmark**
2 **Theoretical Computer Science, IT-University of Copenhagen, Denmark**
3 **Department of Information and Computing Sciences, Utrecht University, the Netherlands**
4 **Department of Mathematics and Computer Science, TU Eindhoven, the Netherlands**

Funding This work was supported by the the VILLUM Foundation grant (VIL37507) "Efficient Recomputations for Changeful Problems".

Abstract

We study approximating the continuous Fréchet distance of two curves with complexity n and m, under the assumption that only one of the two curves is c-packed. We show a simple technique to compute a $(1+\varepsilon)$-approximation in $\mathbb{R}^d$ in time $O(c\frac{n+m}{\varepsilon}\log\frac{n+m}{\varepsilon})$ when one of the curves is c-packed. Our approach is not only simpler than previous work, but also significantly improves the dependencies on c, ε, and d (which is only linear). Moreover, it almost matches the asymptotically tight bound for when both curves are c-packed. Our algorithm is robust in the sense that it does not require knowledge of c, nor information about which of the two input curves is c-packed.

Related Version *Full version*: `https://arxiv.org/abs/2508.10537`

1 Introduction

The Fréchet distance is a widely studied similarity measure for curves, with numerous real-world applications such as handwriting recognition [25], map-matching [28], comparing coastlines [23], time series clustering [14], and data analysis of outlines of shapes in geographic information systems [12]. Like the Hausdorff distance, the Fréchet distance is a bottleneck measure that outputs the distance between a pair of points from the two curves. However, unlike the Hausdorff distance, it respects the ordering of points along the curves, making it particularly well-suited for measuring similarity between moving data entries.

Already in 1995, Alt and Godau [1] presented a near-quadratic time algorithm for computing the Fréchet distance between two polygonal curves with n and m vertices, achieving a running time of $O(nm\log(n+m))$. Although there have since been incremental improvements [7, 9], strong evidence suggests that significantly faster algorithms are unlikely. In particular, Bringmann [3] showed that, assuming the *Strong Exponential Time Hypothesis*,

[*] Jacobus Conradi is funded by the iBehave Network: Sponsored by the Ministry of Culture and Science of the State of North Rhine-Westphalia and affiliated with the Lamarr Institute for Machine Learning and Artificial Intelligence.
[†] Ivor van der Hoog is supported by the VILLUM Foundation grant (VIL37507) "Efficient Recomputations for Changeful Problems".
[‡] Tim Ophelders is supported by the Dutch Research Council (NWO) under project no. VI.Veni.212.260.

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

no strongly subquadratic algorithm (i.e., with running time $O((nm)^{1-\delta})$ for any $\delta > 0$) exists. This lower bound also holds for algorithms that approximate the Fréchet distance within a factor less than 3, and even in one-dimensional settings [8].

However, significantly faster algorithms are possible for restricted classes of curves. For example, when the curves consist of sufficiently long edges relative to their Fréchet distance, Gudmundsson, Mirzanezhad, Mohades, and Wenk [18] gave a near-linear time exact algorithm. Additionally, several models have been proposed to capture the structure of "realistic" curves commonly found in applications. These include the notions of *κ-bounded* [2], *ϕ-low density* [24], and *c-packed* curves [13]. Of these three realism assumptions, c-packedness is the most frequently studied [4, 6, 11, 15, 16, 17, 19, 20, 21, 22].

In this work, we study approximating the Fréchet distance for c-packed curves. A polygonal curve is said to be c-packed, for some value $c \in \mathbb{R}$, if the total length of the curve inside any disk of radius r is at most cr. This notion, introduced by Driemel, Har-Peled, and Wenk [13], captures the geometric structure of many real-world trajectories. They showed that for two c-packed curves with n and m vertices, a $(1+\varepsilon)$-approximation of the Fréchet distance can be computed in $O\big(c\,(n+m)\cdot\big(\frac{1}{\varepsilon}+\log(n+m)\big)\big)$ time. The dependence on ε was later improved by Bringmann and Künnemann [4], who gave an algorithm running in $\tilde{O}(c\,(n+m)/\sqrt{\varepsilon})$ time, where the $\tilde{O}$ hides polylogarithmic factors in n and ε^{-1}. Their result is conditionally optimal, assuming SETH, for curves in $\mathbb{R}^d$ when $d \geq 5$ and $m \in \Theta(n)$ [3]. Gudmundsson, Sha, and Wong [20] empirically studied the packedness of curves from real-world data sets, finding that many of them are c-packed for small values of c, often with $c \ll n$. This supports the practical relevance of algorithms whose running time depends on c rather than on the input size directly.

When only one curve is c-packed. The algorithms of Driemel, Har-Peled, and Wenk [13] and Bringmann and Künnemann [4] assume that *both* input curves are c-packed. More recently, Gudmundsson, Mai, and Wong [17] extended this setting to the asymmetric case, in which only one of the two curves is c-packed. For a c-packed curve with n vertices and an arbitrary curve with m vertices in $\mathbb{R}^d$, they gave an algorithm with running time $\tilde{O}\Big(\big(c+\frac{1}{\varepsilon}\big)\Big(\frac{cn}{\varepsilon^2}+\frac{c^2m}{\varepsilon^7}+\frac{1}{\varepsilon^{2d+1}}\Big)\Big)$. While their approach is the most general to date, its running time has significantly worse dependencies on c, ε, and the dimension d compared to previous (less general) results. To obtain this runtime, their algorithm relies on several sophisticated techniques and data structures, including layered graphs over the parameter space of P and Q, as well as geometric range searching structures. These data structures form the bottleneck of their approach and are the source of their exponential dependency on the dimension d.

In this paper we give evidence that the asymptotic running time for the approximation of the Fréchet distance in this asymmetric setting should be the same as in the symmetric setting. We provide a strong structural theorem that bounds the complexity of the free-space between a c-packed and a non-c-packed curve. This strong characterisation of the free-space allows for a simple, straightforward and efficient approach for computing a $(1+\varepsilon)$-approximation of the continuous Fréchet distance in $O(c\,\frac{n+m}{\varepsilon}\log\frac{n+m}{\varepsilon})$, if at least one of P or Q is c-packed. This approach is not only more self-contained and simpler, it improves the dependency on c, ε, and the dimension d compared to [17].

Preliminaries. A d-dimensional (polygonal) *curve* P defined by an ordered set $(p_1,\ldots,p_n) \subset \mathbb{R}^d$ is a piecewise-linear function $P\colon [1,n] \to \mathbb{R}^d$, where for $t \in [i,i+1]$ it is defined by $P(t) = p_i + (t-i)(p_{i+1}-p_i)$. The points $p_1,\ldots,p_n$ are the *vertices* of P. The *edges* of P

are the directed line segments from p_i to p_{i+1}, for $i < n$. A curve P is said to be *c-packed* if, for every $r > 0$, the total length of P inside a ball of radius r is at most cr.

A *reparametrisation* of $[1, n]$ is a non-decreasing surjection $f\colon [0, 1] \to [1, n]$. Two reparametrisations f and g of $[1, n]$ and $[1, m]$, respectively, describe a *matching* (f, g) between two curves P and Q with n and m vertices, where for any $t \in [0, 1]$, the point $P(f(t))$ is matched to $Q(g(t))$. The (continuous) *Fréchet distance* $d_{\mathrm{F}}(P, Q)$ between P and Q is defined as

$$d_{\mathrm{F}}(P, Q) := \min_{(f,g)} \max_{t\in[0,1]} \|P(f(t)) - Q(g(t))\|,$$

where (f, g) range over all matchings between P and Q.

For computing the Fréchet distance one analyses the free space, a subset of the parameter space of the two given curves P and Q. The *parameter space* of curves P and Q with n and m vertices, respectively, is the rectangle $[1, n] \times [1, m]$ together with the regular grid whose cells are the squares $[i, i + 1] \times [j, j + 1]$ for integers i and j. A point (x, y) in the parameter space corresponds to the points $P(x)$ and $Q(y)$. The Δ-*free space* is the set of points $(x, y) \in [1, n] \times [1, m]$ such that $\|P(x) - Q(y)\| \leq \Delta$.

2 Technical overview

Alt and Godau [1] observed that the Δ-free space in every cell coincides with an ellipse intersected with that cell which can be computed in $O(1)$ time. Additionally, the Fréchet distance between two curves P and Q is at most Δ, if and only if a bimonotone path through the Δ-free space exists from $(1, 1)$ to (n, m). Thus, by exploring those cells of the parameter space that intersect the Δ-free space, one can easily decide whether the Fréchet distance between two curves is at most Δ.

The number of cells with a Δ-free point can be quadratic in general. Yet, for suitably simplified versions of the input curves, we prove in Theorem 3.4 the structural property that if at least one of the input curves is c-packed, the $(1+\varepsilon)\Delta$-free space has only $O\Big(c\,\frac{(1+\alpha)(n+m)}{\varepsilon}\Big)$ cells that contain a $(1 + \varepsilon)\Delta$-free point, where $\alpha = d_{\mathrm{F}}(P, Q)/\Delta$. This mirrors a key insight by Driemel, Har-Peled, and Wenk [13], but has two notable differences: (1) they assume that both curves are c-packed, and (2) their bound of $O(c\,\frac{n+m}{\varepsilon})$ does not depend on α. Our bounds asymptotically match when $\Delta = \Omega(d_{\mathrm{F}}(P, Q))$.

Driemel, Har-Peled, and Wenk [13] leveraged their structural insight to define an *approximate decider:* a procedure that, given an approximation parameter $\varepsilon > 0$ and a distance threshold Δ, determines whether $d_{\mathrm{F}}(P, Q) > \Delta$ or $d_{\mathrm{F}}(P, Q) \leq (1 + \varepsilon)\Delta$. They used this approximate decider within a binary search framework to compute an overall $(1 + \varepsilon)$-approximation. In contrast, the approximate decider implied by our bound is guaranteed to be efficient only when $\Delta \in \Omega(d_{\mathrm{F}}(P, Q))$.

Even though our approximate decider is potentially inefficient when Δ is small compared to $d_{\mathrm{F}}(P, Q)$, it can be used to efficiently compute a $(1 + \varepsilon)$-approximation of $d_{\mathrm{F}}(P, Q)$. In the full version of this paper, we show two techniques to achieve this. The first method relies on a polynomial over-estimate Δ^+ of $d_{\mathrm{F}}(P, Q)$ (i.e., a value Δ^+ such that $d_{\mathrm{F}}(P, Q) \leq \Delta^+ \leq (n+m)^{O(1)} d_{\mathrm{F}}(P, Q)$). Various methods exist to compute a such a polynomial over-estimate in near-linear time [10, 26, 27], but these methods are quite complex. For the sake of simplicity and self-containment, we additionally show a way to use a straightforward greedy algorithm by Bringmann and Mulzer [5], which computes an exponential over-estimate.

Our first method is careful and calls the approximate decider only with parameters Δ for which it is efficient; specifically, $\Delta \geq d_{\mathrm{F}}(P, Q)/2$. It starts with $\Delta = \Delta^+$, halving

Δ until the decider reports that $d_{\mathrm{F}}(P,Q) > \Delta$. This results in a range $[\Delta, 2(1+\varepsilon)\Delta]$ that contains $d_{\mathrm{F}}(P,Q)$, which, using a similar approach, can further be refined to obtain a $(1+\varepsilon)$-approximation. We note that if Δ^+ is a polynomial overestimation, i.e., $\Delta^+ \leq (n+m)^{O(1)} \cdot d_{\mathrm{F}}(P,Q)$, then the first part of this process makes only $O(\log(n+m))$ calls to our approximate decider. If we instead rely on the simpler to compute exponential over-estimation of $d_{\mathrm{F}}(P,Q)$ then this method is no longer efficient.

Our second method is a more careless approach, which may call the approximate decider with values Δ that are significantly less than $d_{\mathrm{F}}(P,Q)$ – resulting in an extremely slow decision. In particular, let $\Delta^+ \leq 2^{O(n+m)} d_{\mathrm{F}}(P,Q)$ and define $\Delta_i := \Delta^+/(1+\varepsilon)^i$. Starting with $i = 1$, repeatedly double i and call the decider until it reports that $d_{\mathrm{F}}(P,Q) > \Delta_i$. This will happen after only $O(\log\log_{1+\varepsilon} 2^{O(n+m)}) = O(\log \frac{n+m}{\varepsilon})$ calls to the decider. This approach returns a range $[\Delta_{2^i}, (1+\varepsilon)\Delta_{2^{i-1}}]$ that contains $d_{\mathrm{F}}(P,Q)$, which can further be refined to obtain a $(1+\varepsilon)$-approximation. The complication with this approach is that it may invoke our approximate decider with values Δ that are considerably smaller than $d_{\mathrm{F}}(P,Q)$ – thereby losing our guarantees on the complexity of the free space. To work around this, we create a *fallible approximate decider*. Given estimates (c', Δ) of $(c, d_{\mathrm{F}}(P,Q))$, this decider may report one of three things: (i) $d_{\mathrm{F}}(P,Q) > \Delta$, (ii) $d_{\mathrm{F}}(P,Q) \leq (1+\varepsilon)\Delta$, or (iii) *timeout*, indicating that the current estimates are too inaccurate. The timeout ensures that the fallible approximate decider has a running time of $O\big(c\frac{n+m}{\varepsilon}\big)$ (as long as $c' \in O(c)$) regardless of Δ. We use timeouts to guide updates to either c' or Δ, refining our estimates until we obtain a $(1+\varepsilon)$-approximation of $d_{\mathrm{F}}(P,Q)$.

Note that our decider requires not only an estimate of the Fréchet distance, but also an estimate of the minimum packedness of the curves. We can approximate the minimum packedness c of a curve to within a constant in near-linear time [21], but only with a randomized algorithm. Our algorithm is robust, in the sense that it does not require an initial over-estimate of the packedness of the curves. This additionally makes the algorithm more self-contained.

3 On simplifications and their free spaces

Driemel, Har-Peled and Wenk [13] define the following greedy simplification of a curve P:

▶ **Definition 3.1** (Greedy μ-simplifications)**.** Let $\mu > 0$. The *greedy μ-simplification* of a curve $P = (p_1, \ldots, p_n)$ is the unique maximal subsequence $S = (s_1, \ldots, s_m)$ of P such that $s_1 = p_1$, and each s_j is the first vertex of P after s_{j-1} with distance at least μ from s_{j-1}.

(Unlike [13], we do not necessarily include the last vertex of P. This does not affect their results.) This class of simplifications has some particularly useful properties. The simplifications are easily constructed in linear time. Additionally, Driemel *et al.* [13] show that the simplification retains (roughly) the packedness of the original curve, and is close to the original curve in terms of Fréchet distance. Specifically, the μ-simplification of P is $6c$-packed for every μ, and has Fréchet distance at most μ to P. Driemel *et al.* [13]'s primary result, which is central to existing algorithms for c-packed curves, is a structural lemma that bounds the complexity of the free space between the two simplified c-packed curves:

▶ **Lemma 3.2** (Lemma 4.4 in [13])**.** *Let P and Q be two c-packed curves with n and m vertices, respectively. Let $\Delta \geq 0$ and $\varepsilon \in (0,1)$, and let S_P and S_Q be the greedy $\frac{\varepsilon}{4}\Delta$-simplifications of P and Q, respectively. Then only $O(c\frac{n+m}{\varepsilon})$ parameter space cells of S_P and S_Q intersect $((1+\frac{\varepsilon}{2})\Delta)$-$\mathrm{FS}(S_P, S_Q)$.*

We extend Lemma 3.2 to the asymmetric setting, where only one curve is c-packed. To do so, we split the set of edges of Q into two sets. The first set contains *short* edges, with length at most $4d_{\mathrm{F}}(P,Q)$. The second set contains the remaining *long* edges, with length greater than $4d_{\mathrm{F}}(P,Q)$. Similarly to the edges, we call a cell $C_{i,j}$ of the parameter space of S and Q *short* if its corresponding edge $Q[j,j+1]$ on Q is short, and call $C_{i,j}$ *long* otherwise. The long edges of Q inherit the c-packedness from P (i.e., the total arc length of long edges in any disk of radius r is at most $O(cr)$):

▶ **Lemma 3.3.** *The set of edges of Q with length greater than $4d_{\mathrm{F}}(P,Q)$ is $6c$-packed.*

Proof sketch. Take an arbitrary ball $B(p,r)$ and let E denote the set of long edges of Q that intersect $B(p,r)$. Fix a matching between P and Q attaining the Fréchet distance. Every long edge e of Q is matched to a subcurve P_e of P of length greater than $2d_{\mathrm{F}}(P,Q)$. The length of e inside $B(p,r)$ is at most $2\|P_e \cap B(p, r+4d_{\mathrm{F}}(P,Q))\|$. Summed over all edges in E, we obtain $\|E \cap B(p,r)\| \leq 2\|P \cap B(p, r+4d_{\mathrm{F}}(P,Q))\| \leq 2c \cdot (r+4d_{\mathrm{F}}(P,Q))$. If $r \geq 2d_{\mathrm{F}}(P,Q)$, it directly follows that $\|E \cap B(p,r)\| \leq 6cr$. If $r < 2d_{\mathrm{F}}(P,Q)$, then there are at most $3c$ edges in E, each of which contributes at most length $2r$ to $\|E \cap B(p,r)\|$. ◀

The results of [13] on the complexity of the free space when both input curves are c-packed therefore extend to S and the long edges of Q. The short edges of Q, on the other hand, act as centers of small disks. These disks cannot be intersected by too many edges of S, as S is $6c$-packed. These properties imply the following bound on the complexity of the free space between S and Q:

▶ **Theorem 3.4.** *Let P and Q be curves with n and m vertices, respectively. Let $\Delta > 0$ and $\varepsilon \in (0,1]$, and let S_P and S_Q be the greedy $\frac{\varepsilon}{4}\Delta$-simplifications of P and Q, respectively. If one of P and Q is c-packed, then at most $\left(27c + 48c \cdot \frac{1+\alpha}{\varepsilon}\right)(n+m) = O\left(c\,\frac{(1+\alpha)(n+m)}{\varepsilon}\right)$ parameter space cells of S_P and S_Q intersect $((1+\frac{\varepsilon}{2})\Delta)$-FS$(S_P, S_Q)$, where $\alpha = d_{\mathrm{F}}(P,Q)/\Delta$.*

References

1 Helmut Alt and Michael Godau. Computing the Fréchet distance between two polygonal curves. *International Journal of Computational Geometry & Applications*, 5:75–91, 1995. doi:10.1142/S0218195995000064.

2 Helmut Alt, Christian Knauer, and Carola Wenk. Comparison of distance measures for planar curves. *Algorithmica*, 38(1):45–58, 2004. doi:10.1007/S00453-003-1042-5.

3 Karl Bringmann. Why walking the dog takes time: Fréchet distance has no strongly subquadratic algorithms unless SETH fails. In *Proc. 55th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 661–670, 2014. doi:10.1109/FOCS.2014.76.

4 Karl Bringmann and Marvin Künnemann. Improved approximation for Fréchet distance on c-packed curves matching conditional lower bounds. *International Journal of Computational Geometry & Applications*, 27(1-2):85–120, 2017. doi:10.1142/S0218195917600056.

5 Karl Bringmann and Wolfgang Mulzer. Approximability of the discrete Fréchet distance. *Journal of Computational Geometry*, 7(2):46–76, 2016. doi:10.20382/jocg.v7i2a4.

6 Frederik Brüning, Jacobus Conradi, and Anne Driemel. Faster approximate covering of subcurves under the Fréchet distance. In *Proc. 30th Annual European Symposium on Algorithms (ESA)*, pages 28–1, 2022. doi:10.4230/LIPIcs.ESA.2022.28.

7 Kevin Buchin, Maike Buchin, Wouter Meulemans, and Wolfgang Mulzer. Four soviets walk the dog: Improved bounds for computing the Fréchet distance. *Discrete & Computational Geometry*, 58(1):180–216, 2017. doi:10.1007/s00454-017-9878-7.

8 Kevin Buchin, Tim Ophelders, and Bettina Speckmann. SETH says: Weak Fréchet distance is faster, but only if it is continuous and in one dimension. In *Proc. 30th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 2887–2901, 2019. doi:10.1137/1.9781611975482.179.

9 Siu-Wing Cheng and Haoqiang Huang. Fréchet distance in subquadratic time. In *Proc. 2025 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 5100–5113, 2025. doi:10.1137/1.9781611978322.173.

10 Connor Colombe and Kyle Fox. Approximating the (continuous) Fréchet distance. In *Proc. 37th International Symposium on Computational Geometry (SoCG)*, pages 26:1–26:14, 2021. doi:10.4230/LIPIcs.SoCG.2021.26.

11 Jacobus Conradi, Anne Driemel, and Benedikt Kolbe. $(1+\varepsilon)$-ANN data structure for curves via subspaces of bounded doubling dimension. *Computing in Geometry and Topology*, 3(2):6–1, 2024. doi:10.57717/cgt.v3i2.45.

12 Thomas Devogele. A new merging process for data integration based on the discrete Fréchet distance. In *Advances in spatial data handling*, pages 167–181. Springer, 2002. doi:10.1007/978-3-642-56094-1_13.

13 Anne Driemel, Sariel Har-Peled, and Carola Wenk. Approximating the Fréchet distance for realistic curves in near linear time. *Discrete & Computational Geometry*, 48(1):94–127, 2012. doi:10.1007/s00454-012-9402-z.

14 Anne Driemel, Amer Krivosija, and Christian Sohler. Clustering time series under the Fréchet distance. In *Proc. Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 766–785, 2016. doi:10.1137/1.9781611974331.CH55.

15 Anne Driemel, Ivor van der Hoog, and Eva Rotenberg. On the discrete Fréchet distance in a graph. In *Proc. 38th International Symposium on Computational Geometry (SoCG)*, 2022. doi:10.4230/LIPIcs.SoCG.2022.36.

16 Joachim Gudmundsson, Zijin Huang, André van Renssen, and Sampson Wong. Computing a subtrajectory cluster from c-packed trajectories. In *Proc. 34th International Symposium on Algorithms and Computation (ISAAC)*, pages 34:1–34:15, 2023. doi:10.4230/LIPIcs.ISAAC.2023.34.

17 Joachim Gudmundsson, Tiancheng Mai, and Sampson Wong. Approximating the Fréchet distance when only one curve is c-packed. In *Proc. 35th International Symposium on Algorithms and Computation (ISAAC)*, pages 37:1–37:14, 2024. doi:10.4230/LIPIcs.ISAAC.2024.37.

18 Joachim Gudmundsson, Majid Mirzanezhad, Ali Mohades, and Carola Wenk. Fast Fréchet distance between curves with long edges. *International Journal of Computational Geometry & Applications*, 29(2):161–187, 2019. doi:10.1142/S0218195919500043.

19 Joachim Gudmundsson, Martin P. Seybold, and Sampson Wong. Map matching queries on realistic input graphs under the Fréchet distance. *ACM Transactions on Algorithms*, 20(2):14, 2024. doi:10.1145/3643683.

20 Joachim Gudmundsson, Yuan Sha, and Sampson Wong. Approximating the packedness of polygonal curves. *Computational Geometry*, 108:101920, 2023. doi:10.1016/J.COMGEO.2022.101920.

21 Sariel Har-Peled and Timothy Zhou. How packed is it, really?, 2025. URL: https://arxiv.org/abs/2105.10776, arXiv:2105.10776.

22 Ivor van der Hoog, Eva Rotenberg, and Sampson Wong. Data structures for approximate Fréchet distance for realistic curves. In *Proc. 35th International International Symposium on Algorithms and Computation (ISAAC)*, volume 322, 2024. doi:10.4230/LIPIcs.ISAAC.2024.56.

23 Ariane Mascret, Thomas Devogele, Iwan Le Berre, and Alain Hénaff. Coastline matching process based on the discrete Fréchet distance. In *Progress in Spatial Data Handling*, pages 383–400. Springer, 2006. doi:10.1007/3-540-35589-8_25.

24 Otfried Schwarzkopf and Jules Vleugels. Range searching in low-density environments. *Information Processing Letters*, 60(3):121–127, 1996. doi:10.1016/S0020-0190(96)00154-8.

25 E Sriraghavendra, K Karthik, and Chiranjib Bhattacharyya. Fréchet distance based approach for searching online handwritten documents. In *Proc. Ninth International Conference on Document Analysis and Recognition (ICDAR)*, volume 1, pages 461–465. IEEE, 2007. doi:10.1109/ICDAR.2007.4378752.

26 Thijs van der Horst and Tim Ophelders. Faster Fréchet distance approximation through truncated smoothing. In *Proc. 40th International Symposium on Computational Geometry (SoCG)*, pages 63:1–63:15, 2024. doi:10.4230/LIPICS.SOCG.2024.63.

27 Thijs van der Horst, Marc J. van Kreveld, Tim Ophelders, and Bettina Speckmann. A subquadratic n^{ε}-approximation for the continuous Fréchet distance. In *Proc. 2023 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1759–1776, 2023. doi:10.1137/1.9781611977554.CH67.

28 Carola Wenk, Randall Salas, and Dieter Pfoser. Addressing the need for map-matching speed: Localizing global curve-matching algorithms. In *Proc. 18th International Conference on Scientific and Statistical Database Management (SSDBM)*, pages 379–388, 2006. doi:10.1109/SSDBM.2006.11.

The number of occurrences of the two smallest distances

Cameron Strachan[1] and Konrad Swanepoel[1]

1 **Department of Mathematics, London School of Economics and Political Science**

Abstract

For each value of $\rho > 1$, let $e(n, \rho)$ denote the maximum combined number of occurrences of the smallest and second smallest distances in a set of n points in the plane, given that the smallest distance is 1 and the second smallest distance is ρ. We determine the asymptotics of $e(n, \rho) = f(\rho)n - o(n)$ for all except six values of ρ. For these exceptional values, we determine upper and lower bounds that are quite close. The hardest exceptional value is the golden ratio $\varphi = \frac{1+\sqrt{5}}{2}$, where we give the bounds $53/13 \le f(\varphi) \le 141/34$, with the lower bound disproving a conjecture of Csizmadia. It turns out that for all except finitely many values of ρ, $f(\rho) \in \{3, 7/2\}$.

1 Introduction

In his seminal 1946 paper, Erdős [4] introduced a variety of combinatorial problems involving distances in the Euclidean plane. One of these, to find the maximum number of times that a unit distance can occur in a set of n points is still far from solved, with the best bound more than 40 years old [12]. Another, to find the minimum number of distinct distances in a set of n points is almost solved after a major breakthrough by Guth and Katz 15 years ago [7]. Two of the problems from [4] are more elementary and have been solved completely. The first, already solved in [4], is that the maximum distance in a set of n points can occur at most n times, which is tight. The second is to find the maximum number that the minimum distance can occur. Harborth [8] gave a complete solution to this problem, showing that the maximum number is $\lfloor 3n - \sqrt{12n-3} \rfloor$.

Throughout the years, various variations have been proposed by Erdős and others. Vesztergombi [16] showed that the second-largest distance occurs at most $\lfloor 3n/2 \rfloor$ times, which is tight. Brass [1] found the asymptotically tight bound of $24n/7$ for the maximum number of occurrences of the second-smallest distance. Furthermore, he showed that this bound is attained for only one value of $\rho = d_2/d_1$, where d_1 and d_2 are the smallest and second-smallest distance, respectively, namely $\rho = 2\cos 15°$. He used a linear programming approach for this value of the ratio ρ. For all other values, he showed an upper bound of $3n$, which is also asymptotically tight.

Vesztergombi [15] considered the sum of the number of smallest distances and second-smallest distances, and found the upper bound of $6n$, attained by the triangular lattice, where the ratio is $\rho = \sqrt{3}$. Csizmadia [3], similar to Brass [1], considered the finer problem of estimating this sum for a fixed value of ρ. We continue the work on this problem by finding asymptotically tight estimates for all values of ρ apart from 6 exceptional values, one of them being the golden ratio $\varphi = (1 + \sqrt{5})/2$. While the golden ratio is the hardest case among all ρ, we improve both the upper and lower bounds found by Csizmadia, thereby disproving a conjecture of his.

Further papers on large and small distances are [14, 5, 9, 11, 10, 13]. The book [2] also contains a survey of distance problems up to 2005.

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

1.1 Our results

▶ **Definition 1.1.** Let S be a set of n points in the plane, and denote by $m_1(S)$ and $m_2(S)$ the number of occurrences of the smallest distance d_1 and second smallest distance d_2 in S. For each $\rho > 1$, let

$$e(n,\rho) = \max_{|S|=n,\ d_2/d_1=\rho} m_1(S) + m_2(S),$$

and

$$f(\rho) = \sup_{n\in\mathbb{N}} \frac{e(n,\rho)}{n}.$$

For a fixed ρ, it is easily seen that $e(n,\rho) + e(m,\rho) \le e(n+m,\rho)$ for any $n, m \in \mathbb{N}$. So, the sequence $(e(n,\rho))_{n\in\mathbb{N}}$ is superadditive, which by Fekete's lemma [6] implies that

$$f(\rho) = \lim_{n\to\infty} \frac{e(n,\rho)}{n} \quad \text{for all } \rho > 1.$$

The main problem of this paper is to determine $f(\rho)$ for all values of $\rho > 1$. We first summarize previous work in our terminology. Vesztergombi [15] showed that $f(\rho) \le 6$ for all $\rho > 1$, and observed that the triangular lattice gives $f(\sqrt{3}) = 6$. Brass [1] observed that the second-smallest distance can occur $3n - O(\sqrt{n})$ times for any ρ, which implies the lower bound $3 \le f(\rho)$. Csizmadia [3] showed that $f(\rho) \le 4$ if $\rho \notin \{\varphi, \sqrt{3}, \psi\}$ where $\varphi := \frac{1+\sqrt{5}}{2}$ is the golden ratio and $\psi := 2\cos 15^\circ = (2\sin 15^\circ)^{-1} = 1.931\ldots$ is the ratio occurring in Vesztergombi's example of a set with $24n/7 - o(n)$ second-smallest distances. Csizmadia furthermore showed that $4 \le f(\varphi) \le 4.5$, and observed that Vesztergombi's example for the tight lower bound of the number of occurrences of the second-smallest distance when $\rho = \psi$ gives $f(\psi) \ge \frac{33}{7}$. He also observed that the square lattice, together with his upper bound of 4 gives $f(\sqrt{2}) = 4$. All of these previous results are summarized in Fig. 1.

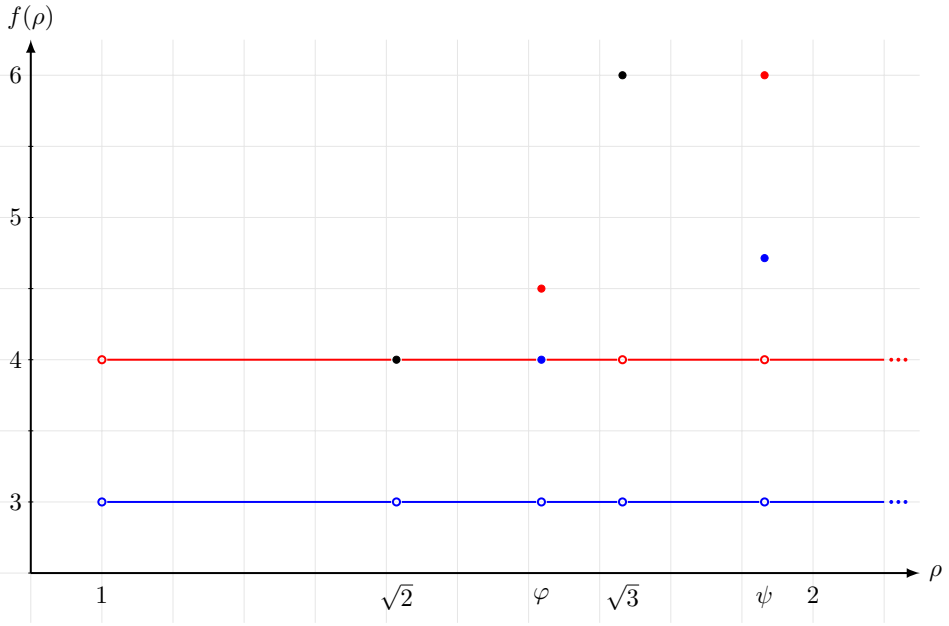


Figure 1 Previous results: lower bounds in blue, upper bounds in red, and known values in black

The main result of this paper is the determination of the exact value of $f(\rho)$ for each $\rho > 1$ except the six values φ and $2\cos\pi/k, k \in \{7,8,9,10,11\}$. For these exceptional values we improve the known bounds of $f(\rho)$. As already mentioned, the values of $f(\sqrt{2})$ and $f(\sqrt{3})$

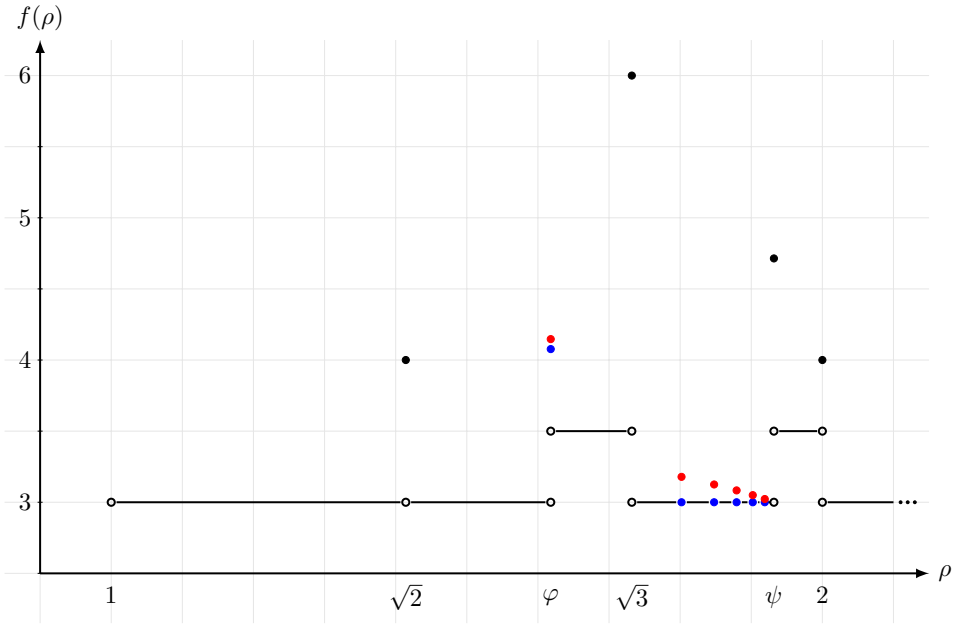


Figure 2 New results: exact values for $f(\rho)$ except for six values

are already known, and we present these values in our theorem for the sake of completeness.

▶ **Theorem 1.2.** *With $f(\rho)$ defined as in Definition 1.1, we have the following exact values.*

$$f(\rho) = \begin{cases} 3 & \rho \notin \{\sqrt{2}\} \cup [\varphi, \sqrt{3}] \cup [\psi, 2] \cup \{2\cos\frac{\pi}{k} : k \in \{7,8,9,10,11\}\} \\ 7/2 & \rho \in (\varphi, \sqrt{3}) \cup (\psi, 2) \\ 4 & \rho \in \{\sqrt{2}, 2\} \\ 33/7 & \rho = \psi \\ 6 & \rho = \sqrt{3} \end{cases}$$

Furthermore, $\frac{53}{13} \le f(\varphi) \le \frac{141}{34}$, and $3 \le f(2\cos\frac{\pi}{k}) \le 3 + \frac{12-k}{4k}$ for $k \in \{7,8,9,10,11\}$.

Our new lower bound $f(\varphi) \ge 53/13$ comes from the construction in Fig. 3, and disproves a conjecture of Csizmadia that $f(\varphi) = 4$.

Throughout the abstract, we use the following notational conventions. For any set of n points in the plane with smallest and second-smallest distance d_1 and d_2, respectively, we define a graph G with the n points as the vertex set and with edges drawn as straight-line segments between any two points at distance d_1 or d_2. We call edges of length d_1 *short* and edges with length d_2 *long*. We denote the number of edges by e. In the remainder of the paper, we assume by scaling that $d_1 = 1$ and $d_2 = \rho > 1$.

1.2 Our methods

The proof of Theorem 1.2 for different values of ρ ranges from straightforward to fiendishly hard.

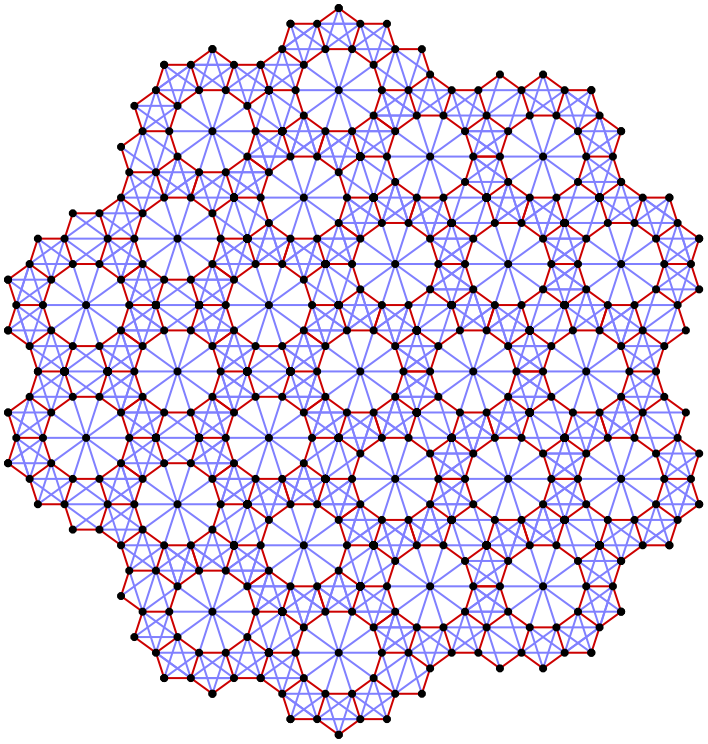


Figure 3 The construction for $\rho = \varphi$

If $\rho > 2$ it is quite easy to see that G is planar, hence $f(\rho) = 3$. For $\rho \leq 2$ we go through a case analysis of when two edges can cross depending on the six distances between the four vertices incident with the crossing edges. From this we obtain that certain crossings only occur in specific ranges of ρ. In particular, this shows that G is planar for all $\rho \notin \{\sqrt{2}\} \cup [\varphi, 2]$, and hence $f(\rho) = 3$ when $\rho \notin \{\sqrt{2}\} \cup [\varphi, 2]$. (As already mentioned, $f(\sqrt{2}) = 4$.)

After this, the unknown values for $f(\rho)$ are in the non-trivial interval $\rho \in [\varphi, 2]$, which naturally split into the subintervals $(\varphi, \sqrt{3})$, $(\sqrt{3}, \psi)$, and $(\psi, 2)$, with endpoints $\varphi, \sqrt{3}, \psi, 2$.

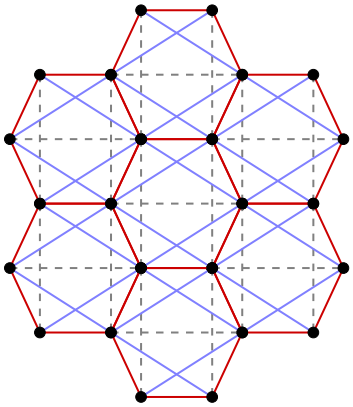


Figure 4 The construction of $f(\rho) \geq 3.5$ when $\rho \in (\varphi, \sqrt{3})$

When $\rho \in (\varphi, \sqrt{3})$ we give a construction for the lower bound $f(\rho) \geq 3.5$ in Fig. 4 by tiling the plane with centrally symmetric hexagons (each vertex has a degree of 7). For the upper bound, each time two edges cross, the four vertices incident with these edges must form a quadrilateral with three sides of unit length and one side with a length greater than ρ (which implies that there is no edge in our graph connecting these two vertices). We redraw one of the edges that cross by deleting it and drawing an edge between the two vertices with no edge between them in the quadrilateral. We show that this redrawing process does not create any new crossings in our graph. It may create double edges between vertices, but these double edges will form a matching in our graph, implying that there are at most $n/2$ of them. By deleting these $n/2$ double edges we obtain a planar graph, which implies the number of edges the graph had before any edges were deleted is at most $7n/2$ and hence $f(\rho) \leq 3.5$.

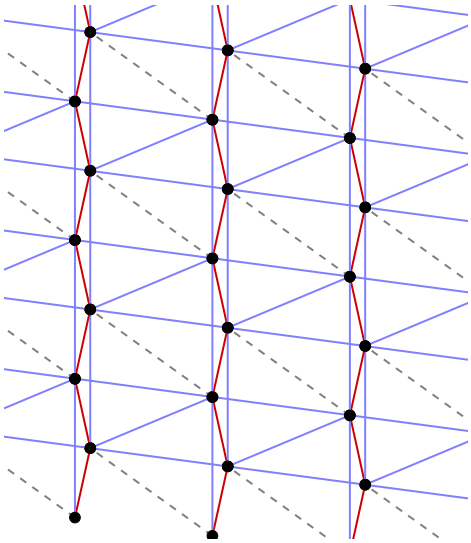


Figure 5 $f(\rho) \geq 3.5$ for $\rho \in (\psi, 2)$

For the interval $(\psi, 2)$ we give the construction for the lower bound in Fig. 5, which is quite different from the one for the interval $(\varphi, \sqrt{3})$ (each vertex again has a degree of 7). We show the upper bound by a somewhat involved charging argument. First, we show that the maximum degree of G is at most 8, and then we show that each vertex of degree 8 has either at least two neighbours of degree 6, or at least one neighbour of degree at most 5. We conclude the proof by showing that each of these low-degree neighbours is a low-degree neighbour of at most two vertices of degree 8. With these statements, the charging argument is as follows. Give each vertex a charge equal to its degree, the total charge in the graph is equal to two times the number of edges. For each vertex with degree 8, if the vertex is adjacent to a degree at most 5 vertex, give a charge of 1 from the degree 8 vertex to the degree at most 5 vertex. If instead the degree 8 vertex is adjacent to two degree 6 vertices, give a charge of 1/2 from the degree 8 vertex to each of the two degree 6 vertices. This process decreases each degree 8 vertex's charge by 1. Since a vertex receives a charge from at most two degree 8 vertices, the charge gained by degree at most 5 vertices is at most 2 and the charge gained by degree 6 vertices is at most 1. So each vertex has a charge of at most 7, and since no charge was lost throughout this process, this implies the total charge we started with is at most $7n$ which implies our graph has at most $3.5n$ edges and hence $f(\rho) \leq 3.5$.

The interval that is hardest to deal with is $(\sqrt{3}, \psi)$, where we show that $f(\rho) \leq 3$ apart from 5 possible exceptions, where we find weaker upper bounds. Here we again attempt to redraw G as a planar graph, but now there are multiple ways crossing edges can interact with each other which require different types of redrawing. This redrawing creates a planar graph, but again may have double edges. When a double edge is created we show that there is a quadrilateral face of G near this double edge that the double edge can be redrawn in. After redrawing in this way, G becomes a planar graph with no double edges and hence $f(\rho) \leq 3$.

This argument also works for the exceptional values in this interval, except in one special configuration that may occur in the graph for these values. Namely, when the ratio is $2\cos\frac{\pi}{k}$ the graph may have a regular k-gon made with short edges, where vertices that share a common neighbor are connected by an edge. This k-gon has 3 too many edges for it to be a triangulation of the k-gon. We can find quadrilaterals around this k-gon to redraw edges in. Unfortunately, we cannot show there are enough quadrilaterals to redraw these edges in to obtain a planar graph.

For the value $\rho = \psi$, Csizmadia found that Vesztergombi's construction gives the lower bound $f(\psi) \geq 33/7$, while the linear-programming approach of Brass [1] easily extends to our problem to show that $33/7$ is also an upper bound.

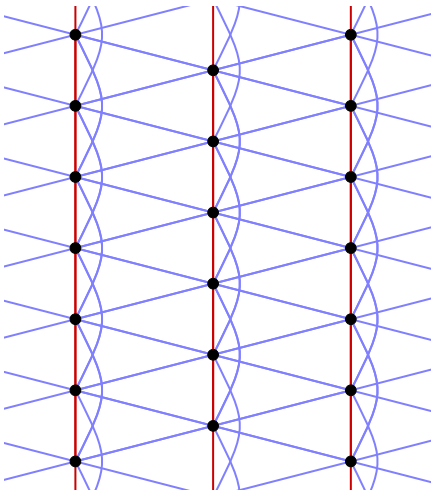


Figure 6 $f(2) \geq 4$

As noted above, $f(\sqrt{3}) = 6$. We show $f(2) = 4$ by giving a construction that matches Csizmadia's upper bound which is depicted in Fig. 6.

The remaining case $\rho = \varphi$ is the hardest. As already mentioned, the construction in Fig. 3 gives us $f(\varphi) \geq 53/13$, and we conjecture that the construction is asymptotically optimal, in other words, that $f(\varphi) = 53/13$. To find an upper bound, we extend the linear-programming approach, although here, after an extensive case analysis, we obtain an LP with 1999 variables and 5593 inequalities, which then gives the upper bound $f(\varphi) \leq 141/34 = 4.147\ldots$. We suspect that the linear-programming approach on its own can at best show $f(\varphi) \leq 45/11 = 4.0909\ldots$, which is slightly higher than the conjectured $f(\varphi) \leq 53/13 = 4.0769\ldots$. The reason why an LP on its own cannot close this gap is that this approach is local, and the extremizer for the value $45/11$ is as in the centre of Fig. 3, where we have a regular decagon surrounded by 10 regular pentagons. It seems that such a pattern cannot be repeated, so we believe that the true optimum is $53/13$.

References

1 Peter Brass. The maximum number of second smallest distances in finite planar sets. *Discrete Comput. Geom.*, 7(4):371–379, 1992. doi:10.1007/BF02187849.

2 Peter Brass, William Moser, and János Pach. *Research problems in discrete geometry*. Springer, New York, 2005.

3 György Csizmadia. The multiplicity of the two smallest distances among points. *Discrete Math.*, 194(1-3):67–86, 1999. doi:10.1016/S0012-365X(98)00116-2.

4 P. Erdős. On sets of distances of n points. *Amer. Math. Monthly*, 53:248–250, 1946. doi:10.2307/2305092.

5 P. Erdős, L. Lovász, and K. Vesztergombi. On the graph of large distances. *Discrete Comput. Geom.*, 4(6):541–549, 1989. doi:10.1007/BF02187746.

6 Angshuman R. Goswami. Revisiting Fekete's lemma, subadditive and periodic sequences. *Results Math.*, 80(5):Paper No. 131, 18, 2025. doi:10.1007/s00025-025-02452-z.

7 Larry Guth and Nets Hawk Katz. On the Erdős distinct distances problem in the plane. *Ann. of Math. (2)*, 181(1):155–190, 2015. doi:10.4007/annals.2015.181.1.2.

8 Heiko Harborth. Solution to problem 664 A. *Elem. Math*, 29:14–15, 1974.

9 Filip Morić. Small distances in convex polygons. *Discrete Math.*, 313(18):1767–1782, 2013. doi:10.1016/j.disc.2013.05.007.

10 Filip Morić and János Pach. Large simplices determined by finite point sets. *Beitr. Algebra Geom.*, 54(1):45–57, 2013. doi:10.1007/s13366-012-0126-2.

11 Filip Morić and David Pritchard. Counting large distances in convex polygons: a computational approach. In *Thirty essays on geometric graph theory*, pages 415–428. Springer, New York, 2013. URL: https://doi.org/10.1007/978-1-4614-0110-0_22, doi:10.1007/978-1-4614-0110-0_22.

12 J. Spencer, E. Szemerédi, and W. Trotter, Jr. Unit distances in the Euclidean plane. In *Graph theory and combinatorics (Cambridge, 1983)*, pages 293–303. Academic Press, London, 1984.

13 István Tomon. On the number of occurrences of the kth smallest distance between points in convex position. *Discrete Math.*, 338(6):990–993, 2015. doi:10.1016/j.disc.2015.01.019.

14 K. Vesztergombi. On the distribution of distances in finite sets in the plane. *Discrete Math.*, 57(1-2):129–145, 1985. doi:10.1016/0012-365X(85)90162-1.

15 K. Vesztergombi. Bounds on the number of small distances in a finite planar set. *Studia Sci. Math. Hungar.*, 22(1-4):95–101, 1987.

16 K. Vesztergombi. On large distances in planar sets. *Discrete Math.*, 67(2):191–198, 1987. doi:10.1016/0012-365X(87)90027-6.

Approximating the Minmax Three-Visiting Routes for m Treasures in a Simple Polygon*

Anna Brötzner[1], Bengt J. Nilsson[1], and Christiane Schmidt[2]

1 **Faculty of Technology and Society, Malmö University, Sweden**
{anna.brotzner, bengt.nilsson.TS}@mau.se
2 **Department of Science and Technology, Linköping University, Sweden**
christiane.schmidt@liu.se

Abstract

We develop an $O(m^7 n)$ time $(6+\varepsilon)$-approximation algorithm for the minmax three-visiting routes problem in a simple n vertex polygon, i.e., given three starting points and a collection of m point sized treasures in the polygon, compute three closed routes, each passing through its respective starting point, that jointly see all the treasures while minimizing the length of the longest route.

1 Introduction

We consider the following variant of the *minmax multiple visiting routes* problem: Find three closed routes that see m point sized treasures in a simple polygon, such that the length of the longest of them is minimized (minmax). Similar problems have been previously studied [2, 3]. For the shortest *visiting route* case, an appropriate modification of the algorithm of Tan and Jiang [7] obtains a shortest route that visits the visibility polygons of the treasures in $O(m^2 n)$ time for the boundary anchored version. This can be extended to an $O(m^3 n)$ time algorithm for the floating version using Tan's technique [5].Brötzner *et al.* [1] show how to modify the algorithm of Tan and Jiang [7] to obtain a shortest visiting route in $O(m^3 n)$ time when the route is anchored at an arbitrary point in the polygon.

The *minmax two-visiting routes* problem can be shown to be NP-hard by modifying the result by Mitchell and Wynters [4]. Recently Brötzner *et al.* [1] provided an $O(m^5 n)$ time $(2+\varepsilon)$-approximation algorithm, for any fixed $\varepsilon > 0$. Here, we develop a $(6+\varepsilon)$-approximation algorithm for the *minmax three-visiting routes* problem running in $O(m^7 n)$ time.

2 Preliminaries

Let $\mathbf{P}$ be a simple polygon with n vertices labeled in clockwise order, let s_1, s_2, and s_3 be three points in $\mathbf{P}$, not necessarily on the boundary of $\mathbf{P}$, the *starting points* of the visiting routes, and let $\mathcal{T} = \{p_1, \ldots, p_m\}$ be a set of points in $\mathbf{P}$, the *treasures*. We assume that no three points among the vertices, the starting points, and the points in $\mathcal{T}$ are collinear.

For two points p and q in $\mathbf{P}$, let $\Pi_{\mathbf{P}}(p, q)$ denote a *geodesic shortest path* between p and q, i.e., a path of minimum length connecting p and q that lies entirely within $\mathbf{P}$.

A point $p \in \mathbf{P}$ is *visible* to a point $q \in \mathbf{P}$ if the segment pq lies in $\mathbf{P}$. For a point $p \in \mathbf{P}$, let $\mathbf{V}(p)$ be the *visibility polygon* of p, the set of points in $\mathbf{P}$ that are visible to p. An edge

* All authors were supported by grant 2021-03810 (Illuminate: provably good algorithms for guarding problems) from the Swedish Research Council (Vetenskapsrådet). Anna Brötzner and Bengt J. Nilsson were additionally supported by grant 20250562 (Illuminate and Move: provably good algorithms for exploration problems) from the Crafoord Foundation.

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

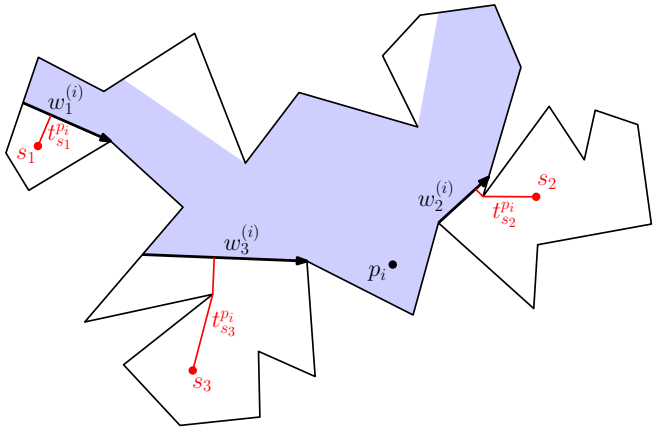


Figure 1 The windows of a treasure p_i. The visibility polygon $\mathbf{V}(p_i)$ is marked in blue.

of $\mathbf{V}(p)$ that is not contained in an edge of $\mathbf{P}$ is called a *window* of p, see Figure 1. Without loss of generality, we may assume that $s_1, s_2, s_3 \notin \mathbf{V}(p_i)$, for any $p_i \in \mathcal{T}$, as otherwise, these p_i are already seen from the starting points. Let $w_j^{(i)}$ be the window that separates the subpolygon of $\mathbf{P} \setminus \mathbf{V}(p_i)$ that contains s_j from $\mathbf{V}(p_i)$, for $j \in \{1, 2, 3\}$. An agent that starts walking at s_j, $j \in \{1, 2, 3\}$, sees point p_i as soon as it visits $w_j^{(i)}$.

For each $p_i \in \mathcal{T}$, we let $t_{s_j}^{p_i}$ be the shortest path from s_j to $\mathbf{V}(p_i)$, i.e., to the window $w_j^{(i)}$.

We assign a direction to each window $w_j^{(i)}$ as follows: if the polygon is rotated so that the window is vertical and $\mathbf{V}(p_i)$ lies locally to the left of $w_j^{(i)}$, then $w_j^{(i)}$ is directed upwards. We call the lower endpoint of $w_j^{(i)}$ the *bottom point* and the upper endpoint the *top point*.

Let W_j be the *shortest visiting route* for $\mathcal{T}$, i.e., the shortest route that sees all the points in $\mathcal{T}$ and passes through s_j. We can compute W_j in $O\big(|\mathcal{T}|^3 n\big)$ time [1].

We denote the minmax three-visiting routes for the points in $\mathcal{T}$ by (T_1, T_2, T_3), each T_j passing through s_j, $j \in \{1, 2, 3\}$. Without loss of generality we assume that each of T_1, T_2, and T_3 is as short as possible. Furthermore, we define $OPT \stackrel{\text{def}}{=} \max\big\{\|T_1\|, \|T_2\|, \|T_3\|\big\}$.

Finding the minmax three-visiting routes is a question of correctly partitioning the points in $\mathcal{T}$ between the three routes starting at each of the starting points. Once the three-partition is obtained, the routes can be computed in $O(|\mathcal{T}|^3 n)$ time [1].

3 The Algorithm

Fix (T_1, T_2, T_3) to be minmax three-visiting routes for the points in $\mathcal{T}$, inside the polygon $\mathbf{P}$.

▶ **Lemma 3.1.** *If one of T_1, T_2, or T_3 intersects the other two routes, then $\|W_j\| \leq 3OPT$, for $j \in \{1, 2, 3\}$.*

Proof. Pick any starting point s_j and follow a Eulerian trace around the subpaths of the three tours between intersection points until all subpaths have been traversed exactly once. Since each intersection point has even degree, the graph of the three tours is Eulerian and such a trace exists. Let R_j be the Eulerian trace, then

$$\|W_j\| \leq \|R_j\| \leq \|T_1\| + \|T_2\| + \|T_3\| \leq 3 \cdot \max\Big\{\|T_1\|, \|T_2\|, \|T_3\|\Big\} \;=\; 3OPT, \qquad (1)$$

proving our claim. ◀

We can thus return the minmax shortest of the triples $(W_1, s_2, s_3), (s_1, W_2, s_3)$, and (s_1, s_2, W_3).

Similarly, we have the next lemma.

▶ **Lemma 3.2.** *If T_1 intersects T_2, but T_3 does not intersect any of the other two, then there exist two routes U_1 and U_3, such that*

$$\max\left\{\|U_1\|, \|s_2\|, \|U_3\|\right\} \leq (4+\varepsilon)\,OPT, \tag{2}$$

for any $\varepsilon > 0$.

Proof. We apply the algorithm by Brötzner *et al.* [1] to obtain a $(2+\delta)$-approximate two-visiting route through s_1 and s_3 for any $\delta > 0$. This can be computed in $O(m^5 n)$ time.

The relative convex hull of the T_1 and T_2 together with T_3, form a feasible two-visiting route for the points in $\mathcal{T}$. Hence

$$\begin{aligned}\max\left\{\|U_1\|, \|U_3\|\right\} &\leq \big(2+\delta\big)\max\left\{\|T_1\| + \|T_2\|, \|T_3\|\right\} \\ &\leq \big(4+2\delta\big)\max\left\{\|T_1\|, \|T_2\|, \|T_3\|\right\} \;\leq\; \big(4+\varepsilon\big)\,OPT,\end{aligned} \tag{3}$$

by choosing $\delta \leq \varepsilon/2$. Thus, (U_1, s_2, U_3) is a $(4+\varepsilon)$ approximation as claimed. ◀

Similar lemmas hold if T_1 and T_2 are not intersected, respectively by the other two and we can return the minmax shortest of the triples (U_1, U_2, s_3), (U_1, s_2, U_3), and (s_1, U_2, U_3) as our approximation in these cases.

We henceforth assume that the three tours T_1, T_2, and T_3 do not intersect as we can obtain a worst case $(4+\varepsilon)$-approximation if they do. We will subdivide the set $\mathcal{T}$ into five subsets and show how to obtain routes that see the points in each subset separately.

- $p_i \in \mathcal{T}_=$, if $w_1^{(i)} = w_2^{(i)} = w_3^{(i)}$. We show how to see this subset of points in Section 3.1.
- $p_i \in \mathcal{T}_{12}$, if $w_1^{(i)} = w_2^{(i)} \neq w_3^{(i)}$, and we have two more symmetric sets $\mathcal{T}_{13}$ and $\mathcal{T}_{23}$, for when two windows are equal but the third is different. We show how to see these subsets of points in Section 3.2.
- $p_i \in \mathcal{T}_{\neq}$, if $w_1^{(i)} \neq w_2^{(i)} \neq w_3^{(i)} \neq w_1^{(i)}$. We show how to see this subset in Section 3.3.

We end this section with a structural lemma proved by Brötzner *et al.* [1] and that we will make extensive use of in the sequel. Let $\mathcal{P}$ be some discrete set of points in $\mathbf{P}$, let s and s' be starting points for two routes and assume that for each $p \in \mathcal{P}$, both t_s^p and $t_{s'}^p$ reach the same window w^p of $\mathbf{V}(p)$. Let p and p' be two distinct points in $\mathcal{P}$. Consider the set $\mathbf{P} \setminus \mathbf{V}(p)$ consisting of disjoint components. Let $\mathbf{P}_p$ be the component that contains both s and s' and let $\mathbf{P}_{p'}$ be defined analogously for p'. If $\mathbf{P}_p \subset \mathbf{P}_{p'}$, we say that p' *dominates* p, since a route through either s or s' that visits $\mathbf{V}(p')$ must also visit $\mathbf{V}(p)$. We call any point in $\mathcal{P}$ *dominating* if it is not dominated by any other point in $\mathcal{P}$. The dominating points can be established in $O\big(|\mathcal{P}|n\big)$ time [6] and we assume henceforth that $\mathcal{P}$ contains only dominating points. We label each window in the order of the first appearance of its bottom point along a clockwise traversal of the boundary of $\mathbf{P}$ starting at some arbitrary boundary point and order the associated points in $\mathcal{P}$ correspondingly. Let X_s and $X_{s'}$ be the two shortest routes that see all points in $\mathcal{P}$ and pass through the points s and s', respectively. X_s and $X_{s'}$ have the following structural property (see also Figure 2).

▶ **Lemma 3.3.** *Let $w_0, w_1, \ldots, w_{|\mathcal{P}|-1}$ be the windows of the points in $\mathcal{P}$, ordered as their bottom points appear along the boundary in clockwise order. If the two routes X_s and $X_{s'}$ do not intersect, then, if X_s visits w_i and $w_{i'-1}$, with $i+1 < i'-1$, then the route $X_{s'}$ cannot visit w_{i+1} and $w_{i'}$. (Implicitly assuming that indices are taken modulo $|\mathcal{P}|$.)*

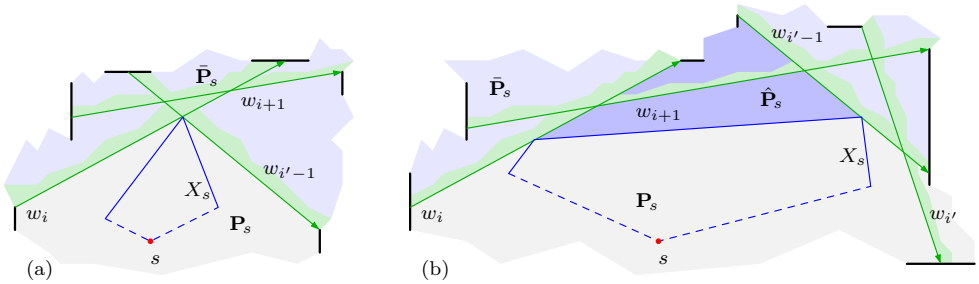


Figure 2 Illustrating Lemma 3.3.

3.1 Seeing the Points in $\mathcal{T}_=$

Let $(T_1^=, T_2^=, T_3^=)$ be minmax three-visiting routes for $\mathcal{T}_=$ in $\mathbf{P}$. Our objective in this section is to show how to compute these three routes.

The routes $(T_1^=, T_2^=, T_3^=)$ form non-intersecting cycles, and there are two possible configurations of such routes because of Lemma 3.3. One case, where each route sees a consecutive sequence of points according to the index order and one case where one route sees two consecutive sequences, while the other two routes see one consecutive sequence each. Consider a window that is visited by $T_j^=$ but where the preceding window is visited by a different route. This can occur at most four times for the three routes, and hence we can encode all possible route configurations using four indices i_1, i_2, i_3, and i_4, where this occurs interleaved with the route that visits the successive windows in order. They are:

1. i_1, $T_j^=$, i_2, $T_{j'}^=$, i_3, $T_{6-j-j'}^=$, and
2. i_1, $T_j^=$, i_2, $T_{j'}^=$, i_3, $T_j^=$, i_4, $T_{6-j-j'}^=$,

with $j \in \{1,2,3\}$, and $j' \in \{1,2,3\}\setminus\{j\}$, giving at most 12 different cases.

Selecting the minmax shortest triple of these, we obtain three routes $(T_1^=, T_2^=, T_3^=)$. There are $O\big(|\mathcal{T}_=|^4\big)$ index quadruples, and computing a shortest route takes $O\big(|\mathcal{T}_=|^3 n\big)$ for each of these quadruples [1]. Thus, the time complexity is bounded by $O\big(|\mathcal{T}_=|^7 n\big)$.

3.2 Seeing the Points in $\mathcal{T}_{12}$, $\mathcal{T}_{13}$, and $\mathcal{T}_{23}$

Our objective in this section is to show how to compute approximate routes of length $1+\gamma$, for any fixed $\gamma > 0$, such that they together see all the points in $\mathcal{T}_x$, for $x \in \{12, 13, 23\}$. We only show the computation for the set $\mathcal{T}_{12}$ as the other cases are completely symmetric.

Let $(T_1^{12}, T_2^{12}, T_3^{12})$ be fixed minmax three-visiting routes for $\mathcal{T}_{12}$ in $\mathbf{P}$ that do not intersect. Assume also that the points in $\mathcal{T}_{12}$ are ordered according to the bottom point order of the windows reached by $t_{s_1}^{p_i}$ and $t_{s_2}^{p_i}$, $p_i \in \mathcal{T}_{12}$. Since T_1^{12} and T_2^{12} do not intersect, there exist, by Lemma 3.3, two indices i and i' such that T_1^{12} and T_3^{12} together see the points $\mathcal{T}_{12}(i,i') = \{p_i, p_{i+1}, \ldots, p_{i'-1}\}$ whereas T_2^{12} and T_3^{12} together see the points $\mathcal{T}_{12}(i',i) = \{p_{i'}, p_{i'+1}, \ldots, p_{i-1}\}$, with $\mathcal{T}_{12}(i,i') \cup \mathcal{T}_{12}(i',i) = \mathcal{T}_{12}$ and assuming indices are taken modulo $|\mathcal{T}_{12}|$. Brötzner *et al.* [1] show how to compute a pair of routes (V_1', V_3') for $\mathcal{T}_{12}(i,i')$ in $O\big(|\mathcal{T}_{12}(i,i')|^3 n\big)$ time, where $\max\big\{\|V_1'\|, \|V_3'\|\big\} \leq (1+\gamma)\max\big\{\|T_1'\|, \|T_3'\|\big\}$, for any fixed $\gamma > 0$, and (T_1', T_3') is the minmax solution in this case. We apply their algorithm, for all index pairs $0 \leq i, i' \leq |\mathcal{T}_{12}| - 1$, to obtain the two pairs of routes (V_1^{12}, V_3^{12}) for $\mathcal{T}_{12}(i,i')$ and (V_2^{12}, V_3^{+}) for $\mathcal{T}_{12}(i',i)$, in $O(|\mathcal{T}_{12}|^5 n)$ time. The route V_1^{12} goes through s_1, V_2^{12} goes through s_2, and V_3^{12} and V_3^{+} go through s_3. The length of all the routes is bounded

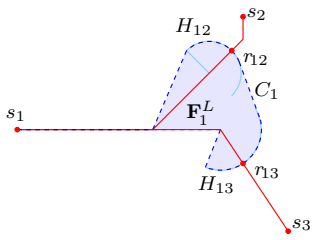


Figure 3 Defining the region $\mathbf{F}_1^L$.

by $\max\{\|V_1^{12}\|, \|V_2^{12}\|, \|V_3^{12}\|, \|V_3^{+}\|\} \leq (1+\gamma)\max\{\|T_1^{12}\|, \|T_2^{12}\|, \|T_3^{12}\|\} \leq (1+\gamma)OPT$. Thus, for each set $\mathcal{T}_x$, $x \in \{12, 13, 23\}$, the algorithm returns four routes.

3.3 Seeing the Points in $\mathcal{T}_{\neq}$

Let $(T_1^{\neq}, T_2^{\neq}, T_3^{\neq})$ be minmax three-visiting routes for $\mathcal{T}_{\neq}$ in $\mathbf{P}$. In this section, we show how to compute three routes $V_1^{\neq}$, $V_2^{\neq}$, and $V_3^{\neq}$ of lengths at most $(1+\gamma)\max\{\|T_1^{\neq}\|, \|T_2^{\neq}\|, \|T_3^{\neq}\|\}$, for arbitrary $\gamma > 0$, such that they together see all the points in $\mathcal{T}_{\neq}$.

Let $D = \min_{jj' \in \{12,13,23\}}\{\|\Pi_{\mathbf{P}}(s_j, s_{j'})\|\}$, let $L \leq D$ be some arbitrary distance value, and let r_{12} and r_{13} be the two points at distance L from s_1 on $\Pi_{\mathbf{P}}(s_1, s_2)$ and $\Pi_{\mathbf{P}}(s_1, s_3)$, respectively. Let H_{12} be the geodesic half circle of radius L constructed by tracing a geodesic quarter circle of radius L clockwise from r_{12} and a geodesic quarter circle of radius L counterclockwise from r_{12}, in both cases stopping the trace early if it reaches the polygon boundary. Construct the geodesic half circle H_{13} from r_{13} in the same way. Let F_1^L be the relative convex hull of s_1, H_{12} and H_{13}, and let $\mathbf{F}_1^L$ be the connected region bounded by F_1^L; see Figure 3. We define $\mathbf{F}_2^L$ from s_2 and $\mathbf{F}_3^L$ from s_3 similarly.

Let $B = \max_{p_i \in \mathcal{T}_{\neq}}\Big\{\min\{\|t_{s_1}^{p_i}\|, \|t_{s_2}^{p_i}\|, \|t_{s_3}^{p_i}\|\}\Big\}$ and let $R = \min\{6B, D\}$ and consider the regions $\mathbf{F}_j^R$, $j \in \{1, 2, 3\}$. The choice for the value of R is motivated by the following argument: No point $p \in \mathcal{T}_{\neq}$ can be seen by $T_j^{\neq}$ in an optimal solution, if

$$\|t_{s_j}^{p}\| > 6B = 6 \cdot \max_{p_i \in \mathcal{T}_{\neq}}\left\{\min\left\{\|t_{s_1}^{p_i}\|, \|t_{s_2}^{p_i}\|, \|t_{s_3}^{p_i}\|\right\}\right\}, \tag{4}$$

since the boundaries of the regions $\mathbf{F}_1^B$, $\mathbf{F}_2^B$, and $\mathbf{F}_3^B$, together see all points in $\mathcal{T}_{\neq}$ and each boundary forms a route that has length bounded by

$$\|\partial\mathbf{F}_j^B\| \leq (4+2\pi)B < 11B < 2(6B) < 2\|t_{s_j}^{p}\| \leq \max\left\{\|T_1^{\neq}\|, \|T_2^{\neq}\|, \|T_3^{\neq}\|\right\}, \tag{5}$$

i.e., strictly shorter than the optimum, a contradiction. If $\|t_{s_j}^{p}\| > D$, then $p \notin \mathcal{T}_{\neq}$, also a contradiction.

For any fixed $k \geq \lceil 94/\gamma \rceil$, we partition each of the regions $\mathbf{F}_j^R$, $j \in \{1, 2, 3\}$, into k^2 cells as follows. Let $\Pi_{\mathbf{P}}(s_1, r_0)$ and $\Pi_{\mathbf{P}}(s_1, r_k)$ be two paths of length exactly R in $\mathbf{F}_1^R$, $\Pi_{\mathbf{P}}(s_1, r_0)$ making left turns when it can and $\Pi_{\mathbf{P}}(s_1, r_k)$ making right turns when it can. Let $C_1 \stackrel{\text{def}}{=} \partial\mathbf{F}_1^R \setminus \big(\Pi_{\mathbf{P}}(s_1, r_0) \cup \Pi_{\mathbf{P}}(s_1, r_k)\big)$ and let $r_0, \ldots, r_k$ be equidistant points on C_1 with r_0 and r_k at the ends of C_1. Evidently, $\|C_1\| \leq (2\pi+2)R$. Let $\mathbf{F}_1^R(\kappa)$ be the subregion of $\mathbf{F}_1^R$ between $\Pi_{\mathbf{P}}(s_1, r_{\kappa-1})$ and $\Pi_{\mathbf{P}}(s_1, r_\kappa)$ and define the cells $\mathbf{c}_1^{\kappa,\kappa'} = \big\{p \in \mathbf{F}_1^R(\kappa) \,|\, (\kappa'-1)R/k < \|\Pi_{\mathbf{P}}(s_1, p)\| \leq \kappa' R/k\big\}$. We construct the cells for the regions $\mathbf{F}_2^R$ and $\mathbf{F}_3^R$ similarly. We say that the window $w_j^{(i)}$ *belongs to* cell $\mathbf{c}_j^{\kappa,\kappa'}$, if $t_{s_j}^{p_i}$ touches $w_j^{(i)}$ in $\mathbf{c}_j^{\kappa,\kappa'}$. Thus, each point

$p_i \in \mathcal{T}_{\neq}$ is associated to three cells, those that the three windows $w_1^{(i)}$, $w_2^{(i)}$, and $w_3^{(i)}$ belong to, giving us a similarity decomposition of the points in $\mathcal{T}_{\neq}$ into k^6 disjoint subsets. We pick one point from each non-empty subset as a representative for the subset and denote this set $\mathcal{R}$. We try all possible three-partitions of the points in $\mathcal{R}$ (at most 3^{k^6} combinations) to obtain a three-partition corresponding to optimal three-visiting routes $(T_1^{\mathcal{R}}, T_2^{\mathcal{R}}, T_3^{\mathcal{R}})$ for the points in $\mathcal{R}$. Hence, $\max\{\|T_1^{\mathcal{R}}\|, \|T_2^{\mathcal{R}}\|, \|T_3^{\mathcal{R}}\|\} \leq \max\{\|T_1^{\neq}\|, \|T_2^{\neq}\|, \|T_3^{\neq}\|\} \leq OPT$. Let $(V_1^{\neq}, V_2^{\neq}, V_3^{\neq})$ be the three routes obtained by partitioning the points $p_i \in \mathcal{T}_{\neq}$ in each subset according to the optimal three-partitioning of the representative point for the subset.

Each route $T_j^{\mathcal{R}}$ is contained in the region $\mathbf{F}_j^R$, by the choice of R, which means that it has at most one sequence of convex vertices. We extend each route $T_j^{\mathcal{R}}$ by moving the edges connected to convex vertices outward a distance of $(2\pi+2)\|T_j^{\mathcal{R}}\|/k + R/k$, connecting the vertices with circular arc pieces that sum to at most $(4\pi^2+4\pi)\|T_j^{\mathcal{R}}\|/k + 2\pi R/k$ and add two extra edges of length $(2\pi+2)\|T_j^{\mathcal{R}}\|/k + R/k$. Let this new route be $V_j^{\mathcal{R}}$. If $T_j^{\mathcal{R}}$ sees $p_i \in \mathcal{R}$ by intersecting $w_j^{(i)}$ belonging to $\mathbf{c}_j^{\kappa,\kappa'}$, then $V_j^{\mathcal{R}}$ visits all windows that belong to $\mathbf{c}_j^{\kappa,\kappa'}$ and sees all points in the subsets of the similarity decomposition of $\mathcal{T}_{\neq}$ that have windows belonging to $\mathbf{c}_j^{\kappa,\kappa'}$. $V_j^{\neq}$ is the shortest route for this partitioning, so $\|V_j^{\neq}\| \leq \|V_j^{\mathcal{R}}\|$. The three-visiting routes $(V_1^{\neq}, V_2^{\neq}, V_3^{\neq})$ can be computed in $O(|\mathcal{T}_{\neq}|^3 n)$ time, they see all points in $\mathcal{T}_{\neq}$, and have length

$$\begin{aligned}\|V_j^{\neq}\| \leq \|V_j^{\mathcal{R}}\| &\leq \|T_j^{\mathcal{R}}\| + 2\frac{(2\pi+2)\|T_j^{\mathcal{R}}\|}{k} + 2\frac{R}{k} + \frac{(4\pi^2+4\pi)\|T_j^{\mathcal{R}}\|}{k} + \frac{2\pi R}{k} \\ &\leq \left(1+\frac{94}{k}\right)\max\{\|T_1^{\neq}\|, \|T_2^{\neq}\|, \|T_3^{\neq}\|\} \leq (1+\gamma)OPT, \end{aligned} \tag{6}$$

since $k \geq \lceil 94/\gamma \rceil$.

3.4 Combining the Routes for a Complete Solution

Given the eighteen routes obtained for each of the subsets $\mathcal{T}_{=}$, $\mathcal{T}_{12}$, $\mathcal{T}_{13}$, $\mathcal{T}_{23}$, and $\mathcal{T}_{\neq}$ of $\mathcal{T}$, anchored at one of the three starting points, we combine them for each starting point s_j by

$$V_j^* = T_j^{=} \cup V_j^{12} \cup V_j^{13} \cup V_j^{23} \cup V_j^{+} \cup V_j^{\neq}, \tag{7}$$

and since $\|T_j^{=}\| \leq OPT$ and each $\|V_j^x\| \leq (1+\gamma)OPT$, for $x \in \{12, 13, 23, +, \neq\}$, for any fixed $\gamma > 0$, we have $\|V_j^*\| \leq (6+5\gamma)OPT \leq (6+\varepsilon)OPT$, if $\gamma \leq \varepsilon/5$.

We construct our final solution by combining the results from this section.

$$\begin{aligned}(V_1, V_2, V_3) = \arg\min\Big\{&\max\{\|W_1\|, \|s_2\|, \|s_3\|\}, \max\{\|s_1\|, \|W_2\|, \|s_3\|\}, \\ &\max\{\|s_1\|, \|s_2\|, \|W_3\|\}, \max\{\|U_1\|, \|U_2\|, \|s_3\|\}, \max\{\|U_1\|, \|s_2\|, \|U_3\|\}, \\ &\max\{\|s_1\|, \|U_2\|, \|U_3\|\}, \max\{\|V_1^*\|, \|V_2^*\|, \|V_3^*\|\}\Big\}. \end{aligned} \tag{8}$$

and obtain the following theorem.

▶ **Theorem 3.4.** *The three-visiting routes (V_1, V_2, V_3) see all points in $\mathcal{T}$, can be computed in $O(|\mathcal{T}|^7 n)$ time, and $\max\{\|V_1\|, \|V_2\|, \|V_3\|\} \leq (6+\varepsilon)OPT$, for any fixed $\varepsilon > 0$.*

References

1 Anna Brötzner, Bengt J. Nilsson, and Christiane Schmidt. Improved approximation of two watchmen's routes in simple polygons. Manuscript, 2026.

2 Svante Carlsson and Håkan Jonsson. Guarding a treasury. In *Proc. of the 5th Canadian Conference on Computational Geometry*, pages 85–90, 1993.

3 Linda L. Deneen and Shashikant Joshi. Treasures in an art gallery. In *Proc. of the 4th Canadian Conference on Computational Geometry*, pages 17–22, 1992.

4 Joseph S.B. Mitchell and Erik L. Wynters. Watchman routes for multiple guards. In *Proc. of the 3rd Canadian Conference on Computational Geometry*, pages 126–129, 1991.

5 Xuehou Tan. Fast computation of shortest watchman routes in simple polygons. *Inf. Process. Lett.*, 77(1):27–33, 2001. doi:10.1016/S0020-0190(00)00146-0.

6 Xuehou Tan. A linear-time 2-approximation algorithm for the watchman route problem for simple polygons. *Theoretical Computer Science*, 384(1):92–103, 2007. Theory and Applications of Models of Computation. doi:10.1016/j.tcs.2007.05.021.

7 Xuehou Tan and Bo Jiang. Efficient algorithms for touring a sequence of convex polygons and related problems. In *Proceedings of Theory and Applications of Models of Computation, TAMC 2017*, pages 614–627. Springer Verlag, Lecture Notes in Computer Science 10185, 2017. doi:doi.org/10.1007/978-3-319-5.

Point Set Transformations using Given Groups*

Thomas C. van Dijk[1], Erwin Glazenburg[2], Wouter Meulemans[†1], Anna Schenfisch[3], and Arjen Simons[†1]

1 **TU Eindhoven**
{t.c.v.dijk|w.meulemans|a.simons1}@tue.nl
2 **Utrecht University**
e.p.glazenburg@uu.nl
3 **KTH Royal Institute of Technology**
schenf@kth.se

Abstract

We study transforming one point set A to an equal-size point set B optimally, where translating any of a predefined set of *groups* of points incurs a cost independent of its size. We consider two objectives: minimizing the number of groups moved (Cardinality) and minimizing the total translation length of the groups (Length). When a matching between A and B is given – that is, we know which point moves to which other point – we prove that minimizing Cardinality is NP-hard in general, but efficiently solvable when the given groups form a hierarchy. Without a prescribed matching, we prove that both minimizing Cardinality and Length is NP-hard.

1 Introduction

Transforming one point set into another via rigid transformations is a widely studied problem with applications in e.g., computer vision, motion analysis, and robotics [2, 6, 7]. Recent work [8] studied how a given matching (bijection) between point sets can be represented using additive translations applied to subsets (*groups*) of points to capture the "visual complexity" of grouped motion; see Figure 1. This framework introduces two dimensions: how to measure the cost of a grouped translation, and constraints on which points can be grouped together. The considered cost measures are *Cardinality* – the number of groups used – and *Length* – the sum of Euclidean norms of translation vectors applied to groups. One of the constraints is the *Given* variant, where the groups that can be used are prescribed – this paper focuses on this setting with given groups.

* Research on the topic of this paper was initiated at the 9th Workshop on Applied Geometric Algorithms (AGA 2025) in Otterlo, The Netherlands, supported by the Dutch Research Council (NWO) under project no. VI.Veni.212.260.
† W. Meulemans and A. Simons are (partially) supported by the Dutch Research Council (NWO) under project number VI.Vidi.223.137.

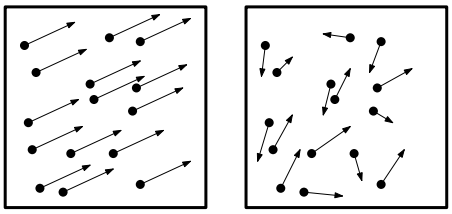


Figure 1 Grouped motion (left) is easier to understand than dispersed motion (right).

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

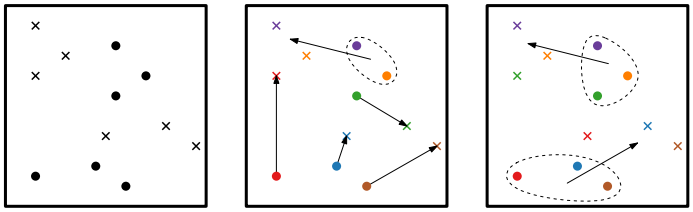


Figure 2 Left: source points A (dots) and target points B (crosses). Middle and Right: different matchings (indicated using colors) result in different grouped motions, with a lower cost on the right.

The above framework [8] considers only settings where a matching between point sets is given. Yet, many applications such as shape matching require determining both the matching and the translations [1, 2, 9]. We thus generalize the framework with another dimension: is the matching given or to be optimized? In the latter, we aim to find the matching minimizing the cost of a grouped translation, subject to constraints on the permitted groups; see Figure 2. This generalization extends the descriptive model of the original framework to a constructive alignment problem related to classical point-set registration and structured motion analysis.

Contributions and organization. In Section 3, we show that minimizing Cardinality for given groups is NP-hard, resolving an open problem from the original framework paper [8]. We also provide a polynomial-time algorithm when the given groups form a hierarchy, that is, any two given groups are either nested or disjoint. In Section 4 we study the previously unexplored problem where no matching is given, and establish NP-hardness for minimizing Cardinality, even if all singletons are included in the hierarchy, and for minimizing Length, even if the hierarchy has no nested groups.

Related work. For the matched case with given groups, Length can be minimized in (weakly) polynomial time and in linear time when the groups form a hierarchy [8].

Classic registration methods typically assume a single global transformation applied uniformly to all points [9], whereas our setting allows multiple translations tied to predefined groups, enabling motions that cannot be expressed in these classic registration models.

The special case of one group of all points together with a group per point effectively describes classical point-set matching. Minimizing the sum of squared distances is achieved by aligning the centroids [6], while other metrics, like Euclidean distance, are more complicated [4]. Additional work on matching point sets exists [1], differing in the matching distances used, allowed transformations, and whether the point sets have the same cardinality or not. Matching costs are typically based on the L_1 (Manhattan), L_2 (Euclidean), or L_2^2 metrics and take the sum of distances of the matched points, or some bottleneck measure. Such an approach treats points independently, whereas group-based translations couple the motion of points within a group, making such distance-based matchings incompatible with our setting. Other well-known metrics, such as the Earth Mover's distance [3], address weighted point sets, which we do not consider.

2 Preliminaries

Matched. The problem is introduced in [8] and equates to their Given constraint: we are given two point sets $A = \{a_1, \ldots, a_n\}$ and $B = \{b_1, \ldots, b_n\}$ in $\mathbb{R}^d$, with a matching $\phi : A \to B$, such that $\phi(a_i) = b_i$, and a family of subsets $\mathcal{F} \subseteq \mathcal{P}([n])$ of $[n] = \{1, \ldots, n\}$

indicating the groups via point indices. We refer to a group $G \in \mathcal{F}$ as a *singleton* group if $|G| = 1$. A solution to this problem consists of a pair $(\mathcal{F}', \tau)$, where $\mathcal{F}' \subseteq \mathcal{F}$ and $\tau : \mathcal{F}' \to \mathbb{R}^d$ describes the translation vectors for the groups defined by $\mathcal{F}'$. Let $\mathcal{F}'_i \subseteq \mathcal{F}'$ be only those groups that contain index i. A solution $(\mathcal{F}', \tau)$ is called *valid* if $\sum_{G \in \mathcal{F}'_i} \tau(G) = b_i - a_i$ for all $i \in [n]$, i.e., if the movement of each group places each point of A to its matched point in B. Note that a valid solution may not exist if $\mathcal{F}$ does not include a singleton group for all $i \in [n]$.

For the Cardinality measure, the goal is to minimize $|\mathcal{F}'|$, the number of groups that are translated; for Length, the goal is to minimize the total length translated by all groups, computed as $\sum_{G \in \mathcal{F}'} \|\tau(G)\|$, where $\|\cdot\|$ indicates the Euclidean norm in $\mathbb{R}^d$.

As only the relative positions between matched points are relevant, we can represent the input equivalently by a single collection of vectors $\Delta = \{\delta_1, \dots \delta_n\}$ with $\delta_i = b_i - a_i$.

We refer to the problems of optimizing Cardinality and Length in the matched case as *Matched Cardinality* and *Matched Length*, respectively.

Unmatched. We generalize [8] by supposing we must now find the matching $\phi : A \to B$ that minimizes the optimal solution for Matched Cardinality or Matched Length. Let Φ be the set of all matchings between A and B. Given two point sets $A, B \subset \mathbb{R}^d$ of equal size n and a family of allowed groups $\mathcal{F}$, we aim to find a matching $\phi^* \in \Phi$ that minimizes Length or Cardinality under the *matched* model described above. Note that each matching ϕ readily defines a collection Δ of difference vectors $\delta_i = \phi(a_i) - a_i$ that can be used as input for the matched problem.

We refer to the problems of finding an optimal matching for Cardinality and Length as *Unmatched Cardinality* and *Unmatched Length*, respectively.

3 Minimizing Matched Cardinality

We prove that, for a given matching, minimizing Cardinality is NP-hard for general $\mathcal{F}$, but polynomial-time solvable if $\mathcal{F}$ is hierarchical.

▶ **Theorem 1.** *Matched Cardinality is NP-hard in $\mathbb{R}^d$ for any $d \geq 1$, even if $\mathcal{F}$ includes a singleton group for all $i \in [n]$.*

Proof sketch. For $d = 1$, we use a reduction from Partition, which is weakly NP-hard [5]: given a set $N = \{x_1, \dots, x_n\}$ of positive integers, is there a subset $N' \subseteq N$ with $\sum_{x \in N'} x = k$, where $k = \frac{1}{2} \sum_{x \in N} x$? The result for higher d follows trivially.

For each $x_i \in N$, the reduction creates a pair of points in A and B with rightward difference vector δ_i of length x_i; additionally, a single pair with δ_{n+1} of length k to the right (see Figure 3). We set the permitted groups to be each i separately (except $n+1$), as well as the pairs $\{i, n+1\}$. If the Partition instance admits a solution, we achieve δ_{n+1} "for free", moving with Cardinality n. Conversely, if the Matched Cardinality instance permits using only n groups, δ_{n+1} must have been achieved for free via some subset of pairs: this subset in N hence sums to k. ◀

Consider the case where the family $\mathcal{F}$ is a hierarchy, that is, for any two groups $G, G' \in \mathcal{F}$, either $G \subseteq G'$, $G' \subseteq G$, or $G \cap G' = \emptyset$. We now represent $\mathcal{F}$ as a rooted tree T, where each node represents the sets induced by their subtree; see Figure 4.

First, consider the restricted case that $\mathcal{F}$ consists only of $[n]$ and all singleton groups. Let $\delta^* \in \Delta$ be the most frequently occurring distance vector. Then δ^* is an optimal translation vector for the root: it aligns the maximum number of leaves; the rest has to be

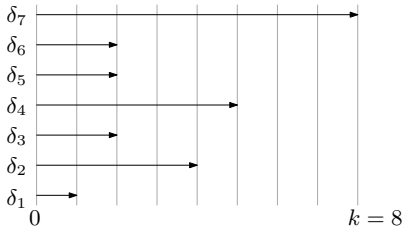


Figure 3 Example construction for $N = \{1, 4, 2, 5, 2, 2\}$, requiring a set N' such that $k = \sum_{x \in N'} x = 8$. Difference vectors $\delta_1, \ldots, \delta_6 \in \Delta$ represent the values in N, and $\delta_7 \in \Delta$ represents k.

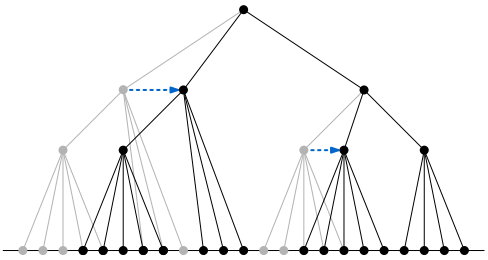


Figure 4 Translating groups in $\mathcal{F}$ (dashed arrows) corresponds to moving entire subtrees.

moved to their destination individually. Our algorithm generalizes this idea to arbitrary hierarchical $\mathcal{F}$. Note that a valid solution exists only if, for any leaf $u \in T$, all delta vectors associated with u are identical; this can be easily verified. In the algorithm described below, we assume that this condition holds and that all singleton groups are included in $\mathcal{F}$. If a leaf is not a singleton set, we can effectively treat it as one.

At every node u, we store a set of *preferred vectors* $V(u)$ defined as follows (see also Figure 5). If u is a leaf corresponding to index i, then $V(u) = \{\delta_i\}$. If u is an internal node, then $V(u)$ consists of the most frequent preferred vectors among its children (in case of a tie, it consists of all vectors with maximum frequency).

▶ **Lemma 2.** *In any constant dimension $d \geq 1$, the sets $V(u)$ for all nodes $u \in T$ can be computed in $O(n \log n + nh)$ time, for a tree of height h.*

Proof sketch. We compute $V(u)$ for every node u in a bottom-up fashion. At each node u, we look at the preferred vector sets $V(u_1), \ldots, V(u_k)$ of its children, count how often each vector occurs among those sets, and keep the ones with maximum frequency as $V(u)$. The work at node u is proportional to the sum of sizes of the preferred vector sets of its children.

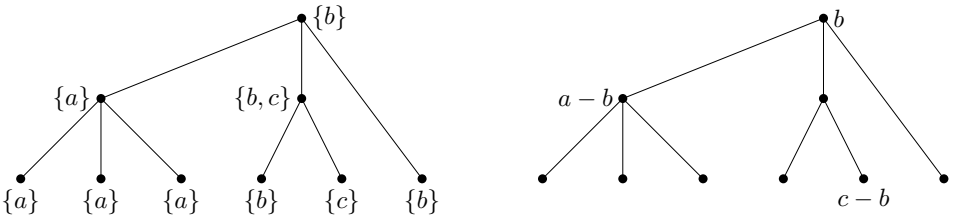


Figure 5 Example two-level hierarchy on six points with translation vectors indicated by letters. (left) Preferred vectors. (right) Minimum Cardinality solution.

Any vector in any $V(u)$ is in fact one of the original leaf vectors $\delta_i \in \Delta$, and each δ_i appears at most h times as it can only be part of $V(u)$ for nodes on the path from its leaf to the root. Therefore, a leaf's vector is processed at most h times across the whole tree, giving a time bound of $O(nh)$ if each occurrence can be handled in constant time. This can be achieved using $O(n \log n)$-time preprocessing: sort Δ, arbitrarily but so that duplicate values become adjacent, then assign unique integer identifiers in the range $[n]$ to the distinct δ_i. The required (multi)set operations can be implemented efficiently over this range. ◀

In an eventual solution, all points in the subtree rooted at $u \in T$ are translated according to all translations in the (strict) ancestors of u: we call the sum of these translations the *ancestral translation* of u. Moreover, we use *cost* of u to indicate the minimum number of group translations the subtree requires, given some ancestral translation. Our goal is thus to minimize the cost of the root (which necessarily has an ancestral translation of $\vec{0}$).

▶ **Lemma 3.** *For every node $u \in T$, the cost of u is minimum if its ancestral translation is a preferred vector in $V(u)$. Any other ancestral translation increases this cost by exactly 1.*

Proof. We use induction on the height of the subtree at u. For a leaf node, there is only a single preferred vector in $V(u)$: if the ancestral translation matches this vector, then the subtree is already solved and has cost 0; otherwise, the point needs to move to its target and the subtree has cost 1.

For an internal node, choosing a translation (or none) fixes the ancestral translation for its K children. Any choice can match the preferred vector of a certain number k of children. These k out of the K children receive their minimal cost; the other $K - k$ children have their minimal cost plus 1, by induction. Therefore, the total cost of the children is the sum of their minimum costs plus $K - k$. Clearly, this is minimized by maximizing k, that is, via an ancestral translation that matches a preferred vector in $V(u)$.

If the ancestral translation is not a preferred vector, we can readily apply a single translation at the root of a subtree to make it match a preferred translation: the cost is exactly one more than the minimum cost. ◀

The algorithm now simply computes all preferred vectors in a bottom-up fashion, and then chooses optimal translations in a top-down fashion. We obtain the following result.

▶ **Theorem 4.** *Let Δ be a collection of n vectors in $\mathbb{R}^d$, and let $\mathcal{F}$ be a given hierarchy represented by a tree T with height h. We can compute a solution $(\mathcal{F}', \tau)$ of minimal Cardinality in $O(n \log n + nh)$ time, for any constant dimension $d \geq 1$.*

Proof. By Lemma 2 computing the $V(u)$ vectors for all nodes $u \in T$ takes $O(n \log n + nh)$ time. We recursively compute the translations to apply via a pre-order traversal of T, such that the ancestral translation is known when treating a node. Let node $u \in T$ have an ancestral translation a. If $a \in V(u)$, we apply no translation; otherwise, we select an arbitrary vector $x \in V(u)$ and apply $x - a$. This yields minimal cost by Lemma 3. This recursion runs in $O(nh)$ time, as the total number of preferred vectors on any height is $O(n)$. ◀

4 Computing an optimal matching

We prove that, given two point sets without a matching and a family of allowed groups, computing an optimal matching is NP-hard for the Cardinality measure, even when all singleton groups are part of $\mathcal{F}$. For the Length measure, computing an optimal matching is also NP-hard, though we omit the restriction on including all singleton groups.

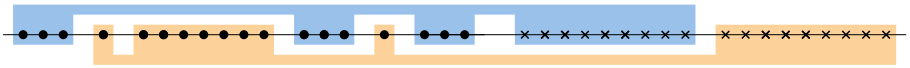


Figure 6 Example construction for Theorem 5 using the set $N = \{3, 1, 7, 3, 1, 3\}$. The points of A (dots) are grouped by these values. We place points of B (crosses) into groups of size $t = 9$. A matching minimizing Cardinality is indicated by the colored regions, and corresponds to a 3-partition.

▶ **Theorem 5.** *Unmatched Cardinality is NP-hard in $\mathbb{R}^d$ for any $d \geq 1$, even if $\mathcal{F}$ includes a singleton group for all $i \in [n]$.*

Proof sketch. For $d = 1$, we reduce from 3-Partition, which is strongly NP-hard [5]: given a set $N = \{x_1, \ldots, x_{3n}\}$ of $3n$ positive integers with $t = \frac{1}{n}\sum_{i=1}^{3n} x_i$, such that all $t/4 < x_i < t/2$, can N be partitioned into triples such that each triple sums to t? The result for higher d follows trivially.

For each $i \in [3n]$, we define a set A_i of x_i points, with unit distance spacing; each A_i is placed at distance 2 from A_{i-1}. See Figure 6. We use $A = \bigcup_i A_i$ and family $\mathcal{F}$ of permitted groups consists of all A_i, as well as all singletons. For each $i \in [n]$, we define a set B_i of t points, with unit distance spacing; each B_i is placed at distance 2 from B_{i-1}. We use $B = \bigcup_i B_i$, and ensure they are placed disjoint from all points in A.

If the 3-Partition instance admits a solution, then each triple $\{x_i, x_j, x_k\}$ in the solution is matched to a unique set B_x and we can move the sets A_i, A_j, A_k to match exactly to the points in B_x. This uses $3n$ groups in total.

A solution to the Unmatched Cardinality instance with $3n$ groups must move exactly all A_i groups, as each point in A must move. Each group A_i must move to a single B_x group, due to the spacing between the groups in B, and each point in A maps to a unique point in B. As $t/4 < x_i < t/2$, this implies that each B_x is covered by exactly three A_i sets. ◀

▶ **Theorem 6.** *Unmatched Length is NP-hard in $\mathbb{R}^d$ for any $d \geq 1$, even if $\mathcal{F}$ consists of pairwise disjoint groups.*

Proof sketch. We use a similar reduction as in Theorem 5, but do not include singleton groups in $\mathcal{F}$. As such, the 3-Partition instance admits a solution if and only if there exists a matching that results in a valid solution for the Matched Length problem. ◀

5 Future work

The framework of [8] considered several variants of the matched problem in which the groups are not given as part of the input, under different permitted-groups constraints. We have additional preliminary results indicating that most unmatched variants are NP-hard, making the study of approximability or parameterized complexity a natural direction of future work.

References

1 Helmut Alt and Leonidas J Guibas. Discrete geometric shapes: Matching, interpolation, and approximation. In *Handbook of computational geometry*, pages 121–153. Elsevier, 2000.

2 K. S. Arun, T. S. Huang, and S. D. Blostein. Least-squares fitting of two 3-D point sets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9(5):698–700, 1987.

3 Scott Cohen and L Guibas. The Earth Mover's distance under transformation sets. In *Proceedings of the 7th IEEE International Conference on Computer Vision*, volume 2, pages 1076–1083, 1999.

4 David Eppstein, Marc van Kreveld, Bettina Speckmann, and Frank Staals. Improved grid map layout by point set matching. *International Journal of Computational Geometry & Applications*, 25(02):101–122, 2015.

5 Michael R. Garey and David S. Johnson. *Computers and Intractability; A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., USA, 1990.

6 Berthold Horn. Closed-form solution of absolute orientation using unit quaternions. *Journal of the Optical Society A*, 4:629–642, 1987.

7 Liang Li, Ming Yang, and Chunxiang Wang. Graph correspondence-based point set registration. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 54(7):4101–4112, 2024.

8 Wouter Meulemans, Arjen Simons, and Kevin Verbeek. Visual Complexity of Point Set Mappings. In *Proceedings of SOFSEM 2025: Theory and Practice of Computer Science*, LNCS 15539, pages 157–171, 2025.

9 Andriy Myronenko and Xubo Song. Point set registration: Coherent point drift. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(12):2262–2275, 2010.

On (Directed) Width-Parameters of Geometric Spanners

Kevin Buchin[1], Carolin Rehs[1,2], and Torben Scheele[1]

1 **TU Dortmund, Germany**
{kevin.buchin,carolin.rehs,torben.scheele}@tu-dortmund.de
2 **TU Eindhoven, The Netherlands**

Abstract

Given a set of points P in Euclidean space, a geometric t-spanner G is a graph on P such that for every pair of points, the shortest path in G between those points is at most a factor t longer than the Euclidean distance between those points. The value $t \geq 1$ is called the dilation of G.

It was recently shown that there always exists an $O(n/k^{d/(d-1)})$-spanner of tree-width at most k for sets of n points $\mathsf{P} \subset \mathbb{R}^d$. In this paper, we strengthen this result qualitatively by extending it to the parameter path-width. We further argue that this result implies the same upper bound on the dilation for spanners of bounded cut-width and that a different previously known result on plane spanners of bounded tree-width implies a dilation bound for plane spanners of bounded clique-width.

A directed $O(n/k^{d/(d-1)})$-spanner of directed tree-width k can easily be obtained by replacing every edge of the undirected spanner by two opposing arcs. Complementing this, we show that the previously known lower bound of $\Omega(n/k^{d/(d-1)})$ on the dilation of tree-width k bounded spanners extends to directed tree-width for directed geometric spanners.

1 Introduction

While there has been research on geometric spanners for multiple decades now (see [3, 14] for surveys) the topic of geometric spanners with bounded width parameters has been studied only quite recently. Graphs with bounded parameters, for example *tree-width* [15, 16], often admit efficient algorithms for in general NP-hard problems.

Given a set of points $\mathsf{P} \subset \mathbb{R}^d$ a geometric spanner G with dilation t is a graph on P s.t. for every pair of points $p, q \in \mathsf{P}$ it holds that $d_G(p, q) \leq t \cdot \|pq\|$, where $\|pq\|$ is the Euclidean distance between p and q and $d_G(p, q)$ is the sum of the distances along the edges of a shortest path from p to q in G. A directed geometric spanner is defined analogously for digraphs D.

In [5] it was show that, for sets $\mathsf{P} \subset \mathbb{R}^d$ with n points, there always exists an $O(n/k^{d/(d-1)})$-spanner of tree-width at most k. We first look into stronger parameters than tree-width, specifically at the path-width. We show that the upper bound of [5] is extendible to path-width. Since every graph of bounded path-width is also a graph of bounded tree-width, the lower bound also extends trivially. Being at least as large as path-width, one would assume that for the cut-width, which is an even more restrictive parameter, it may be harder to find a spanner with small dilation. However, we show that the same dilation bounds hold for the cut-width as well. Also the clique-width, which admits less algorithmic properties than the tree-width, with the additional property of planarity is subject to the same bounds.

Recently, also directed spanners received some attention [4]. Regarding directed spanners of bounded directed tree-width, the upper dilation bound of the undirected case directly yields a directed spanner with the same bounds by simply bidirecting every edge. We show that also the lower dilation bound of $\Omega(n/k^{d/(d-1)})$ from the undirected case extends to directed tree-width on directed spanners.

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

2 Upper Bound for Path-Width Bounded Geometric Spanners

In this section we present a constructive upper bound on the dilation of geometric spanners of path-width k.

▶ **Theorem 2.1.** *Given a set of n points $\mathsf{P} \subset \mathbb{R}^{\mathsf{d}}$ and some positive integer $k \leq n^{1-1/d}$, there is a geometric spanner of path-width k and dilation $O(n/k^{d/(d-1)})$.*

As the upper bound for spanners of bounded path-width implies the same upper bound on spanners of bounded tree-width, our result can be seen as a strengthening of the bound in [5]. The basic idea used to show this bound is to construct smaller spanners on subsets of size $O(n/k^{d/(d-1)})$ of the given set of points, which are computed by recursively removing separator edges from the Euclidean minimum spanning tree. Then, a path for each of these subsets is calculated. Finally, a greedy spanner [14] is computed on top of the smaller spanners to connect them. This algorithm is based on the same ideas as the algorithm for the tree-width bound in [5], though we need some additional tools. Most importantly, given a set P of n points, we need to be able to compute a path with dilation $O(n)$ on P. The construction of such a path was described by Matoušek in [13], though the result was stated in terms of embedding a metric into the real number line. Given a set of points P we use $\mathrm{EMST}(\mathsf{P})$ to denote the Euclidean minimum spanning tree of P and given a geometric graph G we use $G[\leq d]$ to denote the geometric graph obtained from G by deleting all edges of length $> d$. Going further, we assume that the distance between two points in P is at least 1.

▶ **Lemma 2.2** (Statement 3.2. in [13])**.** *Let $\mathsf{P} \subset \mathbb{R}^d$ be a set of points. There is a path π containing all points of P such that for every s that is a power of two the induced subgraphs $\pi[\mathsf{P}_i]$ are connected where $\mathsf{P}_1, \ldots, \mathsf{P}_m$ denote the points of the connected components of $(EMST(\mathsf{P}))[\leq s]$. The length of every path $\pi[\mathsf{P}_i]$ is $\leq 6|\mathsf{P}_i| \cdot s$ and the dilation is $\leq 12|\mathsf{P}_i|$.*

Algorithm 1

1: **if** $k = 1$ **then**
2: **return** path of Lemma 2.2 on P with dilation $\leq 12n$
3: $T_{\mathrm{MST}} \leftarrow \mathrm{EMST}(\mathsf{P})$
4: $m \leftarrow \left\lceil (k/C)^{d/(d-1)} + 1 \right\rceil$ where C is some constant that is chosen later.
5: Compute a set $\mathcal{T}$ of m disjoint subtrees of T_{MST}, each containing $O(n/m)$ points, as for example explained in the proof of Lemma 8 in [5].
6: For $T \in \mathcal{T}$ let $R(T)$ be the vertices in T incident to edges removed by the previous step.
7: For each $T \in \mathcal{T}$ iterate through its edges e from long to short. If e lies on a path between two vertices in $R(T)$, remove the edge and thereby split T into two new subtrees T' and T''. Remove T from $\mathcal{T}$ and insert T' and T''. Set $R(T') = R(T) \cap V(T')$ and $R(T'') = R(T) \cap V(T'')$. This is done until for every $T \in \mathcal{T}$ we have $|R(T)| = 1$.
8: For each $T \in \mathcal{T}$ let $\pi(T)$ be the path of Lemma 2.2 on $V(T)$
9: $E' \leftarrow \bigcup_{T \in \mathcal{T}} \pi(T)$
10: $(R, E'') \leftarrow$ greedy 3/2-spanner for $R = \bigcup_{T \in \mathcal{T}} R(T)$
11: **return** $G = (\mathsf{P}, E' \cup E'')$

Similar to the construction in [5], we use the fact that the greedy t-spanner for $t \in (1, 3/2]$ has sublinear separators, as shown by Le and Than in [12], to construct a path-decomposition, i.e. the next lemma follows from Theorem 20 (iii) in [2] and Theorem 2 in [12].

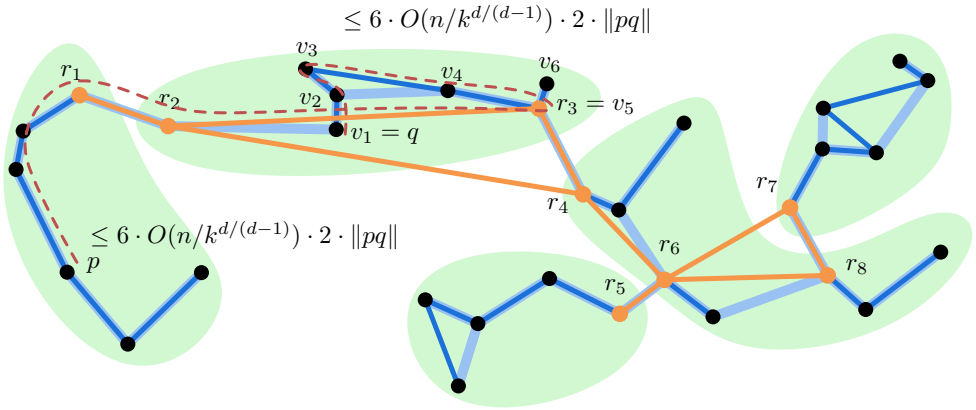


Figure 1 The subtree-paths (dark blue) on the original subtrees (green) and the greedy spanner (orange) as well as a shortest path (red) between points p and q. The EMST is shown in light blue.

▶ **Lemma 2.3.** *Let $\mathsf{P} \subset \mathbb{R}^d$ be a set of n points and G the greedy $3/2$-spanner of P. G has path-width $O(n^{1-1/d})$.*

We will call the paths computed in line 8 of the algorithm the *subtree-paths* and the points of R the *representatives*. The set of representatives has size at most $2m-2$, since each representative is incident to at least one of the $m-1$ edges removed. Thus, the greedy spanner (R, E'') of line 10 is a greedy spanner on at most $2m-2$ points. By Lemma 2.3 and an appropriate choice of the constant C the path-width of (R, E'') is at most $k-2$. After line 7 of the algorithm each of the subtrees of the subtrees $T \in \mathcal{T}$ contains only one representative and therefore each subtree-path contains one representative. Thus, the graph G consists of (R, E'') where each vertex is contained in an additional path. To construct a path-decomposition for G we simply take a path-decomposition of width $k-2$ for (R, E'') and modify it. We locate a bag containing the representative of a subtree-path and take a path-decomposition of width 1 for the subtree-path. Then we copy the located bag multiple times and merge the bags of the path-decomposition adding the points of the subtree-path.

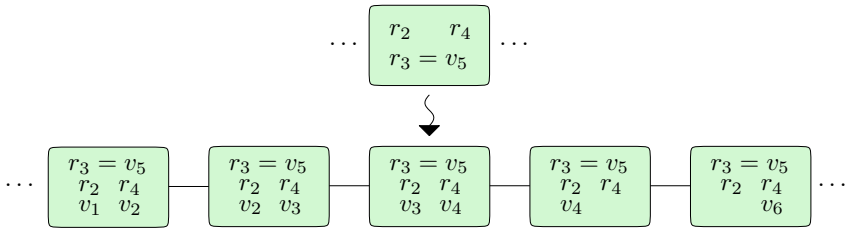


Figure 2 Construction of a path-decomposition for the spanner in Figure 1 based on the path-decomposition of the greedy spanner.

▶ **Lemma 2.4.** *Given a set $\mathsf{P} \subset \mathbb{R}^d$ of n points and some positive integer $k \leq n^{1-1/d}$, Algorithm 1 computes a spanner of dilation $O(n/k^{d/(d-1)})$.*

Proof idea. For two points p, q in the same subtree-path the dilation is small by Lemma 2.2 and the fact that there are only $O(n/k^{d/(d-1)})$ edges on the path between p and q in the EMST and that all these edges are short. For points in different subtree-paths the path in

the EMST either contains the representatives of each of the two subtrees or the edges in the EMST to the representatives are all short, meaning that in the subtree-paths the length of the paths to the representatives is small. This can be observed when setting s to be the smallest power of two for which there is a path in the induced subgraph of the EMST in Lemma 2.2. Between the representatives we use the edges of the greedy spanner. The length of the path along the edges of the greedy spanner is bounded by the length of the paths to the representatives in the subtree-paths and the distance between p and q. ◀

3 Bounds on the Clique-Width and Cut-Width of Geometric Spanners

Having the result above as well as the results from [5] we can quite easily obtain geometric spanners bounded in other width parameters. Constructing a geometric spanner of low clique-width is trivial to do, as every complete graph has clique-width 2. For spanners of clique-width k and the additional requirement of planarity we can show the following result:

▶ **Corollary 3.1.** *Given a set $\mathsf{P} \subset \mathbb{R}^2$ of n points and some integer $k \in [4, 72\sqrt{n-3}]$, a plane spanner with clique-width k, maximum vertex degree 4 and dilation $O(n/k^2)$ can be constructed in $O(n \log n)$ time. The dilation bound is asymptotic worst-case optimal for plane spanners.*

This result follows from the algorithm of Corollary 4 in [5] for plane spanners of tree-width k and the fact that a planar graph of tree-width k has clique-width at most $6k-2$ [7]. The lower bound follows from the lower bound on the tree-width in [5] and the lower bound on the clique-width of $(k+1)/6$ for planar graphs of tree-width k [9].

The next result on the cut-width follows from the path-width bound above:

▶ **Corollary 3.2.** *Given a set of n points $\mathsf{P} \subset \mathbb{R}^d$ and some positive integer $k \leq n^{1-1/d}$, there is a geometric spanner of cut-width k, and dilation $O(n/k^{d/(d-1)})$. The dilation bound is asymptotic worst-case optimal.*

This result holds since the cut-width of a graph G with path-width k is bounded by $\Delta(G) \cdot k$ [6] where $\Delta(G)$ is the maximum degree, which for the constructed spanner is bounded by the degree of the greedy 3/2-spanner, which is constant [14], plus 2 for the decompositions of width 1. As the cut-width is a lower bound for the tree-width [1] the lower bound follows from the lower bound in [5].

4 Lower Bound on the Directed Tree-Width of Geometric Spanners

Historically, there have been many different definitions of tree-width for directed graphs, most of which are equivalent up to some constant factor. The notion of directed tree-width we are using is the one introduced by Giannopoulou, Kawarabayashi, Kreutzer, and Kwon in [8]. Kim, Hatzel and Kreutzer showed in [10] that this definition is equivalent up to a constant factor to all other common definitions of directed tree-width.

We denote a directed tree-decomposition for a digraph D as a triple $\mathcal{T} = (T, \mathcal{X}, \mathcal{W})$, where T is a rooted directed tree, $\mathcal{X} : V(T) \to 2^{V(D)}$ and $\mathcal{W} : E(T) \to 2^{V(D)}$. For every $t \in V(T)$, we define $T_t := T[\{t' \in V(T) \mid t' \text{ is reachable from } t \text{ by a directed path in } T\}]$ and for a subtree $S \subseteq T$, we define $\mathcal{X}(S) := \bigcup_{t \in V(S)} \mathcal{X}(t)$. For a directed tree-decomposition, the following two conditions need to hold:

(dtw1) $\{\mathcal{X}(t) \mid t \in V(T)\}$ is a partition of $V(D)$ into possibly empty sets such that $\mathcal{X}(r) \neq \emptyset$, where r is the root of T and

(dtw2) for all $e = (s,t) \in E(T)$, there is no closed walk in $D - \mathcal{W}(e)$ containing a vertex of $\mathcal{X}(T_t)$ and a vertex of $V(D) - \mathcal{X}(T_t)$.

The width $w(\mathcal{T})$ of $\mathcal{T}$ is $\max\{|\mathcal{X}(t) \cup \bigcup_{e \sim t} \mathcal{W}(e)| - 1 \mid t \in V(T)\}$ where $e \sim t$ means that t is an endpoint of the edge e. The directed tree-width of D, is defined as $\min\{w(\mathcal{T}) \mid \mathcal{T}$ is a directed tree decomposition of $D\}$.

In this section we show, that the lower bound on the dilation of tree-width bounded spanners in [5] can be extended to the directed setting.

▶ **Theorem 4.1.** *Let $d \geq 2$ be a fixed integer. For positive integers n and $k \leq n^{(d-1)/d} \cdot (12d)^{1/d-2}$ there is a set of n points in $\mathbb{R}^d$, such that every directed geometric spanner of directed tree-width k on this set has dilation $\Omega(n/k^{d/(d-1)})$.*

To prove this bound we use a similar set of points as in [5] that will resemble the $(h+1)^d$-grid (d-dimensional $(h+1) \times ... \times (h+1)$-grid), where $h = \lceil (9d \cdot (k+2))^{1/(d-1)} - 1 \rceil$. We will make use of the fact, that the bidirected version of this d-dimensional grid has large directed tree-width.

▶ **Lemma 4.2.** *The bidirected n^d-grid has directed tree-width $\geq \frac{1}{9d} \cdot n^{d-1} - 1$.*

This bound can be shown using Lemma 12 in [5], as well as an observation regarding strict brambles in [11] and Corollary 2.9 in [10].

The number of points that will represent an edge is $m = \lfloor n/(d \cdot (h+1)^d - d \cdot (h+1)^{d-1}) \rfloor$. Formally we define the set of points which serves as our lower bound instance as $\mathsf{P}_{d,n,k} = \bigcup_{i=1}^{d} \mathsf{P}_i$ where

$$\mathsf{P}_i = \bigcup_{j_1=0}^{h} \cdots \bigcup_{j_d=0}^{h} \{(j_1 \cdot m, \ldots, j_{i-1} \cdot m, x_i, j_{i+1} \cdot m, \ldots, j_d \cdot m) \mid x_i \in \{0, \ldots, h \cdot m\}\}.$$

The proof in [5] for the lower bound on the dilation heavily relies on contractions of paths. Unfortunately directed tree-width is not closed under arbitrary edge contraction. Instead we are using contractions of strongly connected subdigraphs. It was shown in [10], that the definition of directed tree-width used here is closed under a related but different notion of contractions, but the following Lemma can be shown along similar lines:

▶ **Lemma 4.3.** *Directed tree-width is closed under strongly connected contractions.*

Proof idea. Let D be a digraph of directed tree-width k, C a strongly connected subdigraph in D and D' the digraph obtained by contracting C in D to a vertex v. By removing all vertices of C in the $\mathcal{X}$ of a directed tree-decomposition of D except one and instead replace it by v and replacing all vertices of C by v in $\mathcal{W}$ we obtain a directed tree-decomposition $(T, \mathcal{X}', \mathcal{W}')$ of D'. ◀

Using these contractions on a spanner on $\mathsf{P}_{d,n,k}$ we can show the following lemma:

▶ **Lemma 4.4.** *If D is a directed geometric $o(n/k^{d/(d-1)})$-spanner on $\mathsf{P}_{d,n,k}$ then it must be of directed tree-width $> k$.*

Proof idea. Let D be a directed geometric spanner with dilation $o(n/k^{d/(d-1)})$ on $\mathsf{P}_{d,n,k}$. For every pair of points s and t with distance 1 there must be a shortest path from s to t and a shortest path from t to s in D. The union of these paths is a closed walk through s and t. This means that for every pair of points of distance 1 along an edge of the underlying grid there is a closed walk. Because of the dilation bound, the union of these walks must

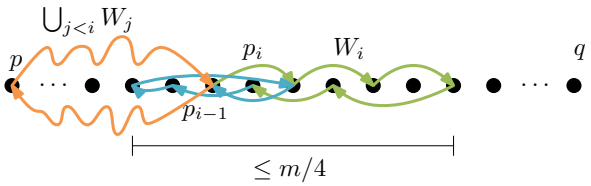


Figure 3 The two shortest paths between p_{i-1} and p_i form a closed walk and the union of all closed walks between points of distance 1 form a strongly connected component.

stay within a specific region around the edge of the underlying grid. We can also show that the walks for points close to a point representing a vertex of the grid must be disjoint to all other closed walks for points close to a different such point. This means that we can contract the closed walks close to these points without affecting the walks between points close to different such points too much. Since the closed walks are strongly connected, these contractions do not increase the directed tree-width. The only points left after performing these contractions are the points representing the vertices of the underlying grid. The resulting digraph is a bidirected $(h+1)^d$-grid, which means that D by Lemma 4.2 must have directed tree-width $> k$. ◀

References

1 Hans L. Bodlaender. Some classes of graphs with bounded treewidth. *Bull. EATCS*, 36:116–125, 1988.

2 Hans L. Bodlaender. A partial k-arboretum of graphs with bounded treewidth. *Theor. Comput. Sci.*, 209(1-2):1–45, 1998. doi:10.1016/S0304-3975(97)00228-4.

3 Prosenjit Bose and Michiel H. M. Smid. On plane geometric spanners: A survey and open problems. *Comput. Geom.*, 46(7):818–830, 2013. doi:10.1016/J.COMGEO.2013.04.002.

4 Kevin Buchin, Antonia Kalb, Anil Maheshwari, Saeed Odak, Carolin Rehs, Michiel Smid, and Sampson Wong. Computing oriented spanners and their dilation. In *Proceedings of the International Symposium on Computational Geometry (SoCG)*, volume 332 of *LIPIcs*, pages 27:1–27:17. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2025. doi:10.4230/LIPICS.SOCG.2025.27.

5 Kevin Buchin, Carolin Rehs, and Torben Scheele. Geometric spanners of bounded tree-width. In *Proceedings of the International Symposium on Computational Geometry (SoCG)*, volume 332 of *LIPIcs*, pages 26:1–26:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2025. doi:10.4230/LIPICS.SOCG.2025.26.

6 Fan R. K. Chung and Paul D. Seymour. Graphs with small bandwidth and cutwidth. *Discret. Math.*, 75(1-3):113–119, 1989. doi:10.1016/0012-365X(89)90083-6.

7 Bruno Courcelle. On quasi-planar graphs: Clique-width and logical description. *Discret. Appl. Math.*, 278:118–135, 2020. doi:10.1016/J.DAM.2018.07.022.

8 Archontia C. Giannopoulou, Ken-ichi Kawarabayashi, Stephan Kreutzer, and O-joung Kwon. Directed tangle tree-decompositions and applications. In *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 377–405. SIAM, 2022. doi:10.1137/1.9781611977073.19.

9 Frank Gurski and Egon Wanke. The tree-width of clique-width bounded graphs without $K_{n,n}$. In *Proceedings of International Workshop on Graph-Theoretic Concepts in Computer Science (WG)*, volume 1928 of *Lecture Notes in Computer Science*, pages 196–205. Springer, 2000. doi:10.1007/3-540-40064-8_19.

10 Gunwoo Kim, Meike Hatzel, and Stephan Kreutzer. Directed treewidth is closed under taking butterfly minors, 2025. URL: https://arxiv.org/abs/2503.23977, arXiv:2503.23977.

11 Kyohei Kozawa, Yota Otachi, and Koichi Yamazaki. Lower bounds for treewidth of product graphs. *Discret. Appl. Math.*, 162:251–258, 2014. doi:10.1016/J.DAM.2013.08.005.

12 Hung Le and Cuong Than. Greedy spanners in euclidean spaces admit sublinear separators. *ACM Trans. Algorithms*, 20(3):23, 2024. doi:10.1145/3590771.

13 Jiří Matoušek. Bi-lipschitz embeddings into low-dimensional euclidean spaces. *Commentationes Mathematicae Universitatis Carolinae*, 031(3):589–600, 1990. URL: http://eudml.org/doc/17879.

14 Giri Narasimhan and Michiel H. M. Smid. *Geometric spanner networks*. Cambridge University Press, 2007.

15 Neil Robertson and Paul D. Seymour. Graph minors. i. excluding a forest. *J. Comb. Theory B*, 35(1):39–61, 1983. doi:10.1016/0095-8956(83)90079-5.

16 Neil Robertson and Paul D. Seymour. Graph minors. II. algorithmic aspects of tree-width. *J. Algorithms*, 7(3):309–322, 1986. doi:10.1016/0196-6774(86)90023-4.

Explicit High-Chromatic Hypergraphs Realized by Axis-Parallel Rectangles *

Gábor Damásdi[1]

1 **HUN-REN Alfréd Rényi Institute of Mathematics and ELTE Eötvös Loránd University, Budapest**
damasdigabor@caesar.elte.hu

Abstract

We present a new family of hypergraphs with arbitrarily large uniformity and chromatic number. As an application, we give a new proof of a theorem of Chen, Pach, Szegedy, and Tardos. They showed that for any constants $c, k > 1$, there exists a set P of points in the plane with the following property: for every coloring of P with c colors, there is an axis-parallel rectangle containing at least k points, all of the same color. Their original proof is probabilistic; we present an explicit construction. Moreover, in the case $k = 2$, we show that one can even realize a graph with arbitrarily high girth and chromatic number using points and axis-parallel rectangles.

Related Version arXiv:2602.18235

1 Introduction

Coloring questions on graphs and hypergraphs that arise from geometry have always played an important role in combinatorics. If we are given a set P of points and a family $\mathcal{F}$ of sets in the plane, they naturally define a hypergraph by incidence. Such hypergraphs are known as *geometric hypergraphs*. A proper c-coloring of a hypergraph $H = (V, E)$ is a map $V \to \{1, \dots, c\}$ such that no edge is monochromatic. The chromatic number of H is the smallest c for which we have a proper c-coloring of H.

The study of the chromatic number of geometric hypergraphs is a vast topic; see [4] for a recent survey. In this paper, we focus on the case where $\mathcal{F}$ contains axis-parallel rectangles. This was first considered by Chen, Pach, Szegedy, and Tardos [3]. Using a probabilistic argument, they showed that we can realize hypergraphs of arbitrarily large chromatic number and uniformity using axis-parallel rectangles.

▶ **Theorem 1.1** (Chen, Pach, Szegedy, and Tardos [3]). *For any constants $c, k > 1$, there exists a set P of points in the plane with the following property: for every coloring of P with c colors, there is an axis-parallel rectangle containing k points, all of the same color.*

Chekan and Ueckerdt gave an explicit set P of points and a family $\mathcal{F}$ of horizontal strips and bottomless axis-parallel rectangles such that the incidence hypergraph $H(V, \mathcal{F})$ is k-uniform and not 2-colorable [2]. As strips and bottomless rectangles can be replaced by normal rectangles, their construction gives an explicit solution for Theorem 1.1 in the case $c = 2$. We generalize this construction for $c > 2$ and show a slightly stronger version of Theorem 1.1. A set of points is called *ascending* (*descending*) if the x-coordinate order of the points is the same as (reverse of) the y-coordinate order of the points.

* Supported by the ERC Advanced Grant no. 882971 GeoScape, by the János Bolyai Research Scholarship of the Hungarian Academy of Sciences, the STARTING-25 project no. 152199 and the EXCELLENCE-24 project no. 151504 Combinatorics and Geometry of the NRDI Fund.

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

▶ **Theorem 1.2.** *For any constants $c, k > 0$, there exists an explicit set P of points in the plane with the following property: for every coloring of P with c colors, there is an axis-parallel rectangle containing k* ***ascending*** *points, all of the same color.*

Finding graphs and hypergraphs that have high girth and high chromatic number at the same time is a classical topic, going back to the works of Tutte [8] (under the pseudonym Blanche Descartes) and Erdős [9]. High girth constructions were recently found in some geometric problems, see for example [6, 7]. Using a graph construction from [11] we show a high girth variant of Theorem 1.1 for the case $k = 2$.

▶ **Theorem 1.3.** *For any constants $c, g > 0$, there exists a set P of points and a set $\mathcal{R}$ of axis-parallel rectangles in the plane such that each rectangle contains exactly two points of P, and the incidence graph $H(P, \mathcal{R})$ has girth at least g and chromatic number at least c.*

2 Constructing the hypergraph

For each $k, c \geq 1$ we inductively construct a vertex-ordered k-uniform hypergraph H_k^c of chromatic number at least c. Our construction is a generalization of a construction by Chekan and Ueckerdt for 2-colors [2], and we mostly follow their terminology. The construction can also be viewed as a generalization of the k-ary tree hypergraph, see [1, 5, 12, 13] and Section 2.2 in [4] for details.

For $c = 1$ we take k vertices in arbitrary order and a single edge containing all of them. Let k and $c > 1$ be fixed. By induction H_k^{c-1} exists, let m denote the number of its vertices. We will start by creating an auxiliary graph, a rooted forest consisting of m^k rooted trees. We will partition the vertex set of H_k^c into *stages*, each stage will contain at most one vertex from each tree. The vertices of a given stage will be at a fixed distance j from the roots of their corresponding tree, which we will call the *level* of the stage. Each stage will be of level $0, 1, \ldots, k-2$ or $k-1$. Each stage S of level j will contain exactly m^{k-j} vertices, and we will have a fixed ordering $<_S$ on these vertices.

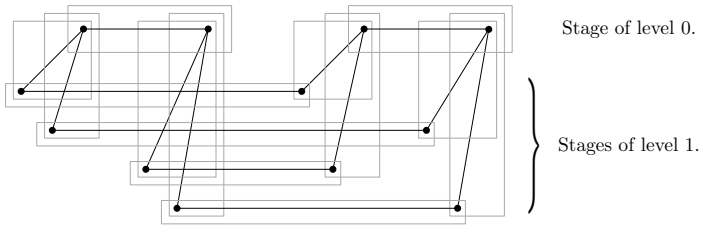


Figure 1 The hypergraph H_2^2 and its realization by axis-parallel rectangles.

Furthermore, we think of the m^{k-j} vertices of a stage as m^{k-j-1} disjoint *groups* of m consecutive vertices in the ordering $<_S$. We will need to talk about subsets of a stage that contain exactly one vertex from each group, hence the following definition will be useful. For an ordered set $A = \{a_1, a_2, \ldots, a_{tm}\}$ let $f_m(A)$ denote the family of subsets of A that contain exactly one element from each set $\{a_{im+1}, a_{im+2}, \ldots, a_{im+m}\}$ where $i = 0, \ldots, t-1$.

We start building the forest by taking m^k ordered vertices, they form the unique stage of level 0. They also serve as the roots of the m^k trees. Suppose that we have already defined a stage S of level $j < k-1$. Then for each subset $S' \in f_m(S)$ we define a new stage $T(S')$ of level $j+1$ on m^{k-j-1} new vertices. Each vertex of S' gets exactly one child from $T(S')$ and the vertices of $T(S')$ inherit the ordering of their parents. That is, we add a perfect matching

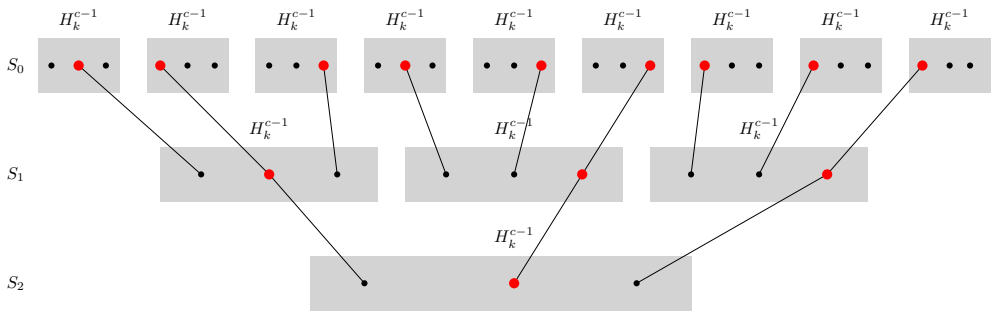


Figure 2 Finding a monochromatic path edge in the $k = 3$ case. Grey rectangles indicate a copy of H_k^{c-1}, red vertices indicate the elements in B_j in each stage.

between S' and $T(S')$. For stages of level $k-1$ we do not continue the process; hence, we obtain a forest of m^k trees, whose vertices are partitioned into stages of level $0, 1, \dots, k-1$.

To clarify the construction, let us calculate some properties of the forest. We have a unique stage S of level 0, containing m^k vertices. If $|A| = tm$ then $|f_m(A)| = m^t$. Hence S has $m^{m^{k-1}}$ child stages of level 1. Each vertex $v \in S$ has $m^{m^{k-1}-1}$ children, one for each $S' \in f_m(S)$ that contains it. Similarly for a stage of level j we have $m^{m^{k-j-1}}$ child stages of level $j+1$ and each vertex has $m^{m^{k-j-1}-1}$ children. Therefore, the number of stages of level j is

$$m^{m^{k-1}} \cdot m^{m^{k-2}} \cdots m^{m^{k-j}} = m^{m^{k-j}\frac{m^j-1}{m-1}}.$$

In total we have $m^{m^{k-j}\frac{m^j-1}{m-1}+k-j}$ vertices of level j.

Let V denote the vertex set of the forest, this will be the vertex set of H_k^c. For a vertex $v \in V$ let $root(v)$ denote the root of the tree that contains v. Let $path(v)$ denote the unique path from v to $root(v)$.

Now we are ready to define the edges of the hypergraph H_k^c. There are two types of edges. Let $\mathcal{E}_P = \{path(v) \mid v \text{ appears in a stage of level } k-1\}$, that is, for each vertex v that appears in a stage of level $k-1$ we add $path(v)$ as an edge to H_k^c. These are called *path edges*. Secondly, for each stage S of level j we do the following. Recall that the vertices of S are partitioned into m^{k-j-1} disjoint groups of m consecutive elements in the ordering $<_S$. In each of these groups we add edges to realize the vertex-ordered hypergraph H_k^{c-1} such that the vertex ordering of H_k^{c-1} matches the ordering $<_S$. That is, we realize m^{k-j-1} disjoint copies of H_k^{c-1} in the stage. The edges appearing in these will be called *transversal* edges and the set of all transversal edges appearing in any stage is denoted by $\mathcal{E}_T$.

▶ **Lemma 2.1.** *The k-uniform hypergraph $H_k^c = (V, \mathcal{E}_P \cup \mathcal{E}_T)$ is not properly c-colorable.*

Proof. We use induction on c. For $c = 1$ we have a single edge; hence the hypergraph is not 1-colorable. For $c > 1$ suppose that there is a proper c-coloring and fix a color, say the color red. The main idea is to use the induction hypothesis for the transversal edges to show that we can follow the color red down to a leaf in at least one of the trees.

Note that each stage S of level j contains at least m^{k-j-1} red vertices, one from each group of m vertices. In other words, there is a red set from $f_m(S)$. Indeed, the transversal hyperedges form m^{k-j-1} disjoint copies of H_k^{c-1}. By induction, each color must appear in each of these copies, hence we find a red vertex in each.

Using this observation and induction on j, we show that for each $j = 0, \dots, k-1$ there is a stage S_j of level j and a subset $B_j \in f_m(S_j)$ such that for each $v \in B_j$ the entire path

$path(v)$ is red. See Figure 2 for an example. For S_0 we take the unique stage of level 0. It contains m^{k-1} copies of H_k^{c-1}, we pick a red vertex from each to obtain B_0. Suppose that we have already found B_j for some $j < k-1$. As $B_j \in f_m(S_j)$ there is a stage $S_{j+1} = T(B_j)$. Using induction on c, we can pick a red vertex in each group of S_{j+1}, giving us B_{j+1}. The parents of the vertices of B_{j+1} all belong to B_j, therefore, for any $v \in B_{j+1}$ the path $path(v)$ is red.

Finally, we find B_{k-1} in a stage of level $k-1$. It contains $m^{k-(k-1)-1} = 1$ vertices, i.e. a single vertex v. The path $path(v)$ is red, a contradiction. Hence, there is no proper c-coloring of the hypergraph H_k^c.

◀

3 Realization using axis-parallel rectangles

In this section we prove Theorem 1.2. By Lemma 2.1 it is enough to show the following.

▶ **Lemma 3.1.** *For every k, c the hypergraph H_k^c can be realized using axis-parallel rectangles such that each rectangle contains an ascending set of k points.*

Proof. We use induction on c. For $c = 1$ the hypergraph H_k^c has a single edge, and hence we can pick any set of k ascending points. For $c > 1$, we find the realization of H_k^c in two phases. In the first phase, we embed the vertices such that the following properties hold:

1. The vertices of a given stage S lie on a unique horizontal line, and are ordered according to $<_S$ along the line. We refer to this line as the *stage line of S*.
2. The path edges are realizable by axis-parallel rectangles in this embedding.

Then, in the second phase, we perturb the vertices in each stage so that the transversal hyperedges also become realizable.

We start by placing the roots on an arbitrary horizontal line in order. Then, in each step, we pick a stage S that is already embedded but whose child stages $T_1, T_2, \ldots, T_r$ are not, and embed the child stages simultaneously in the following way. We pick r distinct horizontal lines below the line of S and above the next stage line, if there is one. For each $i \in \{1, \ldots, r\}$ we place the vertices of T_i on the i-th line so that each vertex has the same x-coordinate as its parent in S.

Then, for each vertex $v \in S$ we slightly shift the children of v to the left in the following way. Let w be the vertex before v in the x-coordinate ordering of the whole set of points, if there is any. We shift the children of v so that each stays on its stage line, they form a descending set, and they stay between v and w in the x-coordinate ordering, see Figure 3.

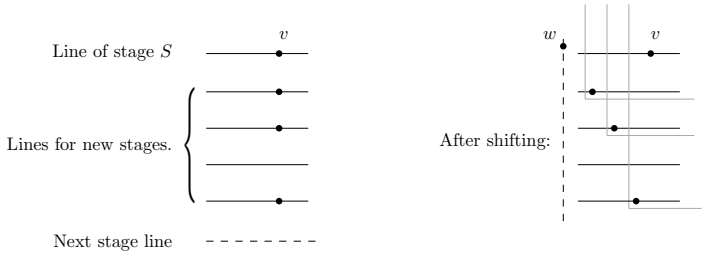


Figure 3 The children of v are placed below v then shifted to the left to form a descending set.

Note that each non-root vertex is below and to the left of its parent, and hence each $path(v)$ set is ascending. Furthermore, when we shift the vertices to the left, the order of the

vertices within a stage does not change.

We claim that at the end for every vertex $v \in V$ the rectangle R_v with bottom-left corner v and top-right corner $root(v)$ contains exactly the vertices of $path(v)$. Since $path(v)$ is ascending, R_v contains $path(v)$. We only need to show that any vertex $w \notin path(v)$ is not in R_v. Let p denote the highest level vertex in $path(v) \cap path(w)$. Let v' and w' denote the two children of p in $path(v)$ and $path(w)$, respectively. By the definition of p, we know that v' and w' are distinct vertices and that they were created when we processed the stage containing p. The children of p were placed in descending order, hence v' is either above and to the left of w' or below and to the right of w', see Figure 4.

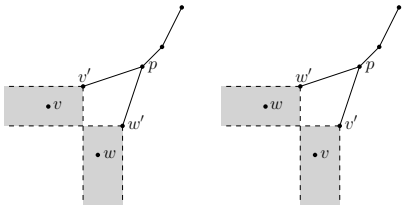


Figure 4 The two cases depending on how v' and w' were placed. The gray regions indicate the possible positions of v' and w' in the plane.

Suppose v' is above and to the left of w'. All descendants of w', including w, are below w'. On the other hand, all descendants of v' are placed later than w', hence they are on lines above the line of w'. Therefore, v is above w', and w cannot be in R_v.

Suppose v' is below and to the right of w'. All descendants of w', including w, are to the left of w'. As the descendants of v' are placed later than w', they are all to the right of it. Hence v is to the right of w and w cannot be in R_v.

Next, we realize the transversal edges. Right now the vertices of a given stage S of level j lie on a horizontal line and their x-coordinate order matches $<_S$, that is, each of the m^{k-j-1} copies of H_k^{c-1} that are formed by the transversal edges is on consecutive m vertices. Hence, we can take each group of m vertices and replace them by a very thin copy of H_k^{c-1} whose vertices are close to the corresponding vertices in the group. If the copy is thin enough, the rectangles cannot contain points from any other stage. If we place the vertices close to their original position on the line, then the existing edges do not change. Hence, we have realized H_k^c using axis-parallel rectangles.

◀

4 High girth variant

Proof of Theorem 1.3. Our goal is to show that for $k = 2$ we can achieve high girth. Let us recall a construction from [11] using our terminology of stages. For each c, g we construct an ordered graph $G^c(g)$ of girth at least g and chromatic number at least c. We use induction on c. For $c = 1$, take the graph K_2, i.e. a single edge.

For $c > 1$, fix an auxiliary hypergraph $H = (V_H, \mathcal{E}_H)$ of uniformity $|G^{c-1}(g)|$, girth g, and chromatic number at least c (such hypergraphs exist by standard probabilistic arguments, see [10]). We use H to define our stages. That is, we will replace the set system $f_m(S)$ by H.

Create a stage of level 0 that contains the vertices of H, but, contrary to the previous construction, do not place any edges inside this stage. All of the remaining stages are going to be of level 1. For each edge $S \in \mathcal{E}_H$ add a stage $T(S)$ of level 1. It contains a child vertex v' for each $v \in S$. We connect v and v', and $<_{T(S)}$ is the ordering inherited from S. As H is

$|G^{c-1}(g)|$-uniform, $T(S)$ contains $|G^{c-1}(g)|$ vertices. Add edges in $T(S)$ to realize a single copy of the graph $G^{c-1}(g)$ such that the ordering of the vertices matches the ordering $<_{T(S)}$.

$G^c(g)$ has chromatic number at least c and girth at least g, see the discussion after Property 4 in [11]. As $G^c(g)$ has the same type of stage structure as the H_k^c hypergraphs, the realization of $G^c(g)$ by axis-parallel rectangles is essentially the same as in Lemma 3.1. We leave the details to the reader. ◀

Constructing hypergraphs with uniformity larger than 2 that are realizable by axis-parallel rectangles and have large girth and chromatic number is a nice open problem.

Acknowledgments

This project was initiated at the Early Career Researchers in Combinatorics 2024 workshop in Edinburgh. The author thanks the organizers for creating a stimulating research environment. We are grateful to Tim Planken for pointing out the connection to the construction of Chekan and Ueckerdt.

References

1 Eyal Ackerman, Balázs Keszegh, and Dömötör Pálvölgyi. Coloring hypergraphs defined by stabbed pseudo-disks and ABAB-free hypergraphs. *SIAM Journal on Discrete Mathematics*, 34(4):2250–2269, 2020.

2 Vera Chekan and Torsten Ueckerdt. Polychromatic colorings of unions of geometric hypergraphs. In *International Workshop on Graph-Theoretic Concepts in Computer Science*, pages 144–157. Springer, 2022.

3 Xiaomin Chen, János Pach, Mario Szegedy, and Gábor Tardos. Delaunay graphs of point sets in the plane with respect to axis-parallel rectangles. *Random Structures & Algorithms*, 34(1):11–23, 2009.

4 Gábor Damásdi, Balázs Keszegh, János Pach, Dömötör Pálvölgyi, and Géza Tóth. Coloring Geometric Hypergraphs: A Survey. *arXiv preprint arXiv:2512.09509*, 2025.

5 Gábor Damásdi and Dömötör Pálvölgyi. Realizing an m-uniform four-chromatic hypergraph with disks. *Combinatorica*, 42(1):1027–1048, 2022.

6 James Davies. Box and segment intersection graphs with large girth and chromatic number. *Advances in Combinatorics*, July 2021.

7 James Davies, Chaya Keller, Linda Kleist, Shakhar Smorodinsky, and Bartosz Walczak. A solution to Ringel's circle problem. *Journal of the European Mathematical Society*, 2024.

8 Blanche Descartes. Solution to advanced problem no. 4526. *Amer. Math. Monthly*, 61:532, 1954.

9 Paul Erdős. Graph theory and probability. *Canadian Journal of Mathematics*, 11:34–38, 1959.

10 Paul Erdős and András Hajnal. On chromatic number of graphs and set-systems. *Acta Mathematica Academiae Scientiarum Hungaricae*, 17:61–99, 1966.

11 Alexandr V Kostochka and Jaroslav Nešetřil. Properties of Descartes' construction of triangle-free graphs with high chromatic number. *Combinatorics, Probability and Computing*, 8(5):467–472, 1999.

12 Nirman Kumar. On the minimum edge size for 2-colorability and realizability of hypergraphs by axis-parallel rectangles. In *CCCG*, pages 226–231, 2017.

13 János Pach, Gábor Tardos, and Géza Tóth. Indecomposable coverings. *Canadian mathematical bulletin*, 52(3):451–463, 2009.

Star-Based Separators for Intersection Graphs of c-Colored Pseudo-Segments

Mark de Berg[1], Bart M.P. Jansen[1], and Jeroen S.K. Lamme[1]

1 **Department of Mathematics and Computer Science, TU Eindhoven, the Netherlands**
{M.T.d.Berg|B.M.P.Jansen|J.S.K.Lamme}@tue.nl

Abstract

In this paper we introduce *star-based separators*, which are separators consisting of a small number of stars. We prove that the intersection graph of any set of axis-aligned segments admits a star-based separator consisting of $O(\sqrt{n})$ stars. In fact, our result is more general, as it applies to any set of n pseudo-segments that is partitioned into c subsets such that the pseudo-segments in the same subset are pairwise disjoint. These results lead to an almost-exact distance oracle for such intersection graphs, which has $O(n\sqrt{n})$ storage and $O(\sqrt{n})$ query time, and that can report the hop-distance between any two query nodes in the intersection graph with an additive error of at most 2.

Related Version This work appeared in ISAAC 2025 [8]. A full version is available on arXiv [6].

1 Introduction

Background. The celebrated Planar Separator Theorem by Lipton and Tarjan [12] states that any planar graph admits a balanced separator of size $O(\sqrt{n})$. More precisely, for any planar graph $G = (V, E)$ with n nodes there exists a set $S \subseteq V$ of size $O(\sqrt{n})$ whose removal partitions G into components of at most $\frac{2n}{3}$ nodes each. This fundamental tool has been used to develop efficient algorithms for many classic problems on planar graphs.

Geometric intersection graphs—graphs whose nodes correspond to objects in the plane and that have an edge between two nodes iff the corresponding objects intersect—are a generalization of planar graphs that have received widespread attention. Unfortunately, geometric intersection graphs do not admit balanced separators of sublinear size, because they can contain arbitrarily large cliques. This led De Berg *et al.* [7] to introduce so-called *clique-based separators*: balanced separators that consist of a small number of disjoint (but potentially large) cliques instead of a small number of nodes. They proved that any disk graph—and more generally, any intersection graph of convex fat objects in the plane—admits a clique-based separator of size $O(\sqrt{n})$. Here the *size* of a clique-based separator is the number of cliques it consists of. They also showed how to use such clique-based separators to obtain sub-exponential algorithms for various classic graph problems, including Independent Set, Dominating Set, and Feedback Vertex Set.

One may wonder if *all* geometric intersection graphs have sublinear clique-based separators. Unfortunately the answer is no, even for intersection graphs of horizontal and vertical line segments. The problem is that such graphs can contain arbitrarily large bicliques, and $K_{n,n}$ does not admit a sublinear clique-based separator. We therefore introduce *biclique-based separators*, which are separators consisting of bicliques, and we show that any set of horizontal and vertical segments admits a balanced separator consisting of a small number of bicliques. In fact, our result is stronger (as it uses *star graphs* in the separator, and not just any biclique) and it applies to a much wider class of intersection graphs, as discussed next.

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

Our contribution. Let $\mathcal{G} = (V, E)$ be an undirected graph with n nodes. A collection $S = \{S_1, \ldots S_t\}$ of (not necessarily induced) disjoint subgraphs from $\mathcal{G}$ is called a balanced[1] *biclique-based separator* for G if it has the following properties:

- Each subgraph S_i is a biclique.
- The removal from $\mathcal{G}$ of all subgraphs S_i and their incident edges partitions $\mathcal{G}$ into connected components with at most $\frac{2n}{3}$ nodes each.

The *size* of a biclique-based separator is the number of bicliques it is comprised of. If each biclique $S_i \in S$ is a star, then we call S a *star-based separator*. Note that the subgraphs S_i in a biclique-based separator (or: in a star-based separator) need not be induced subgraphs of $\mathcal{G}$. This is necessary to be able to handle large cliques. To avoid confusion between our new separators that are comprised of bicliques and the traditional separators that are comprised of individual nodes, we will refer to the latter as *node-based separators*.

We denote the intersection graph induced by a set V of n objects in the plane by $\mathcal{G}^\times[V]$. Thus, the nodes in $\mathcal{G}^\times[V]$ are in one-to-one correspondence with the objects in V and there is an edge between two objects $u, v \in V$ iff u intersects v. We prove that a star-based separator of size $O(\sqrt{n})$ exists for the intersection graph of any set V of axis-parallel segments; the bound on the size of the separator is tight in the worst case. In fact, we will prove that a star-based separator of size $O(\sqrt{n})$ exists for any set V of pseudo-segments[2] that is partitioned into subsets $V_1, \ldots, V_c$ such that the pseudo-segments from each V_i are disjoint from each other. We call such a set V a *c-colored set* of pseudo-segments, and we call $V_1, \ldots, V_c$ its color classes; see Figure 1(left). A c-oriented set of line segments such that no two segments from the same orientation intersect, is a c-colored set of pseudo-segments. Thus, a set of axis-parallel segments such that no two segments of the same orientation intersect, is a 2-colored set of pseudo-segments. In the full version [6] we extend our result to c-oriented polygons, and give a star-based separator of size $O(n^{2/3} \log^{2/3} n)$ for string graphs.

Application to distance oracles. A *distance oracle* for a (potentially weighted) graph $\mathcal{G} = (V, E)$ is a data structure that can quickly report the distance between two query nodes $s, t \in V$. Such queries can trivially be answered in $O(1)$ time if we store the distance between any two nodes in a distance table, but this requires $\Omega(n^2)$ storage. The challenge is to design distance oracles that use subquadratic storage. Unfortunately, this is not possible in general: any distance oracle must use $\Omega(n^2)$ bits of storage in the worst case, irrespective of the query time [13]. Work on distance oracles concentrated on special graph classes and, in particular, on planar graphs. More than two decades of research culminated in an exact distance oracle for weighted planar graphs that uses $O(n^{1+o(1)})$ storage and has $O(\log^2 n)$ query time [5]. For the unweighted case—in other words, if we are interested in the *hop-distance*—there is a $(1+\varepsilon)$-approximate distance oracle with $O(1/\varepsilon^2)$ query time and $O(n/\varepsilon^2)$ storage [11].

For geometric intersection graphs, only few results are known. Gao and Zhang [9], and Chan and Skrepetos [2], provide $(1+\varepsilon)$-approximate distance oracles with $O(n \log n)$ storage and $O(1)$ query time for weighted unit-disk graphs. No exact distance oracles that use subquadratic storage and have sublinear query time are known, even for unweighted unit-disk graphs. Very recently, Chang, Gao, and Le [4] presented an *almost-exact* distance-oracle for unit-disk graphs (and, more generally, for intersection graphs of similarly sized, convex, fat pseudodisks) that uses $O(n^{2-1/18})$ storage and that can report the distance between two

[1] In the sequel we will often omit the adjective *balanced* and simply speak of *separators*.

[2] A set V of curves in the plane is a set of pseudo-segments if any two curves in V are either disjoint or intersect in a single point that is a proper crossing (and not a tangency).

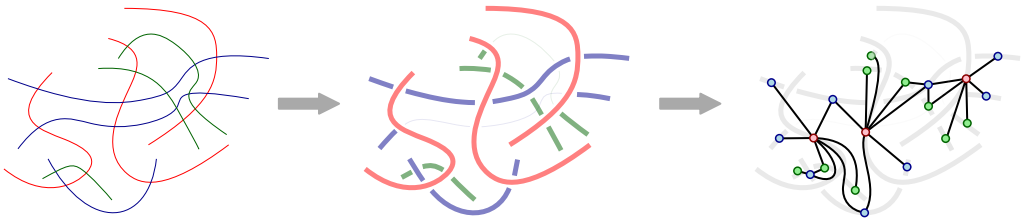


Figure 1 Left: A 3-colored set of pseudo-segments. Middle: The active fragments created by our algorithm. Right: The contact graph induced by the active fragments.

query nodes, up to an additive error[3] of 2, in $O(1)$ time. Another recent result is by Aronov, De Berg, and Theocharous [1], who presented an almost-exact distance oracle that uses clique-based separators. For intersection graphs of geodesic disks in the plane, their oracle uses $O(n^{7/4+\varepsilon})$ storage, has $O(n^{3/4+\varepsilon})$ query time, and can report the hop-distance between two query points up to an additive error of 1. Their approach also works with biclique-based separators; the only difference is that the additive error increases from 1 to 2. Thus, we obtain an almost-exact distance oracle for intersection graphs of c-colored pseudo-segments with $O(n\sqrt{n})$ storage and $O(\sqrt{n})$ query time. This is, to the best of our knowledge, the first almost-exact distance oracle for intersection graphs of non-disk-like objects.

2 A star-based separator for c-colored pseudo-segments

Let V be a c-colored set of pseudo-segments, as defined above. To simplify the terminology, from now on we simply refer to the pseudo-segments in V as *segments*. We assume that the segments in V are in general position and, in particular, that no three segments meet in a common point and that no endpoint of one segment lies on another segment. This assumption is without loss of generality, as it can always be ensured by perturbing the segments slightly.

The construction. Recall that a *contact graph* is the intersection graph of a set of interior-disjoint objects. Contact graphs of curves are known to be planar if no four objects meet in a common point [10]. Our strategy to construct a star-based separator S for $\mathcal{G}^{\times}[V]$ consists of the following steps, illustrated in Figure 1 and explained in more detail later.

Step 1. We partition each segment in V into *fragments*. Some fragments will be *active*, while others will be *inactive*. This partition will be such that active segments do not cross each other, although they may touch.

Step 2. Let $\mathcal{H}$ be the contact graph on the active segments. We construct a separator $S_{\mathcal{H}}$ on $\mathcal{H}$, using a suitable weighting scheme on the nodes of $\mathcal{H}$. Because $\mathcal{H}$ is planar, constructing the separator can be done using the Planar Separator Theorem.

Step 3. We use $S_{\mathcal{H}}$ to construct our star-based separator S for $\mathcal{G}^{\times}[V]$. For a fragment f, let $\text{seg}(f) \in V$ denote the segment containing f. Intuitively, we want to put a star into S for each fragment f in the separator $S_{\mathcal{H}}$, namely, the star consisting of the segment $\text{seg}(f)$ as well as all other segments intersecting $\text{seg}(f)$. For technical reasons, however, we actually have to put a slightly larger collection of stars into S.

[3] In the most recent arxiv version [3] the error has been reduced to 1.

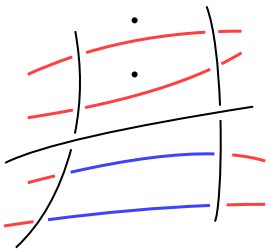


Figure 2 A collection of colored fragments X_i and black fragments from previous iterations. The blue fragments share an equivalence class, each red fragment is only equivalent to itself.

To make this strategy work, we need to control the size of the contact graph $\mathcal{H}$. More precisely, to obtain a star-based separator of size $O(\sqrt{n})$, the size of $\mathcal{H}$ needs to be $O(n)$. Thus we cannot, for example, cut each segment into fragments at its intersection points with all other segments and make all the resulting fragments active. On the other hand, if we ignore certain parts of the segments by making them inactive, we miss certain intersections and we run the risk that our final set of stars is no longer a valid separator. Next, we briefly describe how to overcome these problems by carefully creating the active fragments and why these steps are correct. For a complete description and the correctness proofs, we refer the reader to the full version [6].

Step 1: Creating the active fragments. To construct the active fragments that form the nodes in our contact graph $\mathcal{H}$, we will go over the subsets $V_1, \ldots, V_c$ one by one.

Each segment in V_1 becomes a single fragment, and all these fragments are active. Now consider a subset V_i with $i > 1$. Each segment $s \in V_i$ is partitioned into one or more fragments by cutting it at every intersection point of s with a previously created active fragment. Let X_i be the set of fragments thus created. We need to decide which fragments in X_i become active. To avoid making too many fragments active, we will partition X_i into equivalence classes, and we will activate only one fragment from each equivalence class. Equivalence is defined as follows and an example is shown in Figure 2:

▶ **Definition 2.1.** Let f and f' be two fragments in X_i. We say that f and f' are *equivalent*, denoted by $f \equiv f'$, if $f = f'$ or the following two conditions hold.

1. The fragments f and f' do not contain segment endpoints and have their endpoints on the same pair of fragments g, g'.
2. The region $Q(f, f', g, g')$ enclosed by the fragments f, f', g, g' does not contain an endpoint of any segment in V.

Step 2: Creating the contact graph $\mathcal{H}$ and its separator $S_{\mathcal{H}}$. Let $V_{\mathcal{H}}$ contain all the active fragments created in Step 1. For two active fragments f, f we put the edge (f, f') in $E_{\mathcal{H}}$ iff f and f' are in contact—that is, $f \cap f' \neq \emptyset$—and they do not belong to the same segment in V; see Figure 1. Because of our general position assumption, no three fragments from different segments meet in a point, and therefore $\mathcal{H}$ is planar [10, Lemma 2.1].

For each segment $s \in V$, we give one of the fragments we created for s weight $1/n$ and we give the other fragments created for s weight 0. Then we create a weight-balanced separator $S_{\mathcal{H}}$ for $\mathcal{H}$, using a weighted version of the Planar Separator Theorem [12]. This also yields a partition of $\mathcal{H} \setminus S_{\mathcal{H}}$ into balanced parts A and B.

Step 3: Creating the star-based separator S for $\mathcal{G}^{\times}[V]$. For a segment $s \in V$, define $\text{star}(s)$ to be the subgraph of $\mathcal{G}^{\times}[V]$ consisting of s and all its incident edges.

- If $f \in S_{\mathcal{H}}$ contains a segment endpoint then we put $\text{star}(\text{seg}(f))$ into S.
- Else $f \in S_{\mathcal{H}}$ must have its endpoints on some active fragments g, g'. We put $\text{star}(\text{seg}(f))$, $\text{star}(\text{seg}(g))$, and $\text{star}(\text{seg}(g'))$ into S.

We post-process S to make it a set of disjoint stars.

Correctness. To give a bound for the number of active fragments, we consider the active fragments of iteration i as the edges in a planar graph whose vertices are the active fragments of previous iterations. We can then use known bounds for the number of edges in planar graphs. This yields a bound of $O(4^c \cdot n)$ for the total number of active fragments, thus $|S| = O(\sqrt{n})$. It can be shown that for a segment s, that is not part of a star in S, it holds that all its active fragments belong to either A or B, the sets $V_{\mathcal{H}} \setminus S_{\mathcal{H}}$ was partitioned into. We use A and B to create a balanced partition for $V \setminus (\bigcup_{X \in S} X)$. We do this by assigning each segment $s \in V \setminus (\bigcup_{X \in S} X)$ based on the part that its positively weighted fragment f belongs to. We assign s to A^* if $f \in A$ and to B^* if $f \in B$. From the fact that A and B are balanced it follows that A^* and B^* are balanced.

It remains to show that no $a \in A^*$ and $b \in B^*$ intersect. Suppose such an intersection exists and that this intersection happens between fragments f_a and f_b that are part of A and B, respectively. If both f_a and f_b are active it is easy to show that $f_a \in A$ and $f_b \in B$. That they intersect then contradicts that $S_{\mathcal{H}}$ is a proper separator for H. The case when f_a and f_b are not active is harder. We use the fact that for an inactive fragment f_a there must exist an equivalent fragment f'_a, both lying between some active fragments g, g'. It can be shown that $f'_a, g, g' \in A$. Because b intersects f_a it must either: intersect f'_a, g or g', have one of its endpoints in $Q(f_a, f'_a, g, g')$, or intersect f_a at least twice. It can be shown that if the first case were true, then the intersected segment must be in B. Thus all cases yield a contradiction, by the correctness of $S_{\mathcal{H}}$, the definition of equivalence of fragments, or the definition of pseudo-segments, respectively.

References

1. Boris Aronov, Mark de Berg, and Leonidas Theocharous. A clique-based separator for intersection graphs of geodesic disks in $\mathbb{R}^2$. In *Proc. 40th International Symposium on Computational Geometry (SoCG)*, pages 9:1–9:15, 2024. doi:10.4230/LIPIcs.SoCG.2024.9.
2. Timothy M. Chan and Dimitrios Skrepetos. Approximate shortest paths and distance oracles in weighted unit-disk graphs. *J. Comput. Geom.*, 10(2):3–20, 2019. doi:10.20382/JOCG.V10I2A2.
3. Hsien-Chih Chang, Jie Gao, and Hung Le. Computing diameter+1 in truly subquadratic time for unit-disk graphs. *CoRR*, abs/2401.12881, 2024. doi:10.48550/ARXIV.2401.12881.
4. Hsien-Chih Chang, Jie Gao, and Hung Le. Computing diameter+2 in truly-subquadratic time for unit-disk graphs. In *Proc. 40th International Symposium on Computational Geometry (SoCG)*, pages 38:1–38:14, 2024. doi:10.4230/LIPICS.SoCG.2024.38.
5. Panagiotis Charalampopoulos, Pawel Gawrychowski, Yaowei Long, Shay Mozes, Seth Pettie, Oren Weimann, and Christian Wulff-Nilsen. Almost optimal exact distance oracles for planar graphs. *J. ACM*, 70(2):12:1–12:50, 2023. doi:10.1145/3580474.
6. M. de Berg, B. M. P. Jansen, and J. S. K. Lamme. Star-based separators for intersection graphs of c-colored pseudo-segments, 2025. URL: https://arxiv.org/abs/2511.05371, arXiv:2511.05371.

7 Mark de Berg, Hans L. Bodlaender, Sándor Kisfaludi-Bak, Dániel Marx, and Tom C. van der Zanden. A framework for Exponential-Time-Hypothesis-tight algorithms and lower bounds in geometric intersection graphs. *SIAM J. Comput.*, 49:1291–1331, 2020. doi:10.1137/20M1320870.

8 Mark de Berg, Bart M. P. Jansen, and Jeroen S. K. Lamme. Star-Based Separators for Intersection Graphs of c-Colored Pseudo-Segments. In Ho-Lin Chen, Wing-Kai Hon, and Meng-Tsung Tsai, editors, *36th International Symposium on Algorithms and Computation (ISAAC 2025)*, volume 359 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 12:1–12:16, Dagstuhl, Germany, 2025. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.ISAAC.2025.12.

9 Jie Gao and Li Zhang. Well-separated pair decomposition for the unit-disk graph metric and its applications. *SIAM J. Comput.*, 35(1):151–169, 2005. doi:10.1137/S0097539703436357.

10 Petr Hliněný. The maximal clique and colourability of curve contact graphs. *Discrete Applied Mathematics*, 81(1):59–68, 1998. doi:doi.org/10.1016/S0166-218X(97)00075-9.

11 Hung Le and Christian Wulff-Nilsen. Optimal approximate distance oracle for planar graphs. In *Proc. 62nd IEEE Symposium on Foundations of Computer Science (FOCS 2021)*, pages 363–374. IEEE, 2021. doi:10.1109/FOCS52979.2021.00044.

12 Richard J. Lipton and Robert Endre Tarjan. A separator theorem for planar graphs. *SIAM Journal on Applied Mathematics*, 36(2):177–189, 1979. doi:10.1137/0136016.

13 Mikkel Thorup and Uri Zwick. Approximate distance oracles. *J. ACM*, 52(1):1–24, 2005. doi:10.1145/1044731.1044732.

Rupture-Isolation for the Weak Graph Distance

Maike Buchin[1], Wolf Kißler[1], and Fabian Kubon[1]

1 **Ruhr University Bochum**
{maike.buchin,wolf.kissler,fabian.kubon}@rub.de

Abstract

The weak graph distance is a distance measure for immersed graphs which is NP-complete to decide. We present a new XP-time algorithm to decide it, which is based on vertices that we can initially disregard, called rupture-isolating vertices. Our approach is to truncate the graph by removing these vertices, and use feedback vertex sets on the truncated graph to achieve an XP-time algorithm.

1 Introduction

Immersed graphs are widely used as representations of geometric networks. Informally, in our notation, an *immersed graph* is a drawing of an abstract graph in $\mathbb{R}^d$ in which edges may cross. Often we are interested in comparing two representations of the same or related networks. In recent years, many different distance measures have been proposed, cf. [2] for a survey of the metric properties of some notable approaches. One such measure is the weak graph distance introduced in [1], which is based on mappings between the graphs and the weak Fréchet distance. Even in 2D it is NP-complete to decide in various scenarios, cf. [3]. For planar embedded graphs in $\mathbb{R}^2$, tractability is known under regularity conditions [1, 3]. Thus, we are interested in tractable decision algorithms for more cases, especially weaker regularity conditions as well as non-planar and higher-dimensional graphs.

Here, we consider a parameterized approach for deciding the weak graph distance between two immersed graphs, based on a regularity condition that we call rupture-isolation. Informally, a vertex v is rupture-isolating if it allows first mapping its neighbors arbitrarily, and then mapping v such that its image is consistent with all images of v's neighbors. We present a truncation algorithm that reduces a graph to non-rupture-isolating parts, and show that these parts can be mapped consistently iff the entire graph can be mapped consistently.

To decide the weak graph distance on the non-rupture-isolating parts, we utilize a feedback vertex set on each connected component and exhaustively try whether any combination of possible images of the feedback vertex set can be extended to the whole connected component. Thus, our algorithm will be in XP, where the exponent is the number of vertices in such a feedback vertex set of a connected component of the truncated graph.

Due to space constraints, proofs are sketched and full proofs will appear in the full version.

Related Work The weak graph distance was first considered by Akitaya et al. [1], and further results on its (in)tractability and variations of it were subsequently found by Buchin, Kißler and Kubon [3]. There are many other distance measures for immersed graphs; the metric properties of some notable measures have been surveyed by Buchin et al. [2].

2 Preliminaries

Let $G_1 = (V_1, E_1)$, $G_2 = (V_2, E_2)$ be straight-line immersed graphs in $\mathbb{R}^d$, i.e., each edge is a line segment that connects its vertices, such that no two edges overlap in a segment of positive length. We further assume that vertices are neither superimposed on vertices nor on

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

edges (of the same graph). Further, let $n_1 = |V_1|$, $m_1 = |E_1|$, $n_2 = |V_2|$, and $m_2 = |E_2|$. We slightly abuse notation and refer to the abstract graphs (resp. their vertices/edges) and their immersions synonymously. We recall the weak Fréchet and weak graph distance:

▶ **Definition 2.1.** Let $s_1, s_2 \colon [0,1] \to \mathbb{R}^d$ be curves. Their *weak Fréchet distance* is defined via $\delta_{wF}(s_1, s_2) := \inf_{\alpha,\beta} \max_{t\in[0,1]} d(s_1(\alpha(t)), s_2(\beta(t)))$, where α, β range over all continuous self-surjections of $[0,1]$ that keep the endpoints fixed and d denotes the Euclidean distance.

▶ **Definition 2.2.** A *graph mapping* is a map $s \colon G_1 \to G_2$ that maps
1. each vertex $v \in V_1$ onto a point $s(v)$ on (the immersion of) an edge of G_2 and
2. each edge $\{u, v\} \in E_1$ onto a simple path from $s(u)$ to $s(v)$ in G_2.

While $s(v)$ might be a vertex, mapping vertices to interior points of edges is allowed as well. We also slightly extend the definition of a simple path: When $s(u)$ (resp. $s(v)$) are interior points of edges, then $s(\{u, v\})$ may begin (resp. end) in these points instead of vertices.

▶ **Definition 2.3.** Let G_1, G_2 be immersed graphs. The *directed weak graph distance* between G_1 and G_2 is defined via $\vec{\delta}_{wG}(G_1, G_2) := \min_{s:G_1\to G_2} \max_{e\in E_1} \delta_{wF}(e, s(e))$, where s ranges over all graph mappings and e, $s(e)$ are the corresponding immersions as curves in $\mathbb{R}^d$.

An undirected (symmetric) version can be defined via the maximum of both directions [1].

Informally, the directed weak graph distance is the maximum distance between an edge and its image path under a globally optimal graph mapping. We consider the decision problem for the directed version, that is, given a threshold ε, we want to decide whether $\vec{\delta}_{wG}(G_1, G_2) \le \varepsilon$.

We denote by $B_\varepsilon(v)$ (resp. $T_\varepsilon(e)$) closed balls (resp. tubes) of radius ε around the vertices (resp. edges) of G_1. Next, we define vertex and edge placements:

▶ **Definition 2.4.** Let G_1, G_2 be immersed graphs. A *vertex placement* of $v \in V_1$ is a connected component of $G_2 \cap B_\varepsilon(v)$. A *(weak) edge placement* of $e = \{u, v\} \in E_1$ is a simple path within $G_2 \cap T_\varepsilon(e)$ that connects a placement of u to a placement of v.

It holds that $\vec{\delta}_{wG}(G_1, G_2) \le \varepsilon$ iff there is a graph mapping $s \colon G_1 \to G_2$ that maps each edge onto an edge placement. We call such a graph mapping a *(graph) placement* of G_1. Akitaya et al. [1] further introduced a general decision algorithm (Algorithm 1). The algorithm is based on the notion of validity of vertex placements. Two placements of adjacent vertices are considered *mutually reachable* if they are connected via an edge placement of their incident edge, and a placement C_v of a vertex v is considered *weakly valid* as long as it has mutually reachable placements of all of v's neighbors, and *invalid* otherwise.

Algorithm 1 General Decision Algorithm to decide whether $\vec{\delta}_{wG}(G_1, G_2) \le \varepsilon$ [1]

1: Compute vertex placements.
2: Compute mutual reachability information for vertex placements.
3: Recursively prune (i.e. delete) weakly invalid placements.
4: Decide if there exists a placement of G_1 (based on the weakly valid vertex placements).

Steps 1-3 can be performed in $O(m_1 m_2)$ time using standard algorithms, cf. [1, Lemmas 3 and 4]. In particular, for an edge $\{u, v\}$, placements of u and v are mutually reachable iff they lie within the same connected component of $G_2 \cap T_\varepsilon(e)$. However, Step 4 is nontrivial (and even NP-complete) in general, cf. [1, 3]. They further analyze special cases in which Step 4 can be solved efficiently. These results fall into two categories: Cases where geometric or combinatorial observations imply that Step 4 is trivial [1, Thm. 1, Lm. 6+7], [3, Cor. 1],

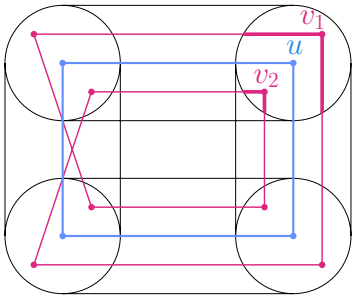


Figure 1 Plane graph G_1 (blue) and immersed graph G_2 (red) such that each vertex of G_1 has two weakly valid placements, but G_1 has no placement

and cases where Step 4 is solved by exhaustive iteration over some search space [1, Thm. 7], [3, Thm. 6+7]. Cf. [3, Table 1] for an overview of various (in)tractability results. Our approach falls into the second category, but is more general in the sense that it works in any dimension and without geometric regularity conditions. However, its running time depends on a parameter whose size is not apparent from the input graphs prior to our preprocessing.

Lastly, we recall the definition of a feedback vertex set, which is a set of vertices whose removal makes a graph acyclic:

▶ **Definition 2.5.** Let $G = (V, E)$ be a graph. A *feedback vertex set* (fvs) F of G is a set $F \subseteq V$ such that the induced graph $G[V \setminus F]$ is acyclic. A *minimum feedback vertex set* is an fvs of minimal size.

While finding a minimum fvs is NP-hard in general, there exist FPT algorithms to find an fvs of bounded size (if one exists), parameterized in the size of the fvs [4].

3 Rupture-Isolating Vertices

In the following, we consider a novel approach for Step 4 of Algorithm 1: Intuitively, the issue with Step 4 is that whenever a vertex on a cycle has multiple weakly valid placements, we do not know whether it is possible to place all edges of the cycle, starting and ending in the same placement. Cf. Fig. 1 for an illustration: If we start placing the cycle at any placement of u, we necessarily end up at the opposite placement of u and hence an inconsistent placement.

Whenever we exclude specific placements from consideration, either by invalidity, by exhaustive search or by observing that they can never be part of a consistent graph mapping, placements of adjacent vertices may become invalid as well, and this may propagate for any number of steps. We call this process a *rupture* of placements. Dealing with ruptures is computationally expensive, since a local choice may influence consistent placeability globally, necessitating a large number of graph explorations. Rupture-isolating vertices are vertices that stop ruptures from propagating from their adjacent vertices through them:

▶ **Definition 3.1.** Let G_1, G_2 be immersed graphs, $v \in V_1$, and let $N(v) = \{w_1, \dots, w_k\}$ be the neighbors of v in G_1. We say that v is *rupture-isolating with respect to* G_2 if for all k-tuples $(C_1, \dots, C_k)$ of weakly valid placements of the w_i in G_2, there is some weakly valid placement C_v of v such that for all i, C_v and C_i are mutually reachable.

Cf. fig. 2: The neighbors of v_1 have two placements each. Regardless of which of their placements both are mapped onto, v_1 still has at least one placement that allows mapping

both edges consistently. Conversely, v_5 is not rupture-isolating, since it has no placement that allows both mapping v_1 onto its topmost placement and v_4 onto its inner placement. Before we discuss how to identify rupture-isolating vertices, we introduce some useful notation:

▶ **Definition 3.2.** Let G_1, G_2 be immersed graphs and let $v \in V_1$. We denote the set of all weakly valid placements of v by $\mathcal{C}(v)$. For an edge $e = \{u, v\}$ of G_1, we define a *placement component* as a connected component I of $G_2 \cap T_\varepsilon(e)$ that contains at least one weakly valid placement of u and v each. We denote the set of all placement components of e as $\mathcal{I}(e)$.

We refer to placements (resp. placement components) of a vertex v (resp. edge e) by C_v (resp. I_e). Next, we consider how to decide whether a vertex is rupture-isolating. If it has only one placement, it is always rupture-isolating. For the general case, we state:

▶ **Lemma 3.3.** *Let G_1, G_2 be immersed graphs. After running steps 1-3 of Algorithm 1, we can decide for each $v \in V_1$ in $O(deg(v) \cdot m_2)$ time whether v is rupture-isolating.*

Proof idea. We can consider placements of v's neighbors within a placement component simultaneously, thus considering tuples of placement components instead of placements. Each placement of v belongs to exactly one placement component of each incident edge, so if there are more than $|\mathcal{C}(v)|$ tuples of placement components, v is not rupture-isolating. Otherwise, we iterate over $\mathcal{C}(v)$ and for each C_v, mark the corresponding tuple of placement components (that all contain C_v) "realized". The vertex v is rupture-isolating iff no tuple remains unmarked. ◀

Lemma 3.3 can be repeated to find all rupture-isolating vertices in G_1 in $O(m_1 m_2)$ time.

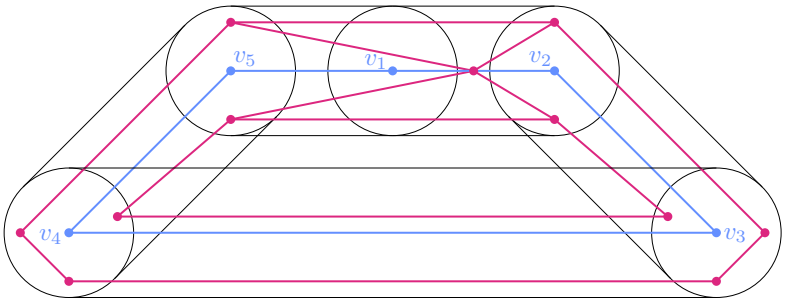


Figure 2 Plane graphs G_1 (blue) and G_2 (red) where initially, only v_1 and v_2 are rupture-isolating, but after disregarding them, v_3 and v_5, and in another step: v_4, become rupture-isolating as well

Like invalidity, rupture-isolation may propagate, cf. fig. 2. Thus, we repeatedly remove rupture-isolating vertices until the resulting graph contains no rupture-isolating vertices:

▶ **Lemma 3.4.** *Algorithm 2 runs in $O(n_1 m_1 m_2)$ time to truncate G_1 w.r.t. G_2 (and ε). Each function f that assigns each vertex from the truncated graph $G_1[V_r]$ a placement such that the assigned placements of adjacent vertices are mutually reachable, can be extended to a graph mapping $s\colon G_1 \to G_2$ that realizes $\vec{\delta}_{wG}(G_1, G_2) \leq \varepsilon$.*

Proof idea. Line 5 runs in $O(m_1 m_2)$ time, which dominates the time needed for each of the $O(n_1)$ iterations before stabilizing. If f places the remaining vertices consistently, by definition of rupture-isolation, we can extend f to also place all vertices that were identified as rupture-isolating in the final iteration consistently. This process can be repeated recursively until G_1 is placed entirely. ◀

Algorithm 2 Truncation algorithm for rupture-isolating vertices in G_1 w.r.t. G_2 (and ε)

1: Apply steps 1-3 from Algorithm 1 on (G_1, G_2, ε).
2: **if** some vertex has no weakly valid placement **then return** $\vec{\delta}_{wG}(G_1, G_2) > \varepsilon$.
3: $V_r \leftarrow V_1$
4: **do**
5: $R' \leftarrow \{v \in V_r \mid v$ is rupture-isolating in $G_1[V_r]$ w.r.t. the placements from line 1$\}$.
6: $V_r \leftarrow V_r \setminus R'$. ▷ Iteratively remove rupture-isolating vertices
7: **while** $R' \neq \varnothing$ ▷ Repeat until V_r stabilizes (no rupture-isolating vertices remain)
8: **return** $G_1[V_r]$. ▷ Return the *truncated graph* induced by the remaining vertices

4 Deciding the Weak Graph Distance using Rupture-Isolation

As seen, we can efficiently truncate G_1 to retain the non-rupture-isolating parts, and it suffices to decide whether the truncated graph can be placed consistently (using the placements initially computed for G_1). For each connected component of the truncated graph this can be decided in isolation. Our approach is to use feedback vertex sets on each connected component: For each tuple of placements of the feedback vertex set's members, mark all other placements of these vertices tentatively invalid and then repeat the pruning step. If all vertices retain a placement, then the connected component can be placed consistently. Conversely, if no such tuple of placements results in all vertices retaining a placement, then the connected component cannot be placed consistently and thus, $\vec{\delta}_{wG}(G_1, G_2) > \varepsilon$.

▶ **Definition 4.1.** Let G_1, G_2 be immersed in $\mathbb{R}^d$. Let $G_t := G_1[V_r]$ denote the *truncated graph* obtained by running Algorithm 2 on G_1, G_2, ε. We say that G_1 is *k-rupture-isolated with respect to G_2* if the feedback vertex number of a connected component of G_t is at most k.

Our parameterized decision algorithm based on k-rupture-isolation is as follows:

Algorithm 3 Deciding the weak graph distance based on rupture-isolating vertices

1: $G_1[V_r] \leftarrow$ apply Alg. 2 on (G_1, G_2, ε) **or return false if** Alg. 2 returns $\vec{\delta}_{wG}(G_1, G_2) > \varepsilon$.
2: **for each** connected component H of $G_1[V_r]$ **do**
3: $F = \{v_1, \ldots, v_k\} \leftarrow$ `ComputeMinimumFVS`(H).
4: **for each** tuple $(C_{v_1}, \ldots, C_{v_k})$ of placements of the $v_i \in F$ **do**
5: **for** $i = 1, \ldots, k$ **do** mark C_{v_i} valid and all other placements of v_i invalid.
6: Recursively prune invalid placements of vertices of H.
7: **if** all $v \in H$ retain a weakly valid placement **then** mark H "successful"; **break**.
8: Undo the pruning steps from line 6.
9: **if** H is not marked "successful" **then return false**.
10: **return true**.

▶ **Theorem 4.2.** *Let G_1, G_2 be immersed graphs and let G_1 be k-rupture-isolated with respect to G_2 and ε, and let $\ell := \max_{v \in V_r} |\mathcal{C}(v)|$ denote the maximum number of placements per vertex of the truncated graph. Further, let $F(n_1, m_1, k)$ denote the time to compute a feedback vertex set of size $\leq k$ in a graph with n_1 vertices and m_1 edges. Then, Algorithm 3 correctly decides whether $\vec{\delta}_{wG}(G_1, G_2) \leq \varepsilon$ in $O(m_1(n_1 m_2 + F(n_1, m_1, k) + \ell^{k+1}))$ time.*

Proof sketch. The correctness of the algorithm follows from Lemma 3.4 as well as the fact that for trees, the existence of a weakly valid placement for each vertex implies the existence

of a graph placement [1]. The time bound follows from Lemma 3.4, Lemmas 3 and 4 in [1] , and since we need to try at most $O(\ell^k)$ tuples per connected component. ◀

Thus, deciding the weak graph distance is in FPT when parameterized in k and ℓ. Further, since $\ell = O(m_2)$, the problem is in XP when parameterized in k. When G_1 is 0-rupture-isolated, i.e., any cycle in G_1 contains at least one rupture-isolating vertex, then G_t is empty, hence Algorithm 3 skips over lines 2-9 and runs in polynomial time. $F(n_1, m_1, k)$ depends on the algorithm used to compute the fvs and is, e.g., $O(3.460^k m_1)$ using [4].

5 Conclusion

We have seen a novel approach to deciding the weak graph distance, based on rupture-isolating vertices that act as "wild cards" for constructing a graph placement. Our method runs in an XP-like time bound when parameterized in the size of an FVS of a connected component of the truncated graph. Possible future work includes refining the method to reduce the exhaustive work and investigating how large the parameter k is for realistic networks.

References

1 Hugo A. Akitaya, Maike Buchin, Bernhard Kilgus, Stef Sijben, and Carola Wenk. Distance measures for embedded graphs. *Computational Geometry*, 95:101743, 2021. doi:10.1016/j.comgeo.2020.101743.

2 Maike Buchin, Erin Chambers, Pan Fang, Brittany Terese Fasy, Ellen Gasparovic, Elizabeth Munch, and Carola Wenk. Distances between immersed graphs: Metric properties. *La Matematica*, 2(1):197–222, 2023. doi:10.1007/s44007-022-00037-8.

3 Maike Buchin, Wolf Kißler, and Fabian Kubon. Hardness and parameterized tractability of the weak graph distance. In Emilio Di Giacomo and Debajyoti Mondal, editors, *WALCOM: Algorithms and Computation*, pages 64–78, Singapore, 2026. Springer Nature Singapore. doi:10.1007/978-981-95-7127-7_5.

4 Yoichi Iwata and Yusuke Kobayashi. Improved analysis of highest-degree branching for feedback vertex set. *Algorithmica*, 83(8):2503–2520, 2021. doi:10.1007/s00453-021-00815-w.

Reconfiguration of Squares Using a Constant Number of Moves Each

Thijs van der Horst[1,2], Maarten Löffler[1], Tim Ophelders*[2], and Tom Peters[2]

1 **Utrecht University, The Netherlands**
[t.w.j.v.d.horst|m.loffler]@uu.nl
2 **TU Eindhoven, The Netherlands**
[t.a.e.ophelders|t.peters1]@tue.nl

Abstract

Multi-robot motion planning is a hard problem. We investigate variants of the problem where square robots are allowed to move only a constant number of times each. We show that the problem remains NP-hard in most cases, except when the squares have unit size and when the problem is unlabeled, i.e. each square can go to an arbitrary target.

Related Version A full version of this paper is available on arXiv: https://arxiv.org/abs/2603.05203.

1 Introduction

Multi-robot motion planning concerns problems where we are given a starting configuration of robots in the plane, as well as a desired target configuration. The goal is to move the robots from their starting location to the target configuration while avoiding collisions. Different variants of this problem have been studied. Except for some trivial cases, solving these problems optimally, minimizing either the makespan or path length of a solution, is hard. In general, it is usually even hard to decide if a target configuration can be reached at all. For a full survey of these results, see [5]. Here, we give a brief overview of the known hardness results.

One of the first hardness results is from 1984, when it was shown that it is PSPACE-complete to decide if a set of arbitrary shaped objects can be reconfigured into a designated target configuration [9], this was then shown to already hold for rectangular objects moving in a rectangular region [10]. The proof was later generalized, and used to show that rectangles of size 1×2 and 2×1 suffice [8]. The related *Rush Hour Problem*, where rectangles are only allowed to slide into one direction, was also proven to be PSPACE-complete [7]. For unit squares in a simple polygon, deciding if a target configuration can be reached is also PSPACE-hard, both if the polygon can have holes [3] as well as when it cannot [1]. For many moving disks it has been shown to be NP-hard to decide if a target configuration is reachable [12], even if the disks have the same size [6].

Abrahamsen and Halperin proposed to consider *monotone* movement schedules: movement schedules where each robot is allowed to move only once [2]. This might reduce the complexity of the problem. In this paper, we investigate the multi-robot motion planning problem for square robots that are allowed to move at most a constant number of times. We show that in most cases, the decision problem remains NP-hard, except for some scenarios with unit size robots in which we can formulate the problem as a flow computation, and

* T. Ophelders is partially supported by the Dutch Research Council (NWO) under project no. VI.Veni.212.260.

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

some trivial scenarios in which the robots are able to move arbitrarily far away and back again.

2 Preliminairies

The *multi-robot motion planning* problem takes a set of robots (or objects) and tries to move them around to create a new configuration, while avoiding collisions. Let a *configuration* be a set of non-overlapping squares. Let a *move* be a square sliding on an arbitrary curve in one continuous motion without rotating. A move is *valid* if it does not collide with any other square. Only a single square is allowed to move at any given time. A *motion schedule* is a sequence of moves. A *valid motion schedule* is a motion schedule where each move is valid. In the *square reconfiguration* problem, we are given a starting configuration S, and a target configuration T with $|S| = |T|$. The goal is to find any valid motion schedule that reconfigures S into T, or decide that such a schedule does not exist.

We consider several variants of this problem, which we call *scenarios*. In the *labeled* scenarios, each square in S has a dedicated target in T. In the *unlabeled* scenarios, each square is allowed to end up at any target in T. In the *bounded* scenarios the squares lie in the interior of a rectilinear polygonal domain with integer coordinates (potentially with holes). The squares are not allowed to intersect the boundary of this domain at any time during a move. In the *unbounded* scenarios, this polygonal domain does not exist and the squares move freely in plane (only colliding with other squares).

We also distinguish between the sizes of the squares. The squares can have a side length of 1 unit, 2 units, or more. In the literature, some papers consider the squares to always have unit side length, and instead allow the squares and the polygonal domain to use non-integer coordinates. These definitions are equivalent. Lastly, we consider the number of moves that a square is allowed to perform. We only consider scenarios where each square is allowed to move at most x times, where x is a constant independent of the number of squares. If each square is allowed to perform only a single move, the problem and resulting movement schedule is also called *monotonic*.

Note that in some scenarios the problem is trivial: in the *unbounded* scenarios in which the number of allowed moves is at least 2, we can simply move all robots arbitrarily far away one by one until our workspace is empty, and then move them back in any different order.

For each scenario, we show that the problem is either NP-hard, or we give a polynomial time algorithm to compute a solution. Our results can be found in Table 1. For the omitted proofs and sections we refer to the full version of this paper.

3 Labeled Monotonic Reconfiguration

In the scenario with labeled squares, where each square is allowed only a single move (monotonic), square reconfiguration is NP-hard. We show this via a reduction from Planar Monotone 3-SAT, which is NP-hard [4]. We will show the reduction for the bounded case. For the unbounded case, note that in any valid reconfiguration schedule, a square needs to move to its target in one move. Hence, if a square already is on its target location, it can never move. We can use this to replace the boundary. The reduction works for squares of arbitrary size. Due to size constraints, the description of the reduction can be found in the full version.

▶ **Theorem 3.1.** *The labeled, monotonic square reconfiguration problem is NP-hard.*

size	# moves 1	2	k
1×1	Thm. 3.1	Thm. 7.1	Thm. 7.1
2×2	Thm. 3.1	Thm. 6.1	Thm. 6.2
$\ell \times \ell$	Thm. 3.1	Thm. 6.1	Thm. 6.2

(a) labeled, bounded

size	# moves 1	2	k
1×1	Thm. 3.1	trivial	trivial
2×2	Thm. 3.1	trivial	trivial
$\ell \times \ell$	Thm. 3.1	trivial	trivial

(b) labeled, unbounded

size	# moves 1	2	k
1×1	Thm. 4.1	Thm. 4.1	Thm. 4.1
2×2	Thm. 5.1	Thm. 6.1	Thm. 6.2
$\ell \times \ell$	Thm. 5.1	Thm. 6.1	Thm. 6.2

(c) unlabeled, bounded

size	# moves 1	2	k
1×1	Thm. 4.1	trivial	trivial
2×2	Thm. 5.1	trivial	trivial
$\ell \times \ell$	Thm. 5.1	trivial	trivial

(d) unlabeled, unbounded

Table 1 An overview of all scenarios and our corresponding results. On the horizontal axis is the number of moves per square (where k is an arbitrary constant). On the vertical axis is the size of the squares (where ℓ is an arbitrary constant). Green cells allow a polynomial time solution. Red cells are NP-hard.

4 Unlabeled 1×1

In the scenario with unlabeled 1×1 squares, the key insight is that whenever a square a wants to move to a target a' via a path π, but it is blocked by another square b, we can take the last such b and move it to a' instead. We can use this to create an algorithm that finds a valid movement schedule if it exists. It creates a graph of all possible square positions and creates a flow-graph out of it, having flow from the starting locations to the target locations. This flow is then used to create a movement schedule. The details can be found in the full version.

▶ **Theorem 4.1.** *The unlabeled, square reconfiguration problem for 1×1 squares has a polynomial time algorithm.*

The algorithm works regardless of if there is a boundary or not, and uses only a single move per square. Hence, if there exists any valid movement schedule, there also exists one that uses only a single move per square.

5 Unlabeled, Big Squares, Monotonic

In the scenario with unlabeled squares that are bigger than 1×1, we will show that the square reconfiguration problem is NP-hard if each square is allowed to move only once, by a reduction from the Hamiltonian Path problem. This problem is NP-hard even for planar directed graphs that have maximum degree 3, and when we designate the start and target vertex of the path that each have degree 1 [11]. Note that each vertex needs at least one incoming and at least one outgoing edge, otherwise the answer is trivial. A vertex with two incoming and one outgoing edge is called a merge vertex, and a vertex with two outgoing and one incoming edge is called a split vertex.

Given an instance of Hamiltonian Path, we will construct a corresponding instance of the unlabeled square reconfiguration problem. We replace all (split and merge) vertices in the graph by (split and merge) vertex gadgets. We replace the edges by edge gadgets, see

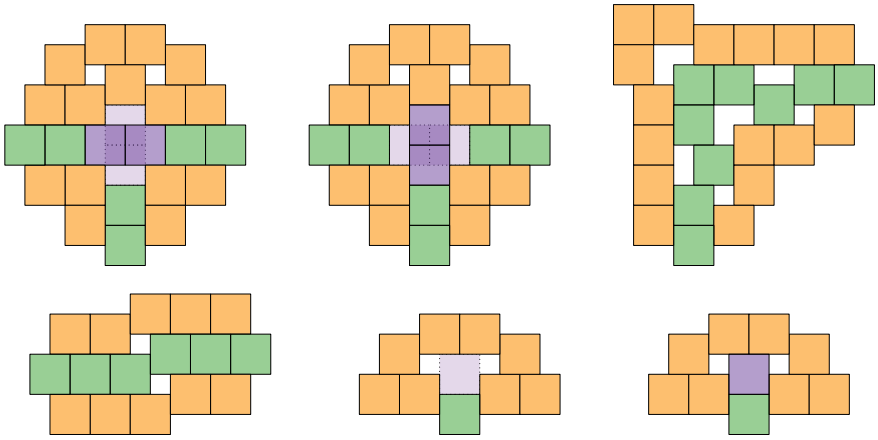


Figure 1 The gadgets for Theorem 5.1. In reading order: the merge, split, corner, wire (with a shift), start, and end gadget. For the merge gadget, the two horizontal edges are incoming. For the split gadget, the two horizontal edges are outgoing. Filled squares are starting locations. The dashed squares are target locations. Both the green as well as the yellow squares cover a start as well as a target square.

Figure 1. Furthermore, we have a start and an end gadget, representing the start and end of the Hamiltonian path.

The gadgets are surrounded by *blocker* squares (yellow in Figure 1). These squares form a boundary and can never move. The green squares represent edges, and the two purple squares represent the vertex. Because of the way the purple vertex squares are positioned, they cannot reconfigure themselves. Hence, the only way to reconfigure the vertex gadgets, is if one of the green squares moves away and makes space. There is a single empty square in the configuration. This empty square will "trace" a Hamiltonian path in the related graph. An example of such a reduction can be seen in Figure 2.

▶ **Theorem 5.1.** *The unlabeled, bounded monotonic square reconfiguration problem with squares of size* 2 *or bigger is NP-hard.*

Note that this construction works for all unbounded scenarios. Hence, the bounded version is also NP-hard, since we can add a large bounding box without affecting the result.

6 Bounded, Big Squares, Multiple Moves

In this section we will show that every scenario of the square reconfiguration problem is NP-hard in which we consider squares of at least 2×2 or bigger, that can make a constant amount of moves k with $k \geq 2$, and when the squares live inside a polygonal domain.

We will first assume that the squares can move only twice. To show that the bounded version of the square reconfiguration problem is NP-hard, we will perform another reduction from Planar Monotone 3-SAT. We replace the variables, clauses, and edges by gadgets as shown in Figure 3. Every square starts on a target (its own in the labeled case), except for one purple *clause checker square* per clause gadget. The squares need to make space for all clause checker squares to move, and then move back to their starting location.

▶ **Theorem 6.1.** *The bounded, square reconfiguration problem with squares of side length at least* 2 *is NP-hard if each square is allowed at most* 2 *moves.*

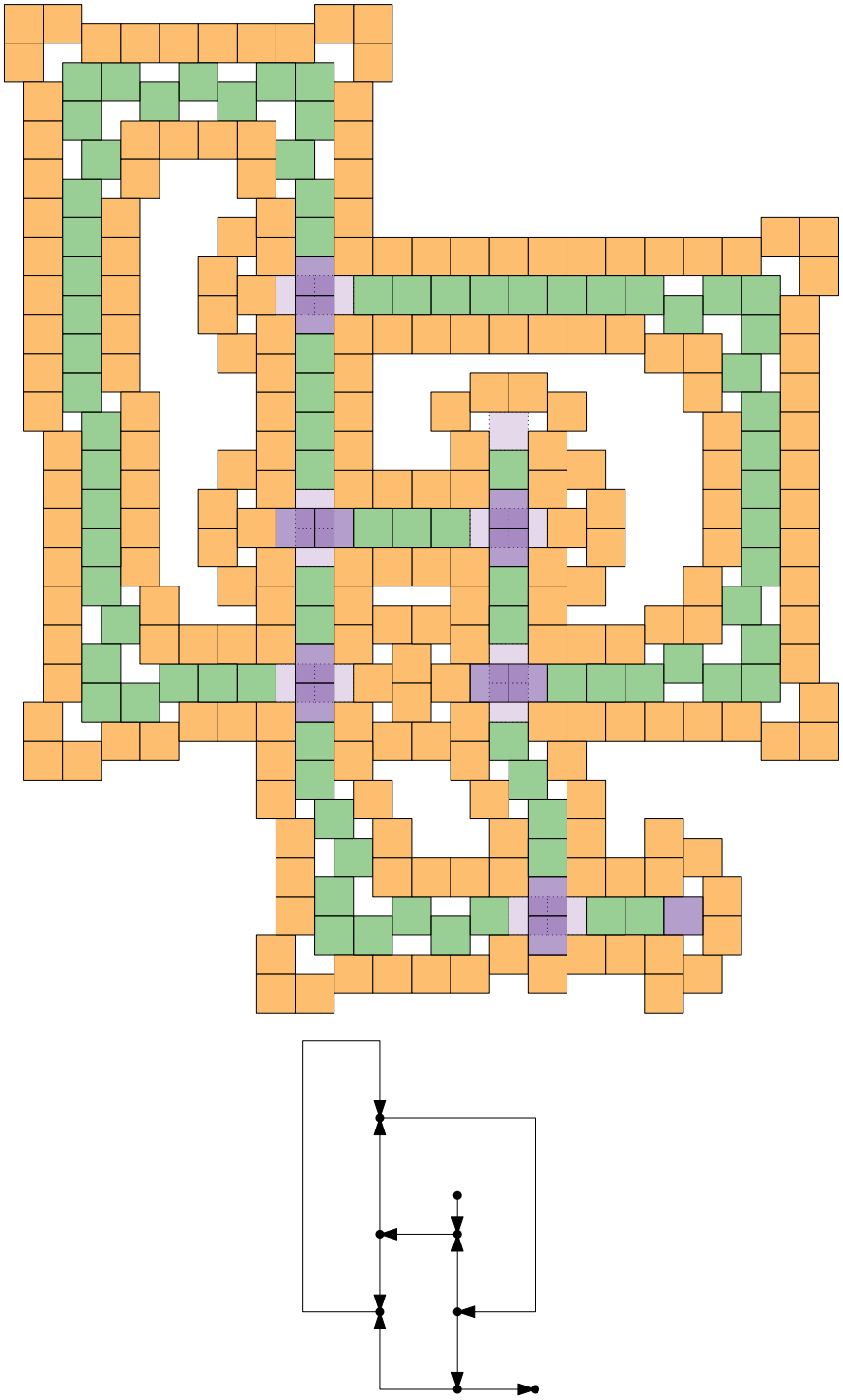


Figure 2 An instance of our Hamiltonian path reduction with a valid movement schedule. Top: The square reconfiguration problem, bottom: The corresponding directed graph.

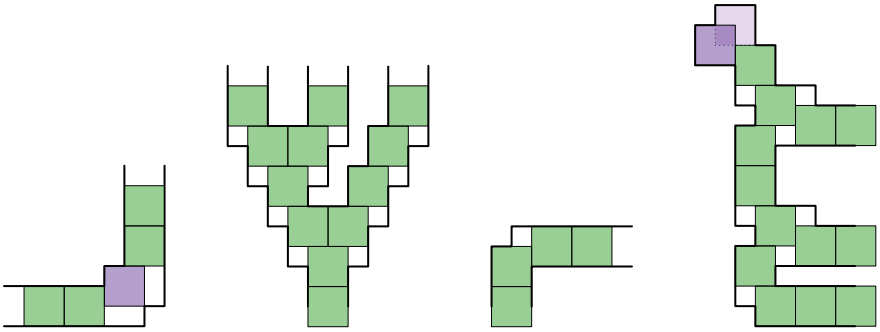


Figure 3 The gadgets for Theorem 6.1. From left to right: Variable gadget, splitter gadget, wire gadget with a corner, and clause gadget. All squares have a target below them, except for the clause checker square.

Once a square is allowed to move more than twice, the current variable gadget can get multiple assignments. We will change the variable gadget when a square is allowed to make k moves, with $k > 2$.

To do so, we use a *latch* gadget, see Figure 4. This gadget can "waste" moves. To be more precise: if we want to move k squares through the latch, the latch needs to move at least k times (or $k-1$ if k is even, since two squares can move through the latch at the same time) and if the latch is allowed to move k times, at most k squares can move through the latch (or $k+1$ if k is odd).

Using this latch gadget we create a variable gadget that can only have a single assignment, see Figure 4. To move either the true or false squares inwards, sufficiently many of the squares below need to move through the latch. The latch can only move k times. We design the variable in such a way that exactly the right amount of squares need to move through the latch for either the true or the false square to be able to move. After they have gone back, the latch has moved k (or $k-2$ depending on the starting location of the latch) times. The latch cannot move enough times to allow the variable to have a different assignment.

▶ **Theorem 6.2.** *The bounded, square reconfiguration problem with squares of side length at least* 2 *is NP-hard if each square is allowed at most k moves, for any constant k.*

7 Bounded, Labeled 1×1, Multiple Moves

The last remaining scenario is when we have squares of size 1×1, each with a predetermined target location, moving inside a polygonal domain. We will again use a reduction from the Hamiltonian Path problem to show NP-hardness. As in Theorem 5.1, an empty square e will trace a path in the configuration. As in Theorem 6.1, the squares need to end on the same location as they started. This way, the reduction works for labeled reconfiguration. The details can be found in the full version.

▶ **Theorem 7.1.** *The bounded, labeled, square reconfiguration problem with squares of side length* 1 *is NP-hard if each square is allowed at most k moves, for any constant $k \geq 2$.*

8 Conclusion

We have shown that deciding if reconfiguration of squares that move only a constant number of times is possible is NP-hard in most scenarios, except for considering unlabeled, unit size

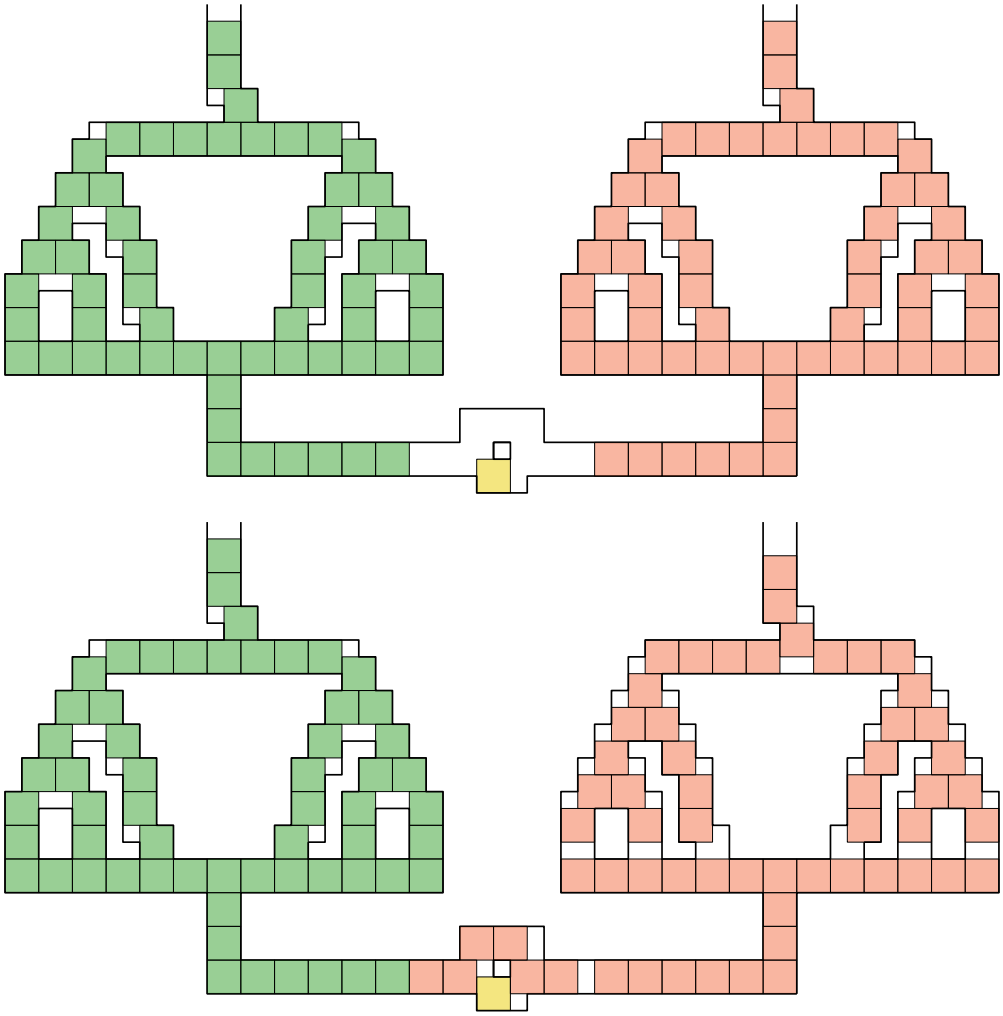


Figure 4 The variable gadget where the yellow latch square can move at most 6 times. Setting the variable to false (bottom) forces the latch square to move twice. Resetting it takes another two moves. Similarly, setting the variable to true takes three moves, and resetting it takes another three.

squares. The algorithms that we gave are not necessarily optimal, we only considered the decision variant of the problem. An interesting question is if optimal movement schedules for these problems can be calculated. Another open question is if these results can be extended to consider other object types, such as disks. Lastly, our reductions use polygonal domains that have holes. It is currently unknown if these results still hold in polygonal domains without holes.

References

1 Mikkel Abrahamsen, Kevin Buchin, Maike Buchin, Linda Kleist, Maarten Löffler, Lena Schlipf, André Schulz, and Jack Stade. Reconfiguration of unit squares and disks: Pspace-hardness in simple settings. In *Proc. Symposium on Computational Geometry (SoCG)*, pages 1:1–1:18, 2025.

2 Mikkel Abrahamsen and Dan Halperin. Ten problems in geobotics. *arXiv preprint arXiv:2408.12657*, 2024.

3 Josh Brunner, Lily Chung, Erik D. Demaine, Dylan H. Hendrickson, Adam Hesterberg, Adam Suhl, and Avi Zeff. 1x1 rush hour with fixed blocks is pspace-complete. In *Proc. 10th International Confernence on Fun wiht Algorithms (FUN)*, pages 7:1–7:14, 2021.

4 Mark de Berg and Amirali Khosravi. Optimal binary space partitions in the plane. In *Computing and Combinatorics (COCOON)*, volume 6196 of *Lecture Notes in Computer Science*, pages 216–225. Springer, 2010.

5 Adrian Dumitrescu. *Motion planning and reconfiguration for systems of multiple objects*. INTECH Open Access Publisher, 2007.

6 Adrian Dumitrescu and Minghui Jiang. On reconfiguration of disks in the plane and related problems. *Computational Geometry*, 46(3):191–202, 2013.

7 Gary William Flake and Eric B Baum. Rush hour is pspace-complete, or "why you should generously tip parking lot attendants". *Theoretical Computer Science*, 270(1-2):895–911, 2002.

8 Robert A Hearn and Erik D Demaine. *Games, puzzles, and computation*. CRC Press, 2009.

9 John E Hopcroft, Jacob Theodore Schwartz, and Micha Sharir. On the complexity of motion planning for multiple independent objects; pspace-hardness of the" warehouseman's problem". *The international journal of robotics research*, 3(4):76–88, 1984.

10 John E Hopcroft and Gordon T. Wilfong. Reducing multiple object motion planning to graph searching. *SIAM Journal on Computing*, 15(3):768–785, 1986.

11 Alon Itai, Christos H Papadimitriou, and Jayme Luiz Szwarcfiter. Hamilton paths in grid graphs. *SIAM Journal on Computing*, 11(4):676–686, 1982.

12 Paul Spirakis and Chee K Yap. Strong np-hardness of moving many discs. *Information Processing Letters*, 19(1):55–59, 1984.

Revisiting Graph Modification via Disk Scaling: From One Radius to Interval-Based Radii*

Thomas Depian[1] and Frank Sommer[2]

1 **Algorithms and Complexity Group, TU Wien, Austria**
`tdepian@ac.tuwien.ac.at`

2 **Institute of Computer Science, Friedrich Schiller University Jena, Germany**
`frank.sommer@uni-jena.de`

Abstract

For a fixed graph class Π, the goal of Π-Modification is to transform an input graph G into a graph $H \in \Pi$ using at most k modifications. Vertex and edge deletions are common operations, and their (parameterized) complexity for various Π is well-studied. Classic graph modification operations such as edge deletion do not consider the geometric aspect of geometric graphs such as unit disk graphs. This led Fomin et al. [ITCS' 25] to initiate the study of disk scaling as a geometric graph modification operation for unit disk intersection graphs: For a given radius r, each modified disk will be rescaled to radius r. We generalize their model by allowing rescaled disks to choose a radius within a given interval $[r_{\min}, r_{\max}]$ and study the (parameterized) complexity (with respect to k) of the corresponding problem Π-Scaling. We show that Π-Scaling is in XP for graph classes Π that can be recognized in polynomial time. Furthermore, we show that Π-Scaling is NP-hard and in FPT for cluster graphs.

Related Version Full Version: `arXiv:2603.05358` [4]

1 Introduction

For a fixed graph class Π, the problem Π-Modification asks whether we can transform a graph G into a graph $H \in \Pi$ using at most k modifications such as vertex or edge deletion, or edge addition. The parameterized complexity (see [3] for an introduction) of Π-Modification when parameterized by k is well-understood [2, 3, 11, 13, 14, 17]. The problem has also been studied for geometric intersection graphs [16], in which each vertex corresponds to a geometric object and two vertices are adjacent if and only if the corresponding objects intersect. However, some of the classic operations ignore the underlying geometry, which raises the question of their appropriateness for geometric intersection graphs. Thus, two "geometry-preserving" modification operations for *unit disk graphs*, where each vertex corresponds to a (closed) disk of radius 1, (and other geometric intersection graphs) were recently studied: disk dispersion [5–7, 9], where disks can move within a bounded radius, and disk scaling [10], where disks can receive a radius different from 1. In the latter model, introduced by Fomin et al. [10], the input contains a non-negative number $\alpha \neq 1$, and each modified disk receives radius α, i.e., all are either *shrunken* ($\alpha < 1$) or *expanded* ($\alpha > 1$). They studied disk-scaling for obtaining an independent set, an acyclic graph, or a connected graph. A limitation of their model is that each scaled disk receives the same radius α. We circumvent this by allowing disks to choose a radius in a given interval $[r_{\min}, r_{\max}]$.

* TD acknowledges support from the Vienna Science and Technology Fund (WWTF) [10.47379/ICT22029], and FS acknowledges support from the Alexander von Humboldt Foundation.

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

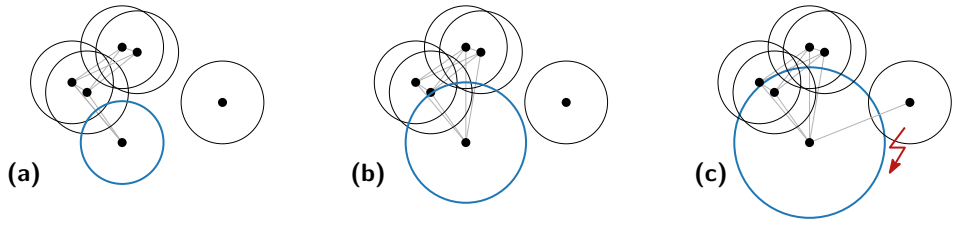


Figure 1 We can turn **(a)** into a cluster graph by scaling the blue disk as shown in **(b)**. However, if we enlarge the disk too much as in **(c)**, the resulting disk graph might no longer be a cluster graph.

Problem Description. We study the problem Scaling To Π (Π-Scaling for short): Given a set $\mathcal{S} \subset \mathbb{R}^2$ of n points, two numbers $0 < r_{\min} \leq r_{\max} \in \mathbb{R}$, and an integer $k \in \mathbb{N}$, is there a set $\mathcal{T} \subseteq \mathcal{S}$ and a *radius assignment* $r \colon \mathcal{S} \to \mathbb{R}_{>0}$ such that $|\mathcal{T}| \leq k$, $r_{\min} \leq r(p) \leq r_{\max}$ for all $p \in \mathcal{T}$ and $r(p) = 1$ otherwise, and $G(\mathcal{S}, r) \in \Pi$, where $G(\mathcal{S}, r)$ is the resulting disk graph. Intuitively, Π-Scaling asks whether one can modify at most k radii (our *budget*) in a given unit disk graph (abbreviated as $G(\mathcal{S}, \mathbb{1})$) to values in $[r_{\min}, r_{\max}]$ such that the resulting disk graph belongs to Π. Therefore, we call the points (disks) in $\mathcal{T}$ *scaled* and all remaining *unscaled*. It is sufficient to ask for r, since we can set $\mathcal{T} = \{p \in \mathcal{S} : r(p) \neq 1\} =: \mathcal{T}(r)$. We observe that many (algorithmic) results by Fomin et al. [10] transfer to our model since for their graph classes it is safe to scale every disks to the same radius $\alpha = r_{\min}$ or $\alpha = r_{\max}$.

Contributions. We first show that Π-Scaling is in XP with respect to k for every graph class Π whose recognition problem can be solved efficiently (Theorem 2.1). Afterwards, we focus on cluster graphs (see the preliminaries for the definition) which are particularly challenging since they can require to choose a radius inside $[r_{\min}, r_{\max}]$ that is different from $r_{\min}$ and $r_{\max}$; see also Figure 1. We show that Scaling To Cluster is NP-hard for any fixed rationals $0 < r_{\min} \leq r_{\max}$ (Theorem 3.2), except for $r_{\min} = 1 = r_{\max}$. Furthermore, we provide an FPT-algorithm parameterized by k (Theorem 3.3), which answers an open question of Fomin et al. [10] affirmatively.

Full proofs of statements marked by ★ can be found in the full version [4].

Preliminaries. Let G be a graph with vertex set $V(G)$ and edge set $E(G)$. For a set $V' \subseteq V(G)$, $G[V']$ denotes the subgraph of G *induced* by V'. For a vertex $v \in V$, $N_G(v) = \{u \in V(G) : uv \in E(G)\}$ denotes the *neighborhood* of v in G. A graph G is a *cluster graph* if every connected component C of G is a clique, also referred to as *cluster*. Equivalently, G is a cluster graph if and only if it does not contain an *induced* P_3, i.e., three vertices $u, v, w \in V(G)$ such that $uv, vw \in E(G)$ but $uw \notin E(G)$. Throughout this abstract, we only consider induced P_3s. Hence, a P_3*-free* graph is a graph without an induced P_3, i.e., a cluster graph. We denote by $\|p - p'\|_2$ the Euclidean distance between two points $p, p' \in \mathbb{R}^2$. We identify disks in $G(\mathcal{S}, r)$ with their center point $p \in \mathcal{S}$. For our algorithmic results, we assume the *Real RAM* model [8, 10].

2 A General XP-Membership Framework

Let Π be a graph class that can be recognized efficiently, i.e., where we can solve Π-Recognition in polynomial time. We show that Π-Scaling is in XP with respect to k by providing a Turing-reduction to $n^{f(k)}$ instances of Constrained Scaling To Graph

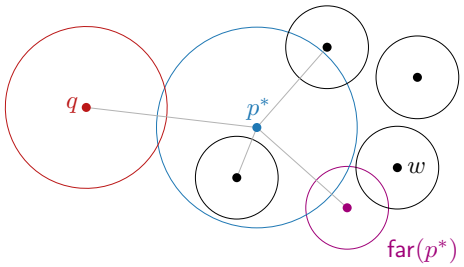


Figure 2 Let $\mathcal{T} = \{p^*, q\}$. The furthest unscaled neighbor $\mathsf{far}(p^*)$ (purple) of p^* (blue) determines if $p^*p' \in E(H)$ for $p' \in \mathcal{S} \setminus \mathcal{T}$. For the edge to q (red), we branch to fix if it exists (as here) or not.

(ConScal), defined as follows. Given a set $\mathcal{S} \subset \mathbb{R}^2$ of n points, a set $\mathcal{T} \subseteq \mathcal{S}$, two real numbers $0 < r_{\min} \leq r_{\max} \in \mathbb{R}$, and a graph $H = (\mathcal{S}, E)$, is there a radius assignment $r\colon \mathcal{S} \to \mathbb{R}_{>0}$ such that $r_{\min} \leq r(p) \leq r_{\max}$ if $p \in \mathcal{T}$ and $r(p) = 1$ otherwise, and $G(\mathcal{S}, r) = H$? ConScal can be seen as a variant of Π-Scaling with a fixed set $\mathcal{T}$ and a known target graph $H \in \Pi$.

To construct these instances, we exhaustively consider all possibilities to construct the set $\mathcal{T}$ of at most k scaled disks and apply the following steps separately for each constructed $\mathcal{T}$– in the remainder of the abstract we refer to this operation as *branching* or *guessing*. Observe that if the radius of the modified disks would be fixed in the input, then the graph H that we obtain after scaling is solely determined by our choice of $\mathcal{T}$. However, since each modified disk can choose a radius from $[r_{\min}, r_{\max}]$, we must further branch to determine which target graph H we want to obtain. To this end, for each possible set $\mathcal{T}$, we determine for the scaled disks their (possible) neighborhood in H by guessing for each $p^* \in \mathcal{T}$ its furthest unscaled neighbor $\mathsf{far}(p^*)$, i.e., $\mathsf{far}(p^*) = \arg\max_{p \in (\mathcal{S} \setminus \mathcal{T}) \cap N_H(p^*)} \|p - p^*\|_2$ (if it exists). This determines all unscaled neighbors of p^*: For $p' \in \mathcal{S} \setminus \mathcal{T}$ we have $p^*p' \in E(H)$ if and only if $\|p^* - p'\|_2 \leq \|p^* - \mathsf{far}(p^*)\|_2$; see Figure 2. We guess the adjacencies among scaled disks. Note that this captures all relevant, i.e., possible, graphs H we could obtain by scaling the disks in $\mathcal{T}$. In particular, each of the $2^{\mathcal{O}(k^2)} \cdot n^{\mathcal{O}(k)}$ branches corresponds to an instance of ConScal. We discard branches with $H \notin \Pi$ and devise for the others a linear program (LP) to solve ConScal (see also the full version [4]): For each $p \in \mathcal{T}$, we have a variable $x_p \in \mathbb{R}$ representing its radius $r(p) = x_p$. Furthermore, we add one "slack-variable" $\varepsilon \geq 0$ to model strict inequalities. We add the constraints $r_{\min} \leq x_p$ and $x_p \leq r_{\max}$ to enforce $r_{\min} \leq r(p) \leq r_{\max}$. Next, we consider the *unscaled neighborhood* $N_H(p) \setminus \mathcal{T}$ of each $p \in \mathcal{T}$. We add the constraint $x_p + 1 \geq \|p - u\|_2$ to ensure $up \in E(G(\mathcal{S}, r))$, for all $u \in N_H(p) \setminus \mathcal{T}$. Similarly, for every disk $u \in \mathcal{S} \setminus (N_H(p) \cup \mathcal{T})$, we add $x_p + 1 \leq \|p - u\|_2 - \varepsilon$ to enforce $up \notin E(G(\mathcal{S}, r))$. This encodes a strict inequality if $\varepsilon > 0$. Finally, we consider every pair $p, q \in \binom{\mathcal{T}}{2}$. If $pq \in E(H)$, we introduce the constraint $x_p + x_q \geq \|p - q\|_2$, and otherwise we add $x_p + x_q \leq \|p - q\|_2 - \varepsilon$. These constraints resemble the definitions of (non-)edges in disk graphs provided $\varepsilon > 0$. Thus, we maximize ε in the optimization function and check if we obtain a solution with $\varepsilon > 0$ (see also the full version [4]). The LP has $k + 1$ variables, $\mathcal{O}(n^2)$ constraints, and can be optimized in $2^{\mathcal{O}(k \cdot \log(k))} \cdot n^2$ time [1]. We conclude:

▶ **Theorem 2.1 (★).** *Let Π be a graph class such that Π-Recognition can be solved in polynomial time. Then, Π-Scaling can be solved in $2^{\mathcal{O}(k^2)} \cdot n^{\mathcal{O}(k)}$ time.*

3 Scaling to Cluster Graphs is NP-Hard and FPT in k

We now consider Π = cluster graphs and sketch in Section 3.1 two reductions that establish NP-hardness of Scaling To Cluster for all $r_{\min}$ and $r_{\max}$ except $r_{\min} = r_{\max} = 1$; see the full version for details [4]. In Section 3.2, we present a fixed-parameter algorithm in k.

3.1 A Sketch of the Hardness Reductions

A point $p \in \mathbb{R}^2$ is *δ-heavy*, denoted p^δ, for an integer $\delta > 1$ if it represents δ co-located *copies* $p^\delta(1)$, $p^\delta(2)$, …, $p^\delta(\delta)$. We define $\mathcal{S}[p] \coloneqq \{p^\delta(i) : 1 \leq i \leq \delta\}$ and call the corresponding disk(s) δ-heavy. A δ-heavy disk induces a δ-clique in every disk graph independent of r. Our main gadget is a *δ-θ-heavy* P_3 with $\theta > k \geq \delta$, or simply *heavy* P_3: Three disks p_1, p_2, p_3 with $\|p_1 - p_2\|_2 = \|p_2 - p_3\|_2 = \xi$, for some $1 < \xi \leq 2$, one disk, usually p_2, is θ-heavy and the remaining two are δ-heavy. They induce $\delta^2 \cdot \theta$-many P_3s and to be P_3-free, we observe:

▶ **Observation 3.1.** *Let $\mathcal{I}$ be an instance of* Scaling To Cluster*, let P be a δ-θ-heavy P_3 in $\mathcal{I}$, and r a solution. We have $|\mathcal{T}(r)| \geq \delta$ and $\mathcal{S}[p] \subseteq \mathcal{T}(r)$ for a δ-heavy disk $p \in V(P)$.*

Hardness for Shrinking Disks ($r_{\min} < 1$). The reduction is inspired by ideas from Fomin et al. for k-Shrinking To Independence [10]. We fix a rational value $0 < r_{\min} < 1$. Let (H, κ) be an instance of Vertex Cover where H is an η-vertex planar cubic graph [12] and let Γ be a planar rectilinear drawing of H of polynomial area [15], scaled by an integer $\lambda(r_{\min})$. We encode edges of H by a series of disks such that the obtained unit disk graph is a suitable subdivision of H. To ensure that an edge $uv \in E(H)$ is covered, we turn the disks along uv into interleaving η^2-θ-heavy P_3s and place, for each $v \in V(H)$, a further θ-heavy disk $p^\theta_{v'}$ next to the disk p_v that represents v; see Figure 3a. We set $k = k_{\text{fix}} + \kappa$, where k_{fix} are the scaling operations required to remove all heavy P_3s within edges of H; recall also Observation 3.1 and that cluster graphs are P_3-free. The heavy P_3s enforce scaling p_u or p_v; otherwise a P_3 remains (along uv) by our choice of k. Removing a P_3 requires shrinking disks, and we can set any $r_{\max} \geq r_{\min}$. Figure 3d shows an example.

Hardness for Enlarging Disks ($1 < r_{\min}$). This reduction follows analogous ideas. Let (H, κ) be an instance of Independent Set where H is an η-vertex planar cubic graph [12]. We represent an edge $uv \in E(H)$ in a (scaled) drawing of H by disjoint η^2-θ-heavy P_3s. By Observation 3.1, we must scale all copies of one endpoint, and their placement requires scaling consistently either their left or their right endpoint; see Figure 3b. We represent each vertex $v \in V(H)$ by a disk p_v, a θ-heavy disk and a 2-heavy disk $p^2_{\neg v}$ that together form a heavy P_3. Thus, we have $p_v \in \mathcal{T}(r)$ or $\mathcal{S}[p_{\neg v}] \subseteq \mathcal{T}(r)$, encoding whether v is in an independent set C of H. Scaling p_u and p_v for $uv \in E(H)$ requires scaling opposite endpoints of heavy P_3s, which is impossible; recall Figure 3b. We set $k = k_{\text{fix}} + \kappa + 2(\eta - \kappa)$, where $2(\eta - \kappa)$ accounts for the $\eta - \kappa$ vertices $v \in V(H) \setminus C$ that require two scaling operations. Enlarging a disk beyond $r_{\min}$ is not helpful and we can set any $r_{\max} \geq r_{\min}$. Figure 3e shows an example.

Combining Both Reductions. The reductions yield NP-hardness for $0 < r_{\min} < 1$ and $1 < r_{\min}$. For $r_{\min} = 1$ and $r_{\max} > 1$, we can modify the P_3s' placement in the construction for $r_{\min} > 1$; see the full version for details [4]. For $r_{\min} = r_{\max} = 1$, it suffices to check if $G(\mathcal{S}, \mathbb{1})$ is P_3-free since our input graph corresponds to a unit disk graph:

▶ **Theorem 3.2 (★).** Scaling To Cluster *is* NP*-hard for every fixed rationals $0 < r_{\min} \leq r_{\max}$ except for the case $r_{\min} = r_{\max} = 1$ where it can be solved in polynomial time.*

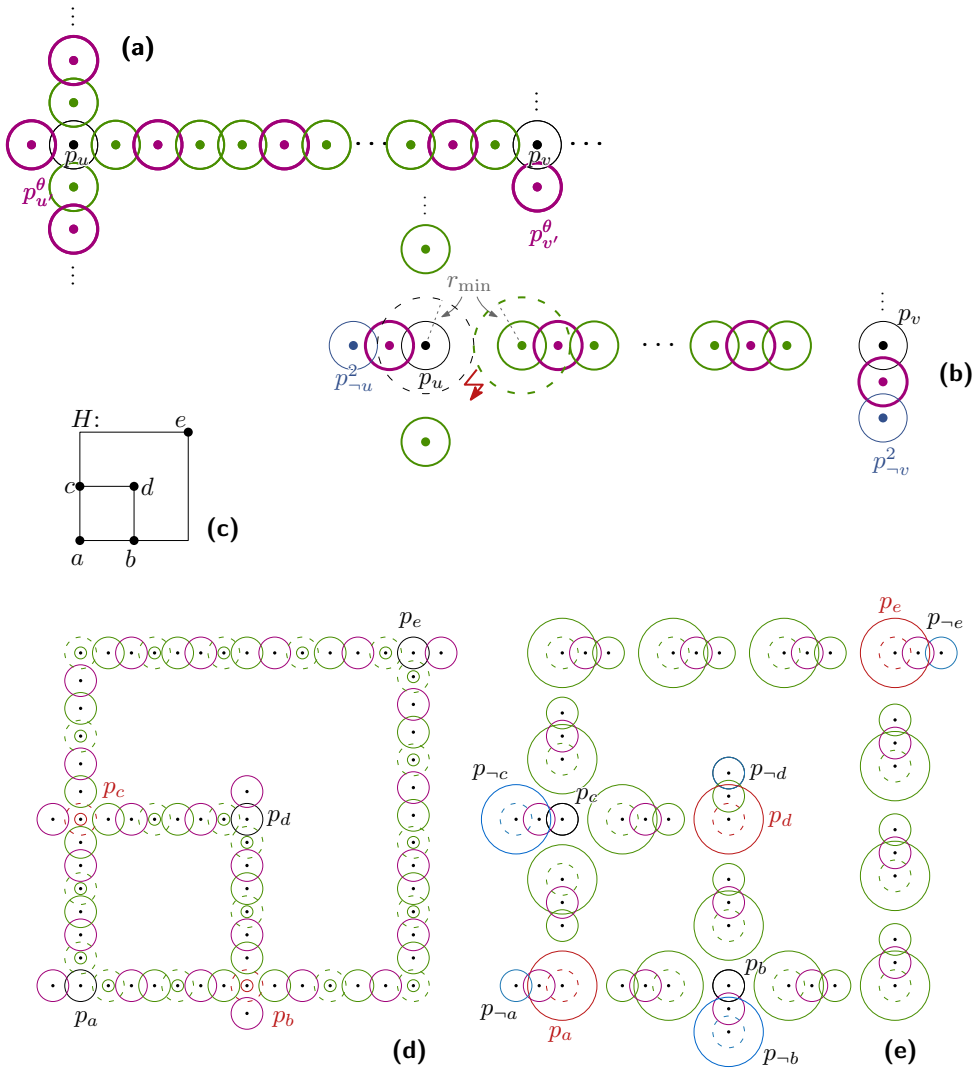


Figure 3 The edge uv for **(a)** $r_{\min} < 1$ and **(b)** $r_{\min} > 1$. Black, blue, green, and purple disks are 1-, 2-, η^2-, and θ-heavy, respectively. In **(b)**, if we enlarge p_u and the left endpoint of a heavy P_3, then two θ-heavy disks are in the same connected component but not adjacent as $\theta > k$. **(c)** A graph H and the corresponding instances of Scaling To Cluster for **(d)** $r_{\min} < 1$ and **(e)** $r_{\min} > 1$. If a disk representing a vertex is red, then the vertex belongs to a solution to the instance **(d)** $(H, 2)$ of Vertex Cover and **(e)** $(H, ^3)$ of Independent Set. Note that we can always slightly shift some disks to make space along an edge of H if required. We simplified H and the constructed instances to ease readability.

3.2 A Fixed-Parameter Tractable Algorithm for Scaling To Cluster

We follow the strategy of the XP-approach behind Theorem 2.1, but use structural properties of cluster graphs to obtain an FPT-running time. We provide in Figure 4 an overview of the algorithm; see the full version [4] for details. Let $\mathcal{I} = (\mathcal{S}, r_{\min}, r_{\max}, k)$ be an instance of Scaling To Cluster and r a hypothetical solution. The algorithm has two phases. In the first phase, we determine the disks in $\mathcal{T}(r)$. For each disk $p \in \mathcal{T}(r)$, we also determine its unscaled neighbors in $G(\mathcal{S}, r)$ via additional branching. In Phase 2, we resolve the (non-)edges between the disks in $\mathcal{T}(r)$ and use the LP from Section 2 to check if suitable radii exist.

Phase 1. We compute in polynomial time a maximal vertex-disjoint P_3 packing $\mathcal{P}$ of $G(\mathcal{S}, \mathbb{1})$, that is, $\mathcal{P}$ is a set of vertex-disjoint P_3s of $G(\mathcal{S}, \mathbb{1})$ such that no further vertex-disjoint P_3 of $G(\mathcal{S}, \mathbb{1})$ can be added to $\mathcal{P}$. By P we denote the set of points in $\mathcal{S}$ whose corresponding disks are contained in at least one $P_3 \in \mathcal{P}$. Every P_3 requires at least one scaling operation, and we can output no if $|\mathcal{P}| \geq k + 1$. Moreover, $G' := G((\mathcal{S} \setminus P), \mathbb{1})$ is a cluster graph. We create a complete edge-colored auxiliary graph H on the vertex set $\mathcal{S}$. The colors model the connectivity after some scaling operations and represent edges (blue), non-edges (red), and (still) undefined edges between scaled disks that we resolve in Phase 2 (green). More precisely, let $\mathcal{T}_i$ denote the set of the first i disks we identified for scaling. The graph H_i encodes a potential disk graph $G(\mathcal{S}, r)$ with $\mathcal{T}(r) = \mathcal{T}_i$; note that we only determine in the end if such a radius assignment r exists. If H_i contains a "fully defined" P_3 X, i.e., two edges, one non-edge, and no undefined vertex-pair in $V(X)$, we guess an involved disk to scale next.

Towards constructing H_i, let $\mathsf{far}(p)$ and $\mathsf{clo}(p)$ denote the furthest *unscaled* disk in the *same* cluster as p and the closest *unscaled* disk in a *different* cluster than p in the solution, respectively. We determine them via branching: Let W be the k closest disks to p. Due to budget k, at least one disk in W remains unscaled. Let $\mathcal{C}$ be the set of at most k clusters of G' which host the disks in W. If the cluster in the solution which hosts p contains disks of a cluster C of G' that "merges" into it, we have $C \in \mathcal{C}$. Moreover, if $\mathsf{far}(p)$ exists and is not part of the P_3 packing $\mathcal{P}$, it must be among the k furthest disks in one of the clusters of $\mathcal{C}$ due to the budget k, yielding overall $\mathcal{O}(k^2)$ choices for $\mathsf{far}(p)$. After determining $\mathsf{far}(p)$, $\mathsf{clo}(p)$ has to be among the k closest disks q to p such that $\|\mathsf{far}(p) - p\|_2 < \|q - p\|_2$, due to budget k. The disks $\mathsf{far}(p)$ and $\mathsf{clo}(p)$ allow us to fix the unscaled neighbors of p. However, there may exist disks q with $\|\mathsf{far}(p) - p\|_2 < \|q - p\|_2 < \|\mathsf{clo}(p) - p\|_2$. Since $\mathsf{far}(p)$ and $\mathsf{clo}(p)$ are *unscaled* disks, we must scale each such disk q; see Figure 4b–d. We maintain a set N_i that keeps track of such disks and update H_i accordingly. If $|\mathcal{T}_i \cup N_i| > k$, we abort the branch.

Phase 2. We resolve vertex-pairs where both endpoints are scaled disks, i.e., the undefined edges, and distinguish between scaled disks with $\mathsf{far}(p) \neq \emptyset$ and $\mathsf{far}(p) = \emptyset$. The former can be resolved in polynomial time and the latter implies that p is in a cluster with scaled disks only, where we guess a clustering in $2^{\mathcal{O}(k \cdot \log(k))} \cdot n^{\mathcal{O}(1)}$ time. We use the LP from Section 2 to check if suitable radii exist.

▶ **Theorem 3.3 (★).** Scaling To Cluster *can be solved in* $2^{\mathcal{O}(k \cdot \log(k))} \cdot n^{\mathcal{O}(1)}$ *time.*

4 Concluding Remarks

We extended the study of disk scaling as a geometric-aware modification operation for unit disk graphs, with a particular focus on cluster graphs. In the full version [4], we furthermore consider Π-Scaling for complete and connected graphs. Two natural open questions remain:

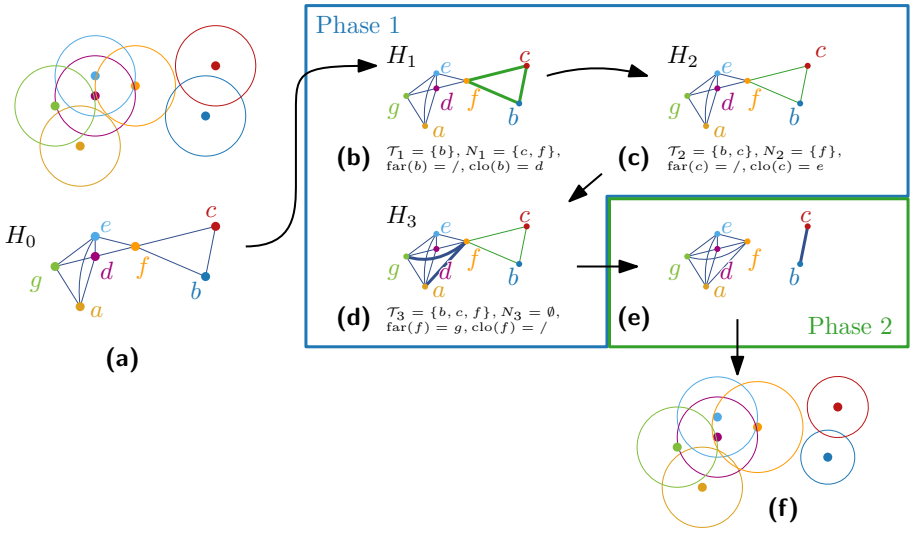


Figure 4 Overview of our FPT-algorithm. **(a)** Input instance and the edge-colored complete graph $H_0 = G(\mathcal{S}, r)$ (we omit red edges for clarity). **(b)**–**(d)** In Phase 1, we iteratively identify disks to scale (sets $\mathcal{T}_i$) and refine our edge-colored complete graph H_i (changes highlighted in bold). **(e)** In Phase 2, we fix a compatible cluster graph H, and check if it can be realized. **(f)** Solution to **(a)**.

(i) whether Scaling To Cluster is in NP and (ii) whether our results can be extended to the optimization variant where we scale k disks and minimize the total amount of scaling, i.e., $\sum_{p \in \mathcal{S}} |1 - r(p)|$, as also studied by Fomin et al. [10]. Note that we cannot simply adapt the LP objective function due to our (maximized) "slack-variable" ε.

References

1 Timothy M. Chan. Improved deterministic algorithms for linear programming in low dimensions. *ACM Transactions on Algorithms (TALG)*, 14(3):30:1–30:10, 2018. doi:10.1145/3155312.

2 Christophe Crespelle, Pål Grønås Drange, Fedor V. Fomin, and Petr A. Golovach. A survey of parameterized algorithms and the complexity of edge modification. *Computer Science Review*, 48:100556, 2023. doi:10.1016/J.COSREV.2023.100556.

3 Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 2015. doi:10.1007/978-3-319-21275-3.

4 Thomas Depian and Frank Sommer. Revisiting graph modification via disk scaling: From one radius to interval-based radii, 2026. doi:10.48550/ARXIV.2603.05358.

5 Nicolás Honorato Droguett, Kazuhiro Kurita, Tesshu Hanaka, and Hirotaka Ono. Algorithms for optimally shifting intervals under intersection graph models. In Bo Li, Minming Li, and Xiaoming Sun, editors, *Proc. 18th International Joint Conference on Frontiers in Algorithmics (IJTCS-FAW'24)*, volume 14752 of *Lecture Notes in Computer Science*, pages 66–78. Springer, 2024. doi:10.1007/978-981-97-7752-5_5.

6 Nicolás Honorato Droguett, Kazuhiro Kurita, Tesshu Hanaka, and Hirotaka Ono. On the complexity of minimising the moving distance for dispersing objects. In Pat Morin and Eunjin Oh, editors, *Proc. 19th International Symposium on Algorithms and Data Structures (WADS'25)*, volume 349 of *LIPIcs*, pages 36:1–36:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2025. doi:10.4230/LIPICS.WADS.2025.36.

7 Nicolás Honorato Droguett, Kazuhiro Kurita, Tesshu Hanaka, Hirotaka Ono, and Alexander Wolff. Further results on rendering geometric intersection graphs sparse by dispersion. In *Proc. 20th International Conference and Workshops on Algorithms and Computation (WALCOM'26)*, 2025. To appear. doi:10.48550/ARXIV.2509.209.

8 Jeff Erickson, Ivor van der Hoog, and Tillmann Miltzow. Smoothing the gap between NP and ER. *SIAM Journal on Computing*, 53(6):S20–102, 2024. doi:10.1137/20M1385287.

9 Fedor V. Fomin, Petr A. Golovach, Tanmay Inamdar, Saket Saurabh, and Meirav Zehavi. Kernelization for spreading points. In Inge Li Gørtz, Martin Farach-Colton, Simon J. Puglisi, and Grzegorz Herman, editors, *Proc. 31st European Symposium on Algorithms (ESA'23)*, volume 274 of *LIPIcs*, pages 48:1–48:16. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2023. doi:10.4230/LIPICS.ESA.2023.48.

10 Fedor V. Fomin, Petr A. Golovach, Tanmay Inamdar, Saket Saurabh, and Meirav Zehavi. Parameterized geometric graph modification with disk scaling. In Raghu Meka, editor, *Proc. 16th Innovations in Theoretical Computer Science (ITCS'25)*, volume 325 of *LIPIcs*, pages 51:1–51:17. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2025. doi:10.4230/LIPICS.ITCS.2025.51.

11 Fedor V Fomin, Daniel Lokshtanov, Saket Saurabh, and Meirav Zehavi. *Kernelization: theory of parameterized preprocessing*. Cambridge University Press, 2019. doi:10.1017/9781107415157.

12 Michael R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, 1979.

13 Christian Komusiewicz. Tight running time lower bounds for vertex deletion problems. *ACM Transactions on Computation Theory (ToCT)*, 10(2):6:1–6:18, 2018. doi:10.1145/3186589.

14 John M. Lewis and Mihalis Yannakakis. The node-deletion problem for hereditary properties is NP-complete. *Journal of Computer and System Sciences (JCSS)*, 20(2):219–230, 1980. doi:10.1016/0022-0000(80)90060-4.

15 Yanpei Liu, Aurora Morgana, and Bruno Simeone. A linear algorithm for 2-bend embeddings of planar graphs in the two-dimensional grid. *Discrete Applied Mathematics*, 81(1–3):69–91, 1998. doi:10.1016/s0166-218x(97)00076-0.

16 Jie Xue and Meirav Zehavi. Parameterized algorithms on geometric intersection graphs. *Computer Science Review*, 58:100796, 2025. doi:10.1016/J.COSREV.2025.100796.

17 Mihalis Yannakakis. Edge-deletion problems. *SIAM Journal on Computing*, 10(2):297–309, 1981. doi:10.1137/0210021.

The Parameterized Complexity of Geometric 1-Planarity*

Alexander Firbas

TU Wien, Austria
afirbas@ac.tuwien.ac.at

Abstract

A graph is *geometric 1-planar* if it admits a straight-line drawing where each edge is crossed at most once. We provide the first systematic study of the parameterized complexity of recognizing geometric 1-planar graphs. By substantially extending a technique of Bannister, Cabello, and Eppstein, combined with Thomassen's characterization of 1-planar embeddings that can be straightened, we show that the problem is fixed-parameter tractable when parameterized by treedepth. Furthermore, we obtain a kernel for Geometric 1-Planarity parameterized by the feedback edge number ℓ. As a by-product, we improve the best known kernel size of $\mathcal{O}((3\ell)!)$ for 1-Planarity [Bannister, Cabello, Eppstein; JGAA 2018] and k-Planarity [Gima, Kobayash, Okada; GD 2025] under the same parameterization to $\mathcal{O}(\ell \cdot 8^{\ell})$. Our approach naturally extends to Geometric k-Planarity, yielding a kernelization under the same parameterization, albeit with a larger kernel. Complementing these results, we provide matching lower bounds: Geometric 1-Planarity remains NP-complete even for graphs of bounded pathwidth, bounded feedback vertex number, and bounded bandwidth.

Related Version Full version available at arXiv:2602.09978.

1 Introduction

A graph is *1-planar* if it admits a drawing in which every edge is crossed at most once; it is *geometric 1-planar* if such a drawing can be realized with straight-line edges. Recognizing 1-planar graphs is NP-complete under various restrictions; see the survey by Kobourov, Liotta, and Montecchiani [14]. Despite this hardness, the parameterized complexity of 1-Planarity is comparatively well understood. In their influential work, Bannister, Cabello, and Eppstein [4] analyzed structural parameterizations, giving an essentially tight classification of which parameters yield fixed-parameter tractability (e.g., treedepth, feedback edge number) and which retain NP-completeness (e.g., bandwidth, pathwidth). This has been recently extended to k-Planarity [10], and there is complementary work that studies orthogonal parameterizations, such as completion from partially predrawn instances [8] and the total number of crossings [11].

In contrast, the geometric setting exhibits different combinatorial and algorithmic behavior. Geometric 1-planar graphs on n vertices have at most $4n - 9$ edges (tight for infinitely many $n \geq 8$) [6]. By comparison, topological 1-planar graphs admit up to $4n - 8$ edges (tight for $n \geq 12$) [14]. Hence, not every 1-planar graph is geometric 1-planar.

From a complexity perspective, there is strong evidence that the geometric variant is strictly harder: for every fixed $k \geq 1$, Geometric k-Planarity is NP-hard [19, 20], and for $k = 867$ it is $\exists\mathbb{R}$-complete [20], implying that, for this fixed k, the problem is not even in NP unless $\exists\mathbb{R} = \mathsf{NP}$. In contrast, the topological counterpart k-Planarity is trivially in NP for every fixed k.

* The author acknowledges support from the Vienna Science and Technology Fund (WWTF) [10.47379/ICT22029] and the Austrian Science Fund (FWF) [10.55776/Y1329], and thanks Robert Ganian for helpful discussions.

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

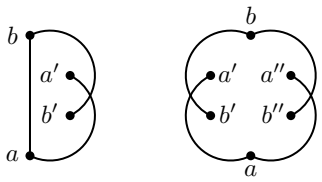


Figure 1 Thomassen's B (left) and W (right) configurations [21].

A central tool is Thomassen's straightening characterization: a 1-planar embedding can be straightened if and only if it contains no B- or W-configurations (see Figure 1). This result was first conjectured by Eggleton [7], proved by Thomassen in 1988 [21], and later rediscovered by Hong et al. [12], who also showed the B/W configurations can be detected in linear time, which implies that the recognition problem Geometric 1-Planarity is in NP. No straightening characterization is known for 2-planar embeddings; moreover, any such characterization would have to be infinite [21]. For 3-planar embeddings, in a certain natural sense, no such characterization exists [15]. If one relaxes topological equivalence and preserves only the set of crossing pairs in a 1-planar embedding, a different characterization is available [13].

Two algorithmic applications enabled by Thomassen's characterization are worth noting. First, every triconnected 1-planar graph admits a drawing that is straight-line except for one bent edge on the outer face [3]. Second, IC-planar graphs (1-planar graphs admitting an embedding where no vertex is incident to two crossings) can be drawn straight-line in linear time [5].

Beyond recognition and concrete drawing algorithms, there is also a quantitative perspective on straight-line drawings. The *rectilinear local crossing number* $\overline{\mathrm{lcr}}(\cdot)$ measures how many crossings per edge are required to draw a graph with straight-line edges [18]. It is known exactly for all complete graphs [2] and for most complete bipartite graphs [1].

2 Contributions

As our first result, we obtain:

▶ **Theorem 2.1 (★).** *Let G be a graph on n vertices with treedepth at most d. Then Geometric 1-Planarity can be decided in time*

$$\mathcal{O}\big(2^{2^{2^{2^{\mathcal{O}(d)}}}} \cdot n^{\mathcal{O}(1)}\big).$$

When deriving a treedepth-FPT algorithm, one essentially aims, given some set of vertices whose removal disconnects the graph (sometimes called a *modulator*) to bound the number of components that connect to the remaining graph only via this set by a function depending only on the treedepth.

For topological 1-planarity, the algorithm of Bannister, Cabello, and Eppstein [4] essentially considers two cases: First, if the modulator has at least three vertices, each component induces a "claw" graph $K_{1,3}$ connected to the modulator. Since bounded treedepth implies that paths are of bounded length, each claw has bounded size. Hence, by Kuratowski's theorem, too many such components will create too many crossings, contradicting 1-planarity. Second, if the modulator has size two, again from the fact that the path length is bounded, if there are more than a certain number of such components, one can show that the modulator vertices need to

be drawn in a shared region, i.e., one can draw an uncrossed line between them. Thus, the instance decomposes into independent instances: if there is a way to draw each component with the two modulator vertices in the same region, one can "glue" them back together. The case of modulator size one does not need to be considered, since in the topological case, via a simple "gluing" argument, one can treat each biconnected component (i.e., each *block*) separately.

The correctness of gluing hinges on one important assumption: for a topological 1-planar drawing, the choice of the outer region is immaterial. I.e., two 1-planar drawings can always be glued at a shared vertex without creating new crossings by selecting the outer regions of the subdrawings accordingly. For geometric 1-planarity, this freedom of choice of the outer region vanishes. Indeed, the straightening characterization of Thomassen [21] is sensitive to the choice of outer region.

As we cannot assume biconnectivity, our algorithm has two stages.

(i) *Inside blocks*, we process the graph along a treedepth decomposition. The *3-modulator case* works as in the topological setting. For *2-modulators*, we process bottom–up at each node, grouping children by their common two-vertex attachment. We retain only a small baseline number of such children so that any valid solution must place the two attachment vertices on a shared region. Among the remaining children we decide by brute force which ones admit a drawing with those two vertices on the *outer* region; these pieces are "glueable"—by a nontrivial application of Thomassen's characterization they can always be reinserted without creating new crossings—and we discard them. If there are too many of the remaining (non-outer) pieces, some edge would be forced to receive more than one crossing, contradicting 1-planarity; otherwise only a bounded number remain. This bottom–up filtering bounds, as a function of the treedepth, the number of vertices inside each block.

(ii) *Across blocks*, we work on the block–cut tree, again bottom–up. At a cut vertex we examine each child subgraph (the union of blocks below that child) and, by brute force, decide which ones admit a drawing with the cut vertex on the *outer* region; these are glueable at the cut vertex and can be safely deleted. If too many non-outer children remain, a 1-planar drawing would be impossible, so we reject; This processing allows us to bound the maximum degree of the block-cut tree. Since paths are of bounded length, the height of the block-cut tree is bounded as well. In total, this allows us to bound the number of blocks (which in turn have only a bounded number of vertices) solely in terms of the treedepth.

For our next set of results, we consider a parameter incomparable with treedepth, the feedback edge number, also referred to as the cyclomatic number. First, we consider topological 1-planarity. The technique of Bannister, Cabello, and Eppstein for their feedback-edge-number kernel yields a kernel of size $\mathcal{O}((3\ell)!)$, where ℓ is the feedback edge number. Essentially, they decompose the input graph into $\mathcal{O}(\ell)$ degree-2 paths and show that, in any hypothetical solution, once a certain threshold of crossings is exceeded, one finds a *local* configuration of consecutive crossings that can be redrawn to reduce the number of crossings. Thus one obtains a kernel by shortening the degree-2 paths so they do not exceed the threshold. They further show that this kernel size is optimal under this strategy [4, Lemma 10].

We break this barrier using a *global*, rather than *local*, redrawing argument. We also decompose the input graph into degree-2 paths. Then, we order them by length, and observe a qualitative shift once there is a sufficiently large gap: the *long* paths are long enough to interact arbitrarily with the *short* paths, and any two long paths can be redrawn to cross at most once by applying *Reidemeister moves*, a foundational tool in knot theory [17]. We then obtain our kernel by shortening the long paths to the smallest number of edges that still qualify them as "long." Using a recursive bound and an observation that allows us to assume

the shortest path has bounded length, we obtain an $\mathcal{O}(\ell \cdot 8^\ell)$-edge kernel for 1-Planarity.

Moreover, at the cost of a higher base in the exponent, applying Thomassen's characterization, we extend this kernel to the geometric case.

▶ **Theorem 2.2 (★).** *1-Planarity, parameterized by the feedback edge number ℓ, admits a kernel with $\mathcal{O}(\ell \cdot 8^\ell)$ edges. Geometric 1-Planarity, under the same parameterization, admits a kernel with $\mathcal{O}(\ell \cdot 27^\ell)$ edges.*

As an immediate corollary, we obtain an improved kernel for k-Planarity under the same parametrization with $\mathcal{O}(\ell \cdot 8^\ell)$ edges, thereby improving upon the previous best-known kernel of size $\mathcal{O}((3\ell)!)$ due to Gima, Kobayashi, and Okada [10].

Our technique also extends to Geometric k-Planarity, at the expense of a larger kernel size. Due to the lack of a straightening characterization for k-Planarity with $k > 1$, we provide a direct redrawing argument: we triangulate the graph with respect to the short paths and redraw the long paths inside each triangle with few crossings. Thus we obtain:

▶ **Theorem 2.3 (★).** *Geometric k-Planarity, parameterized by the feedback edge number ℓ, admits a kernel with $\mathcal{O}\big(2^{\mathcal{O}(3^\ell \log \ell)}\big)$ edges.*

Finally, we show that our results are essentially tight within the usual parameter hierarchy: Geometric 1-Planarity remains NP-complete even in very restricted settings. We provide a novel reduction from Bin Packing and show:

▶ **Theorem 2.4 (★).** *Geometric 1-Planarity remains NP-complete for instances of pathwidth at most 15 or feedback vertex number at most 48.*

For the parameter bandwidth, we observe that replacing every edge by a constant-size gadget causes only a quadratic blow-up in bandwidth. Combining this with a known reduction of Schaefer [19] lets us lift the known bounded-bandwidth hardness for 1-Planarity [4]:

▶ **Theorem 2.5 (★).** *Geometric 1-Planarity remains NP-complete even when restricted to instances of bounded bandwidth.*

For results marked with (★), we refer to the full version of this paper [9], which includes full proofs and additional figures.

3 Conclusion

We have given a comprehensive set of results: a fixed-parameter algorithm for Geometric 1-Planarity parameterized by treedepth; subfactorial kernels parameterized by the feedback edge number for 1-Planarity, k-Planarity, and Geometric 1-Planarity; a kernel for Geometric k-Planarity under the same parameterization; and matching NP-completeness for bounded pathwidth, feedback vertex number, and bandwidth.

We have improved the kernel for k-Planarity parameterized by the feedback edge number ℓ from $\mathcal{O}((3\ell)!)$ to $\mathcal{O}(\ell \cdot 8^\ell)$. Is a polynomial kernel possible?

A natural next step is to consider the (parameterized) complexity of Geometric k-Planarity. In the topological case, the leap from 1 to k is trivial for treedepth+k and feedback edge number: topological k-planarity reduces to 1-planarity by replacing each edge with a path of length k, which does not blow up these parameters [10]. In the geometric case, however, this is far from trivial, as for $k \geq 2$, no analogue of Thomassen's characterization is known, and is in a certain natural sense even impossible for $k \geq 3$ [15]. For feedback edge

number, we have sidestepped this issue and have given an argument that does not rely on such a characterization.

For treedepth, this seems challenging. As a first step, for triconnected graphs, Geometric k-Planarity parameterized by treedepth$+k$ is easily seen to be FPT: only processing akin to the rule for modulators of size at least three is necessary, and the underlying counting argument can be lifted easily. Does this tractability extend to general graphs (in the *uniform* sense)?[1]

More fundamentally, for $k \geq 2$, the complexity of Geometric k-Planarity is not well understood. It is known that, at some point (unless NP $= \exists\mathbb{R}$), the problem is not even in NP, as Geometric 867-Planarity is $\exists\mathbb{R}$-complete [20]. When does this shift occur? Is Geometric 2-Planarity in NP?

References

1 Bernardo M. Ábrego, Kory Dondzila, Silvia Fernández-Merchant, Evgeniya Lagoda, Seyed Sajjadi, and Yakov Sapozhnikov. On the rectilinear local crossing number of $K_{m,n}$. *J. Inf. Process.*, 25:542–550, 2017. doi:10.2197/ipsjjip.25.542.

2 Bernardo M. Ábrego and Silvia Fernández-Merchant. The rectilinear local crossing number of K_n. *J. Comb. Theory A*, 151:131–145, 2017. doi:10.1016/j.jcta.2017.04.003.

3 Md. Jawaherul Alam, Franz J. Brandenburg, and Stephen G. Kobourov. Straight-line grid drawings of 3-connected 1-planar graphs. In *Graph Drawing (GD 2013)*, volume 8242 of *Lecture Notes in Computer Science*, pages 83–94. Springer, 2013. doi:10.1007/978-3-319-03841-4_8.

4 Michael J. Bannister, Sergio Cabello, and David Eppstein. Parameterized complexity of 1-planarity. *J. Graph Algorithms Appl.*, 22(1):23–49, 2018. doi:10.7155/jgaa.00457.

5 Franz J. Brandenburg, Walter Didimo, William S. Evans, Philipp Kindermann, Giuseppe Liotta, and Fabrizio Montecchiani. Recognizing and drawing IC-planar graphs. *Theor. Comput. Sci.*, 636:1–16, 2016. doi:10.1016/j.tcs.2016.04.026.

6 Walter Didimo. Density of straight-line 1-planar graph drawings. *Inf. Process. Lett.*, 113(7):236–240, 2013. doi:10.1016/j.ipl.2013.01.013.

7 R. B. Eggleton. Rectilinear drawings of graphs. *Utilitas Mathematica*, 29:149–172, 1986. URL: https://zbmath.org/0578.05016.

8 Eduard Eiben, Robert Ganian, Thekla Hamm, Fabian Klute, and Martin Nöllenburg. Extending partial 1-planar drawings. In *ICALP 2020*, volume 168 of *LIPIcs*, pages 43:1–43:19. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPIcs.ICALP.2020.43.

9 Alexander Firbas. The parameterized complexity of geometric 1-planarity, 2026. arXiv:2602.09978.

10 Tatsuya Gima, Yasuaki Kobayashi, and Yuto Okada. Structural parameterizations of k-planarity. In Vida Dujmovic and Fabrizio Montecchiani, editors, *33rd International Symposium on Graph Drawing and Network Visualization, GD 2025, Norrköping, Sweden, September 24-26, 2025*, volume 357 of *LIPIcs*, pages 16:1–16:17. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2025. doi:10.4230/LIPICS.GD.2025.16.

11 Thekla Hamm and Petr Hliněný. Parameterised partially-predrawn crossing number. In *SoCG 2022*, volume 224 of *LIPIcs*, pages 46:1–46:15. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPIcs.SoCG.2022.46.

[1] Existence of a *non-uniform* FPT algorithm is always guaranteed for hereditary properties [16]; in particular, (geometric) k-planarity is hereditary for each fixed k.

12 Seok-Hee Hong, Peter Eades, Giuseppe Liotta, and Sheung-Hung Poon. Fáry's theorem for 1-planar graphs. In *Computing and Combinatorics (COCOON 2012)*, volume 7434 of *Lecture Notes in Computer Science*, pages 335–346. Springer, 2012. doi:10.1007/978-3-642-32241-9_29.

13 Seok-Hee Hong and Hiroshi Nagamochi. Re-embedding a 1-plane graph into a straight-line drawing in linear time. In *Graph Drawing and Network Visualization (GD 2016)*, volume 9801 of *Lecture Notes in Computer Science*, pages 321–334. Springer, 2016. doi:10.1007/978-3-319-50106-2_25.

14 Stephen G. Kobourov, Giuseppe Liotta, and Fabrizio Montecchiani. An annotated bibliography on 1-planarity. *Comput. Sci. Rev.*, 25:49–67, 2017. doi:10.1016/j.cosrev.2017.06.002.

15 Hiroshi Nagamochi. Straight-line drawability of embedded graphs. Technical Report Technical Report 2013-005, Department of Applied Mathematics and Physics, Kyoto University, 2013. URL: https://www.amp.i.kyoto-u.ac.jp/tecrep/ps_file/2013/2013-005.pdf.

16 Jaroslav Nešetřil and Patrice Ossona de Mendez. *Sparsity: Graphs, Structures, and Algorithms*, volume 28 of *Algorithms and Combinatorics*. Springer, 2012. doi:10.1007/978-3-642-27875-4.

17 Kurt Reidemeister. Elementare Begründung der Knotentheorie. In *Abhandlungen aus dem Mathematischen Seminar der Universität Hamburg*, volume 5, pages 24–32. Springer, 1927. doi:10.1007/BF02952507.

18 Marcus Schaefer. The graph crossing number and its variants: A survey. *Electron. J. Combin.*, 2012. doi:10.37236/2713.

19 Marcus Schaefer. Picking planar edges; or, drawing a graph with a planar subgraph. In *Graph Drawing (GD 2014)*, volume 8871 of *Lecture Notes in Computer Science*, pages 13–24. Springer, 2014. doi:10.1007/978-3-662-45803-7_2.

20 Marcus Schaefer. Complexity of geometric k-planarity for fixed k. *J. Graph Algorithms Appl.*, 25(1):29–41, 2021. doi:10.7155/jgaa.00548.

21 Carsten Thomassen. Rectilinear drawings of graphs. *J. Graph Theory*, 12(3):335–341, 1988. doi:10.1002/jgt.3190120306.

Semi-discrete convex order and Laguerre tessellation fitting

David P. Bourne[1], Thomas O. Gallouët[2], Quentin Mérigot[3], and Andrea Natale[4]

1 **Maxwell Institute for Mathematical Sciences and Department of Mathematics, Heriot-Watt University, Edinburgh, UK**
d.bourne@hw.ac.uk

2 **Université Paris-Saclay, CNRS, Inria, Laboratoire de mathématiques d'Orsay, ParMA, 91405, Orsay, France**
thomas.gallouet@inria.fr

3 **Université Paris-Saclay, CNRS, Inria, Laboratoire de mathématiques d'Orsay, 91405, Orsay, France / DMA, École normale superieure, Université PSL, CNRS, 75005 Paris, France / Institut universitaire de France (IUF)**
quentin.merigot@universite-paris-saclay.fr

4 **Université Paris-Saclay, CNRS, Inria, Laboratoire de mathématiques d'Orsay, ParMA, 91405, Orsay, France**
andrea.natale@inria.fr

Abstract

In this work we study the problem of reconstructing a Laguerre tessellation from the volumes and the barycenters of its cells. We show that any vector of barycenters arising from a Laguerre tessellation with prescribed cell volumes can be understood as an exposed point of a convex set defined by convex order relations. Leveraging this characterization, we formulate a projection problem that can be solved efficiently and yields an approximation of the desired reconstruction.

Related Version HAL: hal-05008630

1 Introduction

Laguerre tessellations [3] offer an efficient way to parameterize a large class of convex partitions of Euclidean space using only a set of points, called generators, and scalar weights. Due to their versatility, they have become a popular tool in material science to describe the microstructure of materials such as metals or foams. A common problem in this context consists in computing a Laguerre tessellation with prescribed geometrical properties coming from experimental observations [11, 13, 5]. Motivated by these applications, in this work we will study the following problem, which constitutes a simplified version of a common scenario encountered in practice (see Figure 1 for an illustration):

▶ **Question 1.1.** How can one recover a set of generators and weights associated with a given Laguerre tessellation provided only the volumes and the barycenters of its cells?

To answer this question, we make explicit the connection between cell barycenters obtained from Laguerre tessellations with prescribed cell volumes and semi-discrete optimal transport. We leverage this connection to propose an algorithm that approximately solves the reconstruction problem. Importantly, the proposed approach can be used more generally on experimental data which does not necessarily arise from a Laguerre tessellation.

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

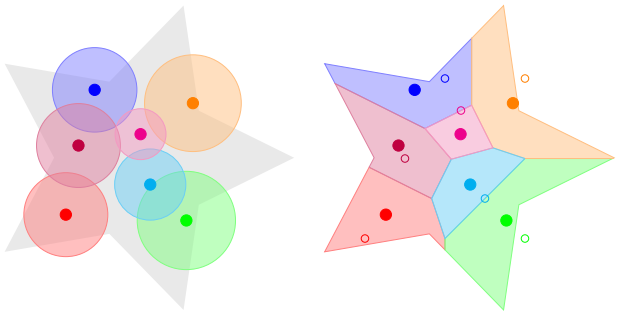


Figure 1 We seek to reconstruct a Laguerre tessellation given its cells' barycenters (i.e., their centroids) and volumes (computed w.r.t. a given density; here constant on a star), represented by the disks on the left. The empty dots on the right are the unknown generators of the tessellation.

2 Preliminaries on semi-discrete optimal transport

Laguerre tessellations Let $Y = (y_1, \dots, y_N) \in (\mathbb{R}^d)^N \setminus \Delta_N$ be a vector of generators, where

$$\Delta_N := \{Y = (y_1, \dots, y_N) \in (\mathbb{R}^d)^N \ ; \ \exists\, 1 \leq i < j \leq N \text{ such that } y_i = y_j\},$$

and let $\phi = (\phi_1, \dots, \phi_N) \in \mathbb{R}^N$ be a vector of weights. The associated Laguerre tessellation is the partition of $\mathbb{R}^d$ into N convex cells $(L_i(\phi, Y))_i$ defined as follows: for $i = 1, \dots, N$,

$$L_i(\phi, Y) := \{x \in \mathbb{R}^d \,;\, \langle x, y_i\rangle - \phi_i \geq \langle x, y_j\rangle - \phi_j \ \forall j \neq i\}.$$

Setting $\phi_i = |y_i|^2/2$, the definition above yields the Voronoi tessellation associated with Y. In general, Laguerre tessellations form a strict subset of the space of all convex partitions of $\mathbb{R}^d$. However, it turns out that for $d \geq 3$ all convex partitions satisfying some mild geometric requirements (specifically, "simple" tessellations) are always Laguerre [2, Theorem 4].

Semi-discrete optimal transport Laguerre tessellations with prescribed cell volumes arise naturally in the theory of optimal transport [4, 12]. Let $\mathcal{P}_1(\mathbb{R}^d)$ denote the set of probability measures μ with finite first moments, i.e., such that $\int |x| \mathrm{d}\mu(x) < \infty$. Let $\rho \in \mathcal{P}_1(\mathbb{R}^d)$ be an absolutely continuous measure. For simplicity, throughout this work we will also assume that ρ has bounded density, although this is not necessary for the results in this section. Given a vector $v = (v_1, \dots, v_N) \in \mathbb{R}^N_{>0}$ of volumes satisfying $\sum_i v_i = 1$, for any $Y \in (\mathbb{R}^d)^N \setminus \Delta_N$, there always exists a vector of weights $\phi^* \in \mathbb{R}^N$ such that

$$\rho(L_i(\phi^*, Y)) = v_i, \quad \forall\, i = 1, \dots, N; \tag{1}$$

see [4, Theorem 3]. To see this, consider the following optimal transport problem:

$$F_N(Y) := \max\left\{ \int_{\mathbb{R}^d \times \mathbb{R}^d} \langle x, y\rangle \,\mathrm{d}\gamma(x, y) \ ; \ \gamma \in \Gamma\left(\rho, \sum_{i=1}^N v_i \delta_{y_i}\right)\right\}, \tag{2}$$

where $\Gamma(\mu, \nu)$ denotes the set of probability measures on $\mathbb{R}^d \times \mathbb{R}^d$ with marginals $\mu, \nu \in \mathcal{P}_1(\mathbb{R}^d)$. Since ρ is absolutely continuous, Brenier's Theorem [14, Section 1.3.1] guarantees that problem (2) admits a unique solution γ, which is characterized as the only transport plan $\gamma \in \mathcal{P}(\mathbb{R}^d \times \mathbb{R}^d)$ satisfying $\gamma = (\mathrm{Id}, \nabla u)_{\#}\rho$ and

$$(\nabla u)_{\#}\rho = \sum_{i=1}^N v_i \delta_{y_i}\,, \tag{3}$$

where $u : \mathbb{R}^d \to \mathbb{R}$ is a convex function known as the Brenier potential, and $\#$ denotes the push-forward of measures. It can be shown that u necessarily takes the form

$$u(x) = \max_{i\in\{1,\dots,N\}} \{\langle x, y_i\rangle - \phi_i\}\,, \tag{4}$$

for an appropriate vector of weights ϕ; see, e.g., [12, Proposition 37]. Consequently, $\nabla u(x) = y_i$, for a.e. $x \in L_i(\phi, Y)$. By expressing condition (3) with this ansatz, one deduces the existence of a vector of weights ϕ^* satisfying (1). Choosing $\phi = \phi^*$ in (4) yields a Brenier potential for the transport problem (2). Moreover, the plan γ^* defined by

$$\int_{\mathbb{R}^d\times\mathbb{R}^d} \varphi(x,y)\,\mathrm{d}\gamma^*(x,y) = \sum_{i=1}^N \int_{L_i(\phi^*,Y)} \varphi(x,y_i)\,\mathrm{d}\rho(x)\,,$$

for all $\varphi \in C_b(\mathbb{R}^d \times \mathbb{R}^d)$, is the unique solution of problem (2).

Well-posedness of the reconstruction problem By disintegrating the plan γ with respect to its second marginal, the function F_N in (2) can be equivalently written as follows:

$$F_N(Y) = \max\left\{ \sum_{i=1}^N \int_{\mathbb{R}^d} \langle x, y_i\rangle v_i \mathrm{d}\theta_i(x) \;\;;\;\; \theta_i \in \mathcal{P}_1(\mathbb{R}^d)\,,\; \sum_{i=1}^N \theta_i v_i = \rho \right\}. \tag{5}$$

By the discussion above, there exists a unique collection of probability measures $(\theta_i)_i$ solving this problem, given by $\theta_i^* = \rho|_{L_i(\phi^*,Y)}/v_i$. Let $b_i^* \coloneqq \mathrm{bary}_\rho(L_i(\phi^*, Y))$, the barycenter of ρ restricted to $L_i(\phi^*, Y)$. As observed in [5], since the objective function only depends on

$$b_i \coloneqq \int x \mathrm{d}\theta_i(x)\,, \tag{6}$$

it follows that no admissible collection of probability measures other than $(\theta_i^*)_i$ satisfies $b_i = b_i^*$. In particular, there exists a unique partition $(P_i)_i$ of $\mathbb{R}^d$ with $v_i = \rho(P_i)$ and $b_i^* = \mathrm{bary}_\rho(P_i)$, and this is precisely the tessellation $(L_i(\phi^*, Y))_i$. However, note that ϕ^* and Y are not uniquely determined from v and B^*. In fact, dilating or translating Y yields the same optimal tessellation. Moreover, other invariances may also hold (see, e.g., Figure 4).

3 Barycenters of Laguerre tessellation and semi-discrete convex order

Let us introduce the set of Laguerre tessellations with given cell volumes, denoted

$$\mathcal{T}^N_{\mathrm{Lag}}(v,\rho) \coloneqq \{(L_i)_{i=1}^N \text{ Laguerre tessellation of } \mathbb{R}^d\,;\, \rho(L_i) = v_i\ \forall i\}\,,$$

and the set of cell barycenters obtained from such tessellations,

$$\mathcal{B}^N_{\mathrm{Lag}}(v,\rho) \coloneqq \{B \in (\mathbb{R}^d)^N \;;\; \exists\,(L_i)_i \in \mathcal{T}^N_{\mathrm{Lag}}(v,\rho) \text{ such that } b_i = \mathrm{bary}_\rho(L_i)\ \forall\, i\}\,.$$

In the following, we will describe in detail the relation between $\mathcal{B}^N_{\mathrm{Lag}}(v,\rho)$ and the set of discrete measures ν dominated in convex order by ρ, denoted by $\nu \preceq_C \rho$. Specifically, we will study the following set (see [6, 7] for the setting where ρ is also discrete):

$$\mathcal{C}_N(v,\rho) \coloneqq \left\{ B \in (\mathbb{R}^d)^N \;;\; \sum_{i=1}^N v_i\delta_{b_i} \preceq_C \rho \right\}.$$

More explicitly, a point $B \in \mathcal{C}_N(v,\rho)$ if and only if

$$\sum_i \varphi(b_i)v_i \le \int_{\mathbb{R}^d} \varphi\,\mathrm{d}\rho\,, \quad \forall\,\varphi : \mathbb{R}^d \to \mathbb{R} \text{ convex.}$$

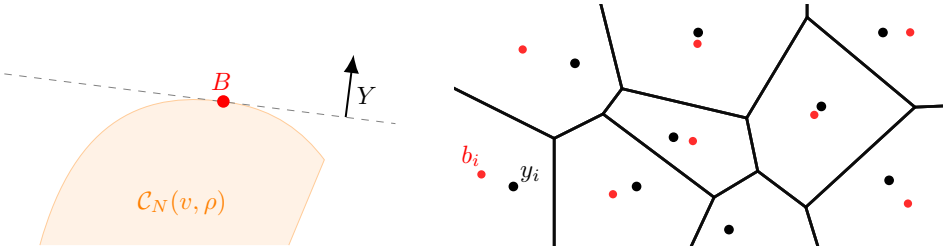


Figure 2 The vector of barycenters $B = (b_i)_i$ of a Laguerre tessellation with generators $Y = (y_i)_i$ and volumes $v = (v_i)_i$ (right figure) is the unique maximizer of $X \mapsto \langle X, Y\rangle_v$ on $\mathcal{C}_N(v,\rho)$ (left).

3.1 Barycenters of Laguerre tessellations as exposed points

We start by observing that $\mathcal{C}_N(v,\rho)$ is a non-empty compact convex subset of $(\mathbb{R}^d)^N$. Moreover, by taking $\varphi(x) = \pm x_i$, we obtain that for any $B \in \mathcal{C}_N(\rho, v)$

$$\sum_{i=1}^{N} b_i v_i = \mathrm{bary}(\rho) \coloneqq \int x \mathrm{d}\rho(x)\,. \tag{7}$$

In particular, one can verify that (7) defines the supporting hyperplane of the set $\mathcal{C}_N$, i.e., the smallest hyperplane containing $\mathcal{C}_N$.

Let us introduce the weighted inner product $\langle X, Y\rangle_v \coloneqq \sum_{i=1}^N \langle x_i, y_i\rangle v_i$, for any $X, Y \in (\mathbb{R}^d)^N$, and denote by $\|\cdot\|_v$ the associated norm. By Strassen's Theorem [15], $B \in \mathcal{C}_N(\rho, v)$ if and only if there exists a collection of probability measures $\theta_i \in \mathcal{P}_1(\mathbb{R}^d)$ such that

$$b_i = \int x \mathrm{d}\theta_i(x) \quad \text{and} \quad \sum_{i=1}^{N} v_i \theta_i = \rho\,.$$

Then, examining the expression of F_N in equation (5) we obtain the following:

▶ **Lemma 3.1.** *The support function of $\mathcal{C}_N$ is the function F_N defined in* (2)*, i.e.,*

$$F_N(Y) = \max\{\langle B, Y\rangle_v \;;\; B \in \mathcal{C}_N\}\,. \tag{8}$$

If $Y \in (\mathbb{R}^d)^N \setminus \Delta_N$, i.e. $y_i \neq y_j$ for all $i \neq j$, then problem (8) has a unique maximizer, F_N is differentiable at Y and $\nabla F_N(Y) = B$. In the language of convex geometry, since B is the unique maximizer of a linear function $\langle \cdot, Y\rangle_v$ on $\mathcal{C}_N$, we say that it is an exposed point, and Y is its exposing direction. On the other hand, if $Y \in \Delta_N$, there are at least two indices $i \neq j$ such that $y_i = y_j$. Then, while there is a unique optimal coupling $\gamma \in \Gamma(\rho, \nu_N(v, Y))$, there is no unique way to decompose it in the form

$$\gamma = \sum_{i=1}^{N} \theta_i \otimes \delta_{y_i}\,.$$

This implies that the direction Y exposes a face of $\mathcal{C}_N$ of dimension strictly larger than zero. Hence, we have the following (see [6, Theorem 2.1] for an analogous result with ρ discrete):

▶ **Theorem 3.2** (Exposed points). $\mathcal{B}^N_{\mathrm{Lag}} = \exp(\mathcal{C}_N)$*, the set of exposed points of $\mathcal{C}_N$.*

Figure 2 provides a graphical illustration of this result. Higher-dimensional faces of $\mathcal{C}_N$ are exposed by directions $Y \in \Delta_N$. Such faces can also be characterized in terms of Laguerre

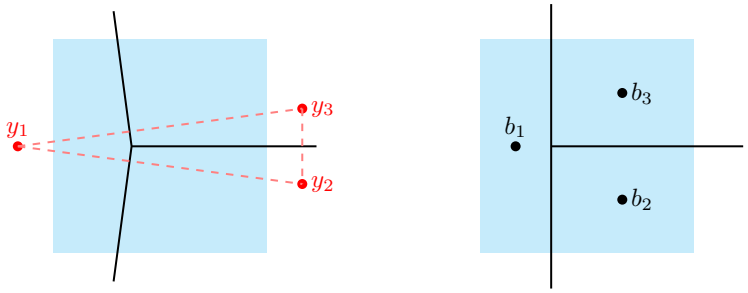


Figure 3 Consider the Laguerre tessellation on the left, defined by the generators (y_1, y_2, y_3) and volumes $v_1 = v_2 = v_3$, where ρ is uniform on a square (shaded area). As y_2 and y_3 approach each other, the tessellation degenerates to the configuration on the right. This limit is not a Laguerre tessellation, but a hierarchical Laguerre tessellation: it can be constructed by first partitioning the square into two cells via a Laguerre tessellation (corresponding to the vertical segment), and then subdividing the right cell using a second Laguerre partition (corresponding to the horizontal segment). In particular, note that the vector of barycenters $B = (b_1, b_2, b_3) \in \mathcal{B}^N_{\mathrm{hLag}}$ is not in $\mathcal{B}^N_{\mathrm{Lag}}$.

tessellations. Suppose $Y \in \Delta_N$ has $M < N$ distinct particles, which we collect in a vector $\tilde{Y} \in (\mathbb{R}^d)^M \setminus \Delta_M$. The unique optimal transport map from ρ to $\sum_{i=1}^N v_i \delta_{y_i}$ is given by a Laguerre tesselation $(L_j)_{j=1}^M$ with generators $\tilde{Y}$. Then the face exposed by Y can be identified with $\bigtimes_{j=1}^M \mathcal{C}_{N_j}((v_i)_{i\in I_j}, \rho|_{L_j})$, where I_j is the set of indices k such that $y_k = \tilde{y}_j$. A precise statement for this result can be found in the full version of this article.

3.2 Hierarchical and irreducible tessellations

When two generators of a Laguerre tessellation become close, the resulting tessellation may converge to one that is not face-to-face, i.e., the intersection of two adjacent cells need not remain a common face of the corresponding polytopes (see Figure 3). From this, one can deduce that $\mathcal{B}^N_{\mathrm{Lag}}$ is not closed. Let us introduce the following classes of tessellations:

▶ **Definition 3.3** (Hierarchical tessellation)**.** A hierarchical Laguerre tessellation is defined recursively as follows: every Laguerre tessellation is a hierarchical Laguerre tessellation; and given a hierarchical Laguerre tessellation $(P_i)_i$, the tessellation obtained by partitioning a given cell P_i with a Laguerre tessellation is also a hierarchical Laguerre tessellation.

▶ **Definition 3.4** (Irreducible tessellation)**.** A Laguerre tessellation is irreducible if it cannot arise from partitioning the cells of a Laguerre tessellation with at least 2 cells (see Figure 4).

We denote the set of cell barycenters of hierarchical and irreducible Laguerre tessellation with N cells and with prescribed volumes v by $\mathcal{B}^N_{\mathrm{hLag}}(\rho, v)$ and $\mathcal{B}^N_{\mathrm{iLag}}(\rho, v)$, respectively.

By considering sequences of generators $(Y^n)_n \subset (\mathbb{R}^d)^N \setminus \Delta_N$ converging to $Y \in \Delta_N$, and using stability of optimal transport plans (see, e.g., Theorem 5.20 in [17]) one can verify that $\overline{\mathcal{B}^N_{\mathrm{Lag}}} = \mathcal{B}^N_{\mathrm{hLag}}$. By Straszewicz's Theorem [16], this implies that $\mathcal{B}^N_{\mathrm{hLag}}$ contains the set of extreme points of $\mathcal{C}_N$, i.e., the points $B \in \mathcal{C}_N$ that cannot be expressed as linear combination of two points in $\mathcal{C}_N \setminus \{B\}$. On the other hand, iterating the argument at the end of Section 2, we still have that for a point $B \in \mathcal{B}^N_{\mathrm{hLag}}$ there exists a unique collection of probability measures $(\theta_i)_i$ as in (5) satisfying (6). Combining this with Strassen's Theorem, we obtain:

▶ **Theorem 3.5** (Extreme points)**.** $\mathcal{B}^N_{\mathrm{hLag}} = \overline{\mathcal{B}^N_{\mathrm{Lag}}} = \mathrm{ext}(\mathcal{C}_N)$*, the set of extreme points of* $\mathcal{C}_N$*.*

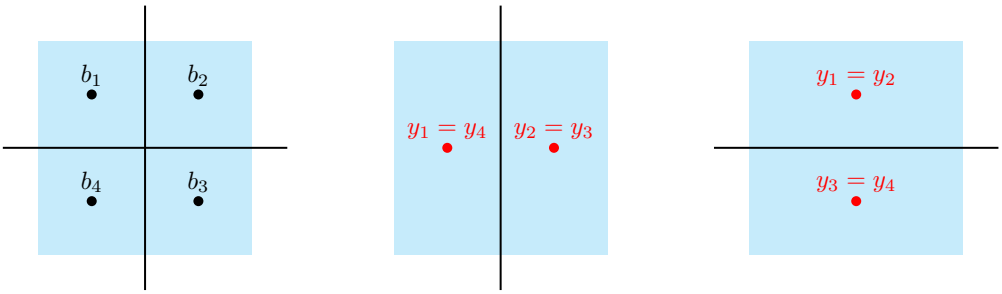


Figure 4 A Laguerre tessellation that is not irreducible (left), arising from partitioning either the center or right tessellation. The vector of barycenters B (w.r.t. ρ uniform on a square, v constant) is exposed by the vectors Y shown in the center and right figure. In particular, it lies on two distinct faces of $\mathcal{C}_N(v,\rho)$ of dimension stritly greater than zero.

Conversely, $\mathcal{B}^N_{\mathrm{Lag}}$ is also not open as a subset of $\operatorname{rel}\partial\mathcal{C}_N$, the relative boundary of $\mathcal{C}_N$. This is because there are tessellations that are not irreducible, i.e., that can be viewed as both Laguerre tessellations and as non-trivial hierarchical partitionings of a Laguerre tessellation (see Figure 4). By the characterization of $\operatorname{rel}\partial\mathcal{C}_N$ described at end of Section 3.1, such points are both exposed and belong to an exposed face of dimension strictly greater than zero and different from $\mathcal{C}_N$. Using this fact we can prove the following:

▶ **Proposition 3.6.** $\mathcal{B}^N_{\mathrm{iLag}}$ is the interior of $\mathcal{B}^N_{\mathrm{Lag}} = \exp(\mathcal{C}_N)$ in $\operatorname{rel}\partial\mathcal{C}_N$.

4 Numerical solution of the reconstruction problem

We propose a numerical method to solve Question 1.1. Given $B \in \mathcal{B}^N_{\mathrm{Lag}}$, we seek to determine a vector of generators of the associated tessellation. This is a vector $Y \notin \Delta_N$ such that $F_N(Y) = \langle Y, B\rangle_v$. While one can solve the problem numerically via convex optimization, this requires explicitly enforcing $Y \notin \Delta_N$, which is costly as it involves $\mathcal{O}(N^2)$ coupled constraints. We propose a method that bypasses this and uses the orthogonal projection:

$$P_{\mathcal{C}_N}(X) \coloneqq \operatorname{argmin}\{\|B - X\|_v^2 \ ; \ B \in \mathcal{C}_N\}. \tag{9}$$

▶ **Proposition 4.1.** Suppose that $B \in \mathcal{B}^N_{\mathrm{iLag}}(v,\rho)$. Then for any $\lambda > 1$ sufficiently small, $P_{\mathcal{C}_N}(\lambda B) \in \mathcal{B}^N_{\mathrm{iLag}}(v,\rho)$ and moreover $Y^* \coloneqq \lambda B - P_{\mathcal{C}_N}(\lambda B) \in (\mathbb{R}^d)^N \setminus \Delta_N$ is a vector of generators associated with $P_{\mathcal{C}_N}(\lambda B)$.

The idea of Proposition 4.1 is illustrated in Figure 5. First, we perturb B by a dilation, i.e. we consider the point λB, with $\lambda > 1$. One can check that $\lambda B \notin \mathcal{C}_N$ and that $B^* = P_{\mathcal{C}_N}(\lambda B)$ is on the relative boundary of $\mathcal{C}_N$. By standard properties of the orthogonal projection, B^* is exposed by the vector $Y^* = \lambda B - B^*$. Finally, by Proposition 3.6, for λ sufficiently small, $B^* \in \mathcal{B}^N_{\mathrm{iLag}}$ if $B \in \mathcal{B}^N_{\mathrm{iLag}}$. From this one can deduce that $Y^* \notin \Delta_N$ and so it is a vector of generators of the unique Laguerre tessellation with barycenters B^*.

▶ Remark. The main drawback of Proposition 4.1 is that it does not provide a quantitative estimate on how small λ should be in order to obtain a set of generators $Y^* \notin \Delta_N$. However, numerically, the approach seems robust with respect to the choice of λ, in the sense that it generally provides a set of distinct generators even for $\lambda \gg 1$. Note also that as $\lambda \to 1$, $B^* \to B$ and by stability of optimal transport maps (see, e.g., Theorem 5.20 in [17]) one can

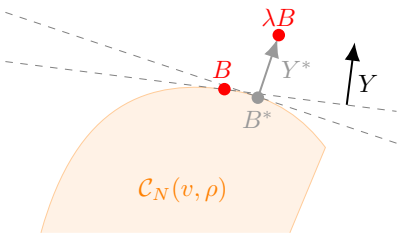


Figure 5 In order to approximate Y, the direction exposing B, we compute $Y^* = \lambda B - B^*$ where $B^* = P_{\mathcal{C}_N(v,\rho)}(\lambda B)$ for $\lambda > 1$ close to 1. By construction, B^* maximizes $\langle \cdot, Y^* \rangle_v$ on $\mathcal{C}_N$, i.e., Y^* exposes B^*, and $B^* \to B$ as $\lambda \to 1$.

deduce that the Laguerre tessellation associated with B^* converges to the one associated with B (i.e., corresponding cells converge in measure). However it is not straightforward to quantify this convergence in terms of $\|B - B^*\|_v$.

Numerical results The projection problem (9) can be solved numerically using the Frank-Wolfe algorithm [8]. This requires maximizing a linear function on $\mathcal{C}_N$ at each iteration, which amounts to solve an optimal transport problem in view of Lemma 3.1. This can be done efficiently using a Newton-based scheme [9], which has been shown to scale to over 10^8 points in 3d [10]; our implementation uses the library `sd-ot` (`https://github.com/sd-ot`). More combinatorial solvers with polynomial-time approximation guarantees also exist [1]. Within the hypotheses of Proposition 4.1, since $\|P_{\mathcal{C}_N}((1+\varepsilon)B) - B\|_v \leq \varepsilon \|B\|_v$, the Frank-Wolfe algorithm computes the generators of a Laguerre tessellation with barycenters B^* satisfying $\|B^* - B\|_v \lesssim \varepsilon$ in $\mathcal{O}(1/\varepsilon^2)$ iterations, although we observe faster convergence in practice.

Figure 6 shows an example of reconstruction obtained using this approach. Note that in this case $B \notin \mathcal{B}^N_{\text{iLag}}$ and the associated Laguerre tessellation can be obtained using different sets of generators, that are not equivalent up to translations and dilations. Due to this, the reconstructed generators differ substantially from those used to produce the data, even if the barycenters are well approximated by the reconstruction.

Note that the same algorithm can be used to fit data that does not necessarily come from Laguerre tessellations, which is the typical scenario in the material science applications motivating this work. Although in this case we cannot guarantee that $P_{\mathcal{C}_N}(\lambda B) \in \mathcal{B}^N_{\text{Lag}}$, this seems to still be the case for typical data (see full version of the article for an example).

References

1 Pankaj K Agarwal, Sharath Raghvendra, Pouyan Shirzadian, and Keegan Yao. Fast and accurate approximations of the optimal transport in semi-discrete and discrete settings. In *Proceedings of the 2024 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 4514–4529. SIAM, 2024.

2 Franz Aurenhammer. A criterion for the affine equivalence of cell complexes in $\mathbb{R}^d$ and convex polyhedra in $\mathbb{R}^{d+1}$. *Discrete & Computational Geometry*, 2(1):49–64, 1987.

3 Franz Aurenhammer. Power diagrams: properties, algorithms and applications. *SIAM Journal on Computing*, 16(1):78–96, 1987.

4 Franz Aurenhammer, Friedrich Hoffmann, and Boris Aronov. Minkowski-type theorems and least-squares clustering. *Algorithmica*, 20(1):61–76, 1998.

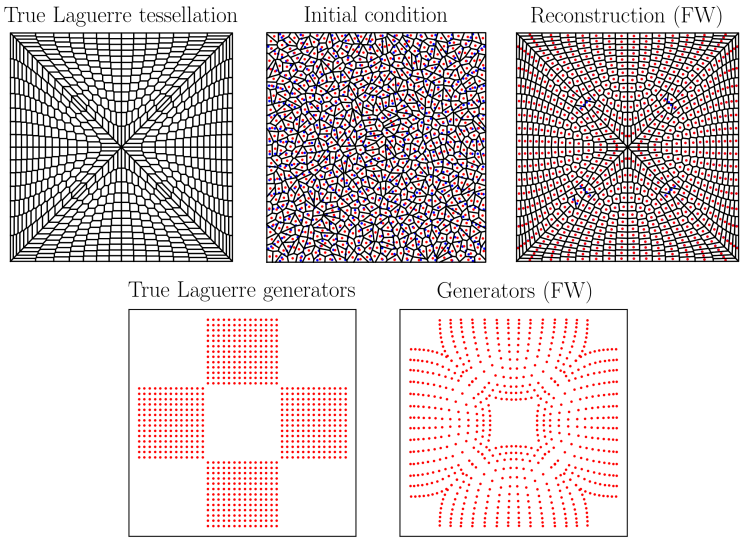


Figure 6 An example of reconstruction obtained using the Frank-Wolfe algorithm with $\lambda = 1.01$. The barycenters of the true solution are in blue and those of the reconstruction are in red.

5 David P. Bourne, Mason Pearce, and Steven M. Roper. Inverting Laguerre tessellations: Recovering tessellations from the volumes and centroids of their cells using optimal transport. *ESAIM: Mathematical Modelling and Numerical Analysis*, 59(5):841–871, 2025.

6 Andreas Brieden and Peter Gritzmann. On optimal weighted balanced clusterings: Gravity bodies and power diagrams. *SIAM Journal on Discrete Mathematics*, 26(2):415–434, 2012.

7 Andreas Brieden, Peter Gritzmann, and Fabian Klemm. Constrained clustering via diagrams: A unified theory and its application to electoral district design. *European Journal of Operational Research*, 263(1):18–34, 2017.

8 Marguerite Frank, Philip Wolfe, et al. An algorithm for quadratic programming. *Naval research logistics quarterly*, 3(1-2):95–110, 1956.

9 Jun Kitagawa, Quentin Mérigot, and Boris Thibert. Convergence of a Newton algorithm for semi-discrete optimal transport. *Journal of the European Mathematical Society*, 21(9):2603–2651, 2019.

10 Bruno Lévy, Nicolas Ray, Quentin Mérigot, and Hugo Leclerc. Large-scale semi-discrete optimal transport with distributed Voronoi diagrams. *Journal of Computational Physics*, page 114374, 2025.

11 Allan Lyckegaard, Erik Mejdal Lauridsen, Wolfgang Ludwig, Richard Warren Fonda, and Henning Friis Poulsen. On the use of Laguerre tessellations for representations of 3D grain structures. *Advanced Engineering Materials*, 13(3):165–170, 2011.

12 Quentin Mérigot and Boris Thibert. Optimal transport: discretization and algorithms. In A. Bonito and R. H. Nochetto, editors, *Geometric Partial Differential Equations - Part II*, volume 22 of *Handbook of Numerical Analysis*, pages 133–212. North-Holland, 2021.

13 Lukas Petrich, Jakub Staněk, Mingyan Wang, Daniel Westhoff, Luděk Heller, Petr Šittner, Carl E Krill III, Viktor Beneš, and Volker Schmidt. Reconstruction of grains in polycrystalline materials from incomplete data using Laguerre tessellations. *Microscopy and Microanalysis*, 25(3):743–752, 2019.

14 Filippo Santambrogio. *Optimal Transport for Applied Mathematicians*. Birkhäuser, 2015.

15 Volker Strassen. The existence of probability measures with given marginals. *The Annals of Mathematical Statistics*, 36(2):423–439, 1965.

16 Stefan Straszewicz. Über exponierte punkte abgeschlossener punktmengen. *Fundamenta Mathematicae*, 24(1):139–143, 1935.

17 Cédric Villani. *Optimal transport: old and new*, volume 338. Springer, 2009.

Simultaneous Embedding of Two Paths on the Grid

Stephen Kobourov[1], William Lenhart[2], Giuseppe Liotta[3], Daniel Perz[3], Pavel Valtr[4], and Johannes Zink[1]

1 **Technical University of Munich, Heilbronn, Germany**
{stephen.kobourov, johannes.zink}@tum.de
2 **Williams College, Williamstown, MA, USA**
wlenhart@williams.edu
3 **University of Perugia, Perugia, Italy**
{giuseppe.liotta, daniel.perz}@unipg.it
4 **Charles University, Prague, Czech Republic**
valtr@kam.mff.cuni.cz

Abstract

We study the problem of simultaneous geometric embedding of two paths without self-intersections on an integer grid. We show that minimizing the length of the longest edge of such an embedding is NP-hard. We also show that we can minimize in $O(n^{3/2})$ time the perimeter of an integer grid containing such an embedding if one path is x-monotone and the other is y-monotone.

Related Version arXiv:2603.09750

1 Introduction

A simultaneous geometric embedding of a pair of planar graphs sharing the same vertex set consists of two straight-line planar drawings of the graphs in which every vertex has the same location in both drawings and distinct edges intersect in at most one point. If an edge is shared by the two graphs, then it is represented by the same straight-line segment which makes it easy to visually discover common patterns in the drawings. The notion of simultaneous geometric embedding was introduced by Brass et al. [5] almost twenty years ago. Since then, it has received considerable attention in the graph drawing literature, partly because of the theoretical challenges it poses and partly because it lies at the core of dynamic graph visualization. For example, the simultaneous embedding of cycles can be used to show max-degree-4 graphs have geometric thickness two [8]. The survey by Bläsius et al. [3] provides extensive discussions about simultaneous geometric embeddings and their variants.

Unsurprisingly, only very restricted pairs of graphs admit a simultaneous geometric embedding. For example, Angelini et al. [1] show that there exists a path and tree of depth four which do not have a simultaneous geometric embedding, while Cabello et al. [6] prove that every tree and a matching admit a simultaneous geometric embedding, but there exist a planar graph and a matching which do not. On the other hand, Brass et al. [5] present an elegant algorithm to compute a geometric simultaneous embedding of two paths sharing the same n-vertex set, based on the two linear orders of the paths which determine the x- and y-coordinates of the vertices on an $n \times n$ integer grid. For the general case, Estrella-Balderrama et al. [10] showed that deciding whether two planar graphs admit a simultaneous geometric embedding is NP-complete. Moreover, the problem for a linear number of planar graphs is hard in the existential theory of the reals [7].

Motivated by the observation that the algorithm by Brass et al. [5] for two paths does not minimize either the maximum edge length or the total edge lengths, we study these two optimization problems. Specifically, we show that the problem of finding a geometric

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

simultaneous embedding of two paths on a grid is NP-complete when the objective is to minimize the maximum edge length, or to minimize the sum of edge lengths.

We then turn to the monotone variant of the problem, for which we present an $O(n^{3/2})$-time algorithm to minimize the perimeter of the bounding box while guaranteeing that one of the paths is x-monotone and the other is y-monotone. Our approach is based on formulating an appropriate system of constraints that can be interpreted as a graph problem.

2 NP-Completeness

For the following theorem, we provide the full proof in the full version [12].

▶ **Theorem 2.1.** *Simultaneous embedding of two paths on a grid is NP-complete for the objectives of minimizing the maximum edge length and minimizing the sum of edge lengths.*

Proof Sketch. Containment in NP is clear. We show NP-hardness by reduction from NotAllEqual3SAT. Our reduction resembles the *logic engine* [2, 9]. For an illustration of our construction see Figures 1 and 2.

Let the input NotAllEqual3SAT instance have n variables and m clauses. On a high level, we have a rigid frame on the left and on the right. In the middle there is a *shaft* connecting the two parts of the frame. The shaft consists of n rigid parts connected by single edges. Each such rigid part belongs to a *striker*; in Figure 1, there are 4 vertically extending strikers. Since each striker is attached to the rest by two single edges, there is one degree of freedom to flip the drawing of each striker individually (in Figure 1, each striker might be mirrored along a horizontal axis through the shaft). Each striker corresponds to a variable and its one drawing represents the assignment true, while its flipped drawing represents the assignment false. For every clause, there are two horizontal strips (called *clause strips*; colored in Figure 1, one above and one below the shaft). Each striker has $4m$ *flags* – one per strip and a "dummy" flag between two clause strips. Each flag is (in its striker) vertically connected to the neighboring flag or the shaft by a single edge and, hence, can be flipped to the left or right individually. A flag is internally rigid and extends to the left and the right side. It can have two short sides (*two-short flag*) or a short and a long side (*long flag*); see also Figure 1 where "flag" is written. All two-short flags are isomorphic and also all long flags are isomorphic. All flags in the clause strips are long flags unless the variable appears in the corresponding clause.

If the variable appears normally in the clause, then a two-short flag is in the clause strip above the shaft when the orientation of the striker corresponds to true. If the variable appears negated in the clause, then a two-short flag is in the clause strip above the shaft when the orientation of the striker corresponds to false. All "dummy" flags are two-short flags and their orientation is not further relevant. For a drawing without crossings, the long side of a striker must be drawn in one of $n-1$ gaps between two strikers (next to the frame, there is no gap that can fit the long side of a flag). Two long sides of flags of neighboring striker cannot be drawn in the same gap. This means, in both clause strips of each clause, there is at least one two-short flag. This corresponds to a truth assignment where at least one literal per clause is false and at least one literal per clause is true. We can use this construction to show that there is a crossing-free drawing with unit length if and only if the input NotAllEqual3SAT instance is a yes-instance. ◀

$(\neg x_1 \vee \neg x_2 \vee \neg x_3) \wedge (\neg x_1 \vee x_2 \vee \neg x_4) \wedge (x_1 \vee \neg x_3 \vee x_4)$

$x_1 = \text{false}, \quad x_2 = \text{true}, \quad x_3 = \text{false}, \quad x_4 = \text{true}$

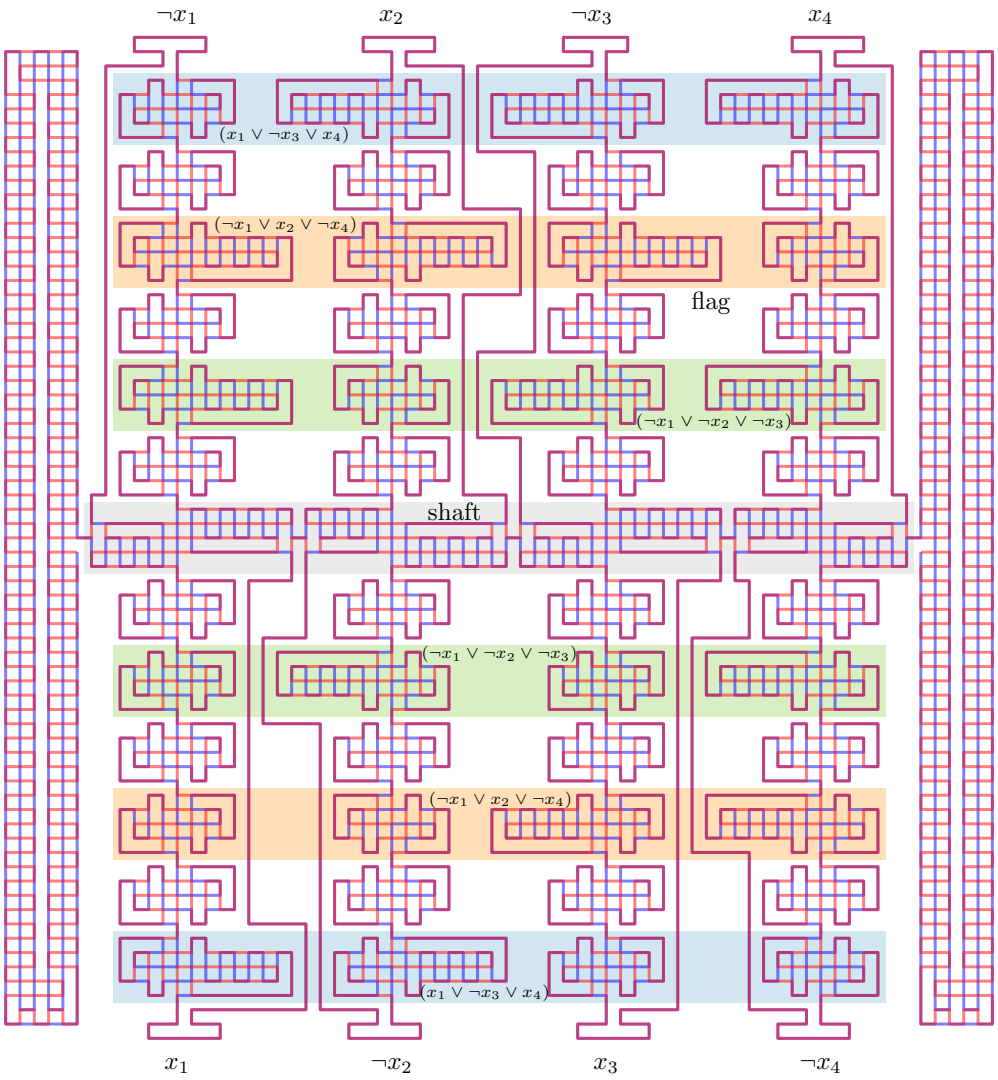


Figure 1 Simultaneous embedding of two paths (all edges have length 1) obtained in our NP-hardness reduction from the NotAllEqual3SAT instance given on top. Edges of P_1 are colored blue, edges of P_2 are colored light red, and shared edges are colored purple.

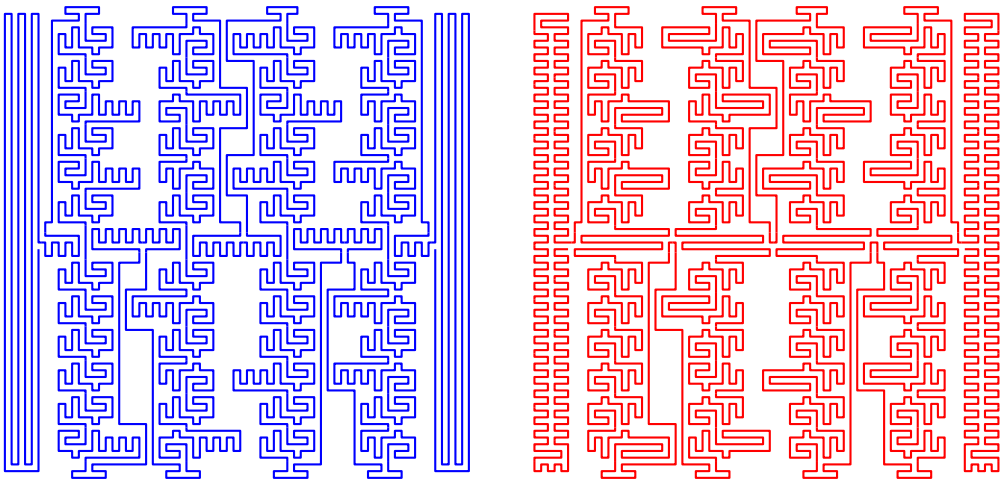


Figure 2 The paths P_1 and P_2 of Figure 1.

3 Minimizing the parameter for two monotone paths

We consider here the problem of simultaneously embedding two directed paths on the grid such that one of the paths is weakly monotonic in the x-direction and the other is weakly monotonic in the y-direction. The *path pair graph* of two directed paths P_x and P_y is given by $G = G(P_x, P_y) = (V, E)$, where $V = \{v_1, \ldots, v_n\}$ is the common vertex set of P_x and P_y and E is the union of the sets of directed edges of P_x and P_y. That is, the simultaneous embedding of P_x and P_y is just an embedding of the path pair graph $G = G(P_x, P_y)$ such that P_x is weakly x-monotonic and P_y is weakly y-monotonic. We call such an embedding a *weakly monotonic grid embedding of G (a WMGE of G).*

▶ **Theorem 3.1.** *Let $G = G(P_x, P_y)$ be be a path pair graph. Then we can compute a WMGE of G with minimum perimeter in polynomial time.*

Proof. Note that P_x and P_y induce permutations π_x and π_y on V in which $\pi_x(i)$ is the i^{th} vertex of P_x and $\pi_y(i)$ is the i^{th} vertex of P_y. Thus $E(G) = E = \{(v_{\pi_x(i)}, v_{\pi_x(i+1)}) : 1 \leq i < n\} \cup \{(v_{\pi_y(i)}, v_{\pi_y(i+1)}) : 1 \leq i < n\}$. A WMGE of G consists of assigning to each v_k a distinct location $S_k = (x_k, y_k)$ on the $2D$ integer grid such that $x_{\pi_x(i)} \leq x_{\pi_x(i+1)}$ and $y_{\pi_y(i)} \leq y_{\pi_y(i+1)}$ for all $1 \leq i < n$, and any two edges of the embedding intersect in at most 1 point (no "segment overlap").

We denote the predecessor and successor of v on P_x by $p_x(v)$ and $s_x(v)$, respectively; $p_y(v)$ and $s_y(v)$ are defined analogously for path P_y. An internal vertex v of P_x is a *switch vertex of P_x* if either (i) both $p_x(v)$ and $s_x(v)$ come before v on P_y or (ii) both come after v on P_y. The pair of edges $(p_x(v), v)$ and $(v, s_x(v))$ are called a *switch pair of P_x*. A switch vertex and switch pair of P_y are defined analogously. A pair of vertices $\{u, v\}$ forms a *shared edge* if they are consecutive on both paths (although their relative order on each path may be different). Any WMGE of G must satisfy:

- For each switch vertex v of P_x, the x-coordinates of $p_x(v)$ and $s_x(v)$ differ by at least 1 (no vertical segment overlap),
- for each switch vertex v of P_y, the y-coordinates of $p_y(v)$ and $s_y(v)$ differ by at least 1 (no horizontal segment overlap), and

- for any shared edge $\{v_i, v_j\}$ (for some $i, j \in \{1, \ldots, n\}$), the sum of the x- and y-extents of the segment S_iS_j is at least 1 (no two points in the same location).

Note that in any WMGE of G, each embedded edge of G will have non-negative integer x- and y-extents; we denote the x- and y-extents of an embedded edge e by $d_x(e)$ and $d_y(e)$, respectively. The three properties above can be reformulated as x- and y-extent constraints on the embedded edges of P_x and P_y:

- For any switch vertex v on P_x, we have $d_x(p_x(v), v) + d_x(v, s_x(v)) \geq 1$,
- for any switch vertex v on P_y, we have $d_y(p_y(v), v) + d_y(v, s_y(v)) \geq 1$, and
- for any shared edge $\{u, v\}$, we have $d_x(u, v) + d_y(u, v) \geq 1$.

In fact, if $\{d_x(e) : e \in P_x\}$ and $\{d_y(e) : e \in P_y\}$ are any non-negative integers satisfying these three constraints, then $d_x()$ and $d_y()$, along with integer values for the x-coordinate of $v_{\pi_x(1)}$ and the y-coordinate of $v_{\pi_y(1)}$, completely determine a WMGE of G:

1. Let $x(v_{\pi_x(1)}) = 0$;
2. let $x(v_{\pi_x(i+1)}) = d_x(v_{\pi_x(i)}, v_{\pi_x(i+1)}) + x(v_{\pi_x(i)})$, for each $i : 1 \leq i < n$;
3. let $y(v_{\pi_y(1)}) = 0$; and
4. let $y(v_{\pi_y(i+1)}) = d_y(v_{\pi_y(i)}, v_{\pi_y(i+1)}) + y(v_{\pi_x(i)})$, for each $i : 1 \leq i < n$.

It is not difficult to show that every vertex gets integer x and y coordinates, P_x is weakly x-monotonic and P_y is weakly y-monotonic, any two edges intersect in at most one point, no vertex lies in the interior of an edge, and no two distinct vertices are assigned to the same point.

We justify the fifth claim: Let two vertices u and v be assigned to the same locations. Note that $\{u, v\}$ cannot form a shared edge, since $d_x(u, v) + d_y(u, v) \geq 1$ holds for shared edges. Let x_0 be the common x-coordinate of u and v; there must be two consecutive vertices w and w' of P_x that have x-coordinate x_0. Then $\{w, w'\}$ is not an edge of P_y, so there is a unique subpath of P_y between w and w'. On P_x, one of the vertices of this subpath must come after all of the other vertices of the subpath. But then this vertex is a switch vertex of P_y and so its predecessor and successor on P_y must have vertical distance at least 1. Therefore u and v also have vertical distance at least one, and are not in the same location.

Note that we do not need to specify, for non-shared edges, y-extents of edges of P_x or x-extents of edges of P_y; these will be implicitly determined by the fact that for any edge $\{u, v\}$ on one of the paths, there is a unique subpath of the other path between u and v.

Note that the perimeter of the embedding is just twice the sum of the x-extents of the P_x edges and the y-extents of the y-path edges, and so does not depend on the assignment of an x-coordinate to $v_{\pi_x(1)}$ and a y-coordinate to $v_{\pi_y(1)}$.

Finally, we note that in a minimum perimeter WMGE of G, each d_x and d_y value is either 0 or 1, since replacing an integer value greater than 1 with 1 does not violate any of the extent constraints. Thus the problem of finding a minimum perimeter grid embedding of G becomes the problem of finding 0/1 values $d_x(e)$ for each edge e of P_x and $d_y(e)$ for each edge e of P_y that satisfy the three extent constraints and that minimize $\sum_{e \in P_x} d_x(e) + \sum_{e \in P_y} d_y(e)$.

To compute a WMGE of a path pair graph $G = (V, E)$, we construct a new graph $C_G = (V_C, E_C)$ from G, called the *constraint graph of* G, see Figure 3. Let $X = \{e : e \in P_x\}$ and $Y = \{e : e \in P_y\}$. The vertices and edges of C_G are defined as follows:

- $V_C = V_X \cup V_Y$, where V_X has a distinct vertex for each $e \in X$ and V_Y has a distinct vertex for each $e \in Y$. Shared edges of G therefore appear twice in V_C.
- $E_C = E_X \cup E_Y \cup M$ where

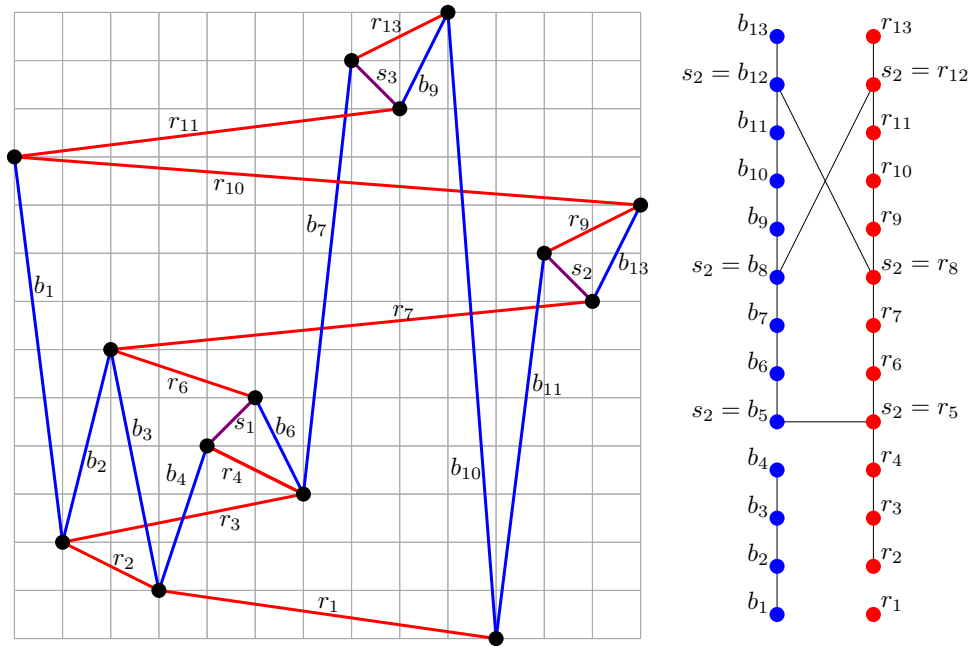


Figure 3 A WMGE of a path pair graph G and the constraint graph C_G. The paths of G and their vertices in C_G are red and blue, shared edges are purple.

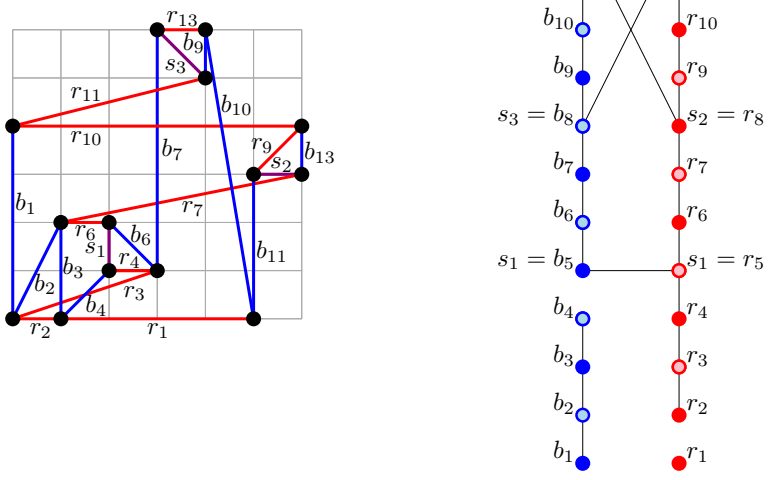


Figure 4 A min-perimeter drawing of G of Figure 3 and the corresponding minimum vertex cover of C_G, indicated by circles. Every blue or red vertex in the vertex cover corresponds to an edge whose endpoints differ by one in the x-direction or y-direction, respectively.

- E_X is the set of all switch pairs for P_x,
- E_Y is the set of all switch pairs for P_y, and
- M is a matching connecting each shared edge in V_X to its counterpart in V_Y.

Note that the induced subgraphs of C_G obtained by restricting the vertex set to V_X and V_Y, respectively, are each path forests. It is easy to see that a WMGE of G corresponds to a vertex cover of C_G since any satisfying assignment of 0/1 values to $d_x()$ and $d_y()$ corresponds to an assignment of 0/1 values to the vertices of C_G such that every edge of C_G has at least one vertex with value 1. The converse also holds. Further, the size of the cover gives (half) the perimeter of the embedding, so we have transformed the embedding problem into the problem of finding a minimum vertex cover of C_G, see Figure 4.

▶ Claim 1. For any path pair graph G, C_G is a bipartite graph.

Proof. We show that any cycle in C_G must have even length. Note that for any edge $e = \{u, v\}$ in G, u and v have two induced orderings determined by π_x and π_y. We say e has *positive alignment* if those orders are the same and *negative alignment* otherwise. Note that any two edges that form a switch pair for one of the two paths will have opposite alignments. Note also that this allows us to assign an alignment value to each vertex of C_G and that the end vertices of an edge of C_G will have equal alignments exactly when those vertices represent the same shared edge of G.

Consider a cycle $C = v_1, \ldots, v_k$ in C_G. Each consecutive pair of vertices corresponds to either a switch pair or a shared edge of G. C must cross between V_X and V_Y an even number of times, since V_X and V_Y induce path forests in C_G. Thus there is an even number of edges of C whose end vertices have the same alignment. Hence, C is an even length cycle. ◀

Since C_G is bipartite, a minimum vertex cover of C_G can be found in $O(n^{3/2})$ time by first using the Hopcroft-Karp [11] algorithm to compute a maximum matching M for C_G in time $O(|E_C|\sqrt{|V_C|}) = O(n^{3/2})$ then using the method of Kőnig [13] to traverse C_G in linear time building M-alternating paths in order compute a minimum vertex cover [4, Theorem 5.3]. ◀

Interestingly, while this problem can also be solved by using linear programming and modifying the solution until integer values are obtained, our proof that this method works relies on Claim 1 and our algorithm is much more efficient.

Acknowledgments. This research was initiated at the Eighteenth Bertinoro Workshop on Graph Drawing 2025. The research by Pavel Valtr was supported by the Czech Science Foundation under the grant agreement no. 23-04949X.

References

1 Patrizio Angelini, Markus Geyer, Michael Kaufmann, and Daniel Neuwirth. On a tree and a path with no geometric simultaneous embedding. *J. Graph Algorithms Appl.*, 16(1):37–83, 2012. doi:10.7155/JGAA.00250.

2 Sandeep N. Bhatt and Stavros S. Cosmadakis. The complexity of minimizing wire lengths in VLSI layouts. *Inf. Process. Lett.*, 25(4):263–267, 1987. doi:10.1016/0020-0190(87)90173-6.

3 Thomas Bläsius, Stephen G. Kobourov, and Ignaz Rutter. Simultaneous embedding of planar graphs. In Roberto Tamassia, editor, *Handbook on Graph Drawing and Visualization*, pages 349–381. Chapman and Hall/CRC, 2013. URL: https://cs.brown.edu/people/rtamassi/gdhandbook/chapters/simultaneous.pdf.

4 J. Adrian Bondy and Uppaluri S. R. Murty. *Graph Theory with Applications*. Macmillan Education UK, 1976. doi:10.1007/978-1-349-03521-2.

5 Peter Brass, Eowyn Cenek, Christian A. Duncan, Alon Efrat, Cesim Erten, Dan Ismailescu, Stephen G. Kobourov, Anna Lubiw, and Joseph S. B. Mitchell. On simultaneous planar graph embeddings. *Comput. Geom.*, 36(2):117–130, 2007. doi:10.1016/J.COMGEO.2006.05.006.

6 Sergio Cabello, Marc J. van Kreveld, Giuseppe Liotta, Henk Meijer, Bettina Speckmann, and Kevin Verbeek. Geometric simultaneous embeddings of a graph and a matching. *J. Graph Algorithms Appl.*, 15(1):79–96, 2011. doi:10.7155/JGAA.00218.

7 Jean Cardinal and Vincent Kusters. The complexity of simultaneous geometric graph embedding. *J. Graph Algorithms Appl.*, 19(1):259–272, 2015. doi:10.7155/JGAA.00356.

8 Christian A Duncan, David Eppstein, and Stephen Kobourov. The geometric thickness of low degree graphs. In *Proceedings of the 20th Annual Symposium on Computational geometry (SoCG 2004)*, pages 340–346, 2004. doi:10.1145/997817.997868.

9 Peter Eades and Sue Whitesides. The logic engine and the realization problem for nearest neighbor graphs. *Theor. Comput. Sci.*, 169(1):23–37, 1996. doi:10.1016/S0304-3975(97)84223-5.

10 A. Estrella-Balderrama, E. Gassner, M. Jünger, M. Percan, M. Schaefer, and M. Schulz. Simultaneous geometric graph embeddings. In S.-H. Hong, T. Nishizeki, and W. Quan, editors, *Proceedings of the 15th International Symposium on Graph Drawing (GD 2007)*, volume 4875 of *Lecture Notes in Computer Science*, pages 280–290. Springer, 2008. doi:10.1007/978-3-540-77537-9_28.

11 John E. Hopcroft and Richard M. Karp. A $n^{5/2}$ algorithm for maximum matchings in bipartite graphs. *SIAM J. Comput.*, 2(4):225–231, 1973.

12 Stephen Kobourov, William Lenhart, Giuseppe Liotta, Daniel Perz, Pavel Valtr, and Johannes Zink. Simultaneous embedding of two paths on the grid. arXiv preprint, 2026. arXiv:2603.09750.

13 Dénes Kőnig. Gráfok és mátrixok. *Matematikai és Fizikai Lapok*, 38:116–119, 1931.

Online searching for rays in the half-plane*

Florian Gans and Elmar Langetepe

University of Bonn, Institute of Computer Science V, Algorithms and Complexity
s6flgans@uni-bonn.de elmar.langetepe@informatik.uni-bonn.de

Abstract

We consider the problem of searching for special unknown rays in the half-plane and apply a generalized online cow-path like strategy with exponentially increasing X-distances with respect to a fixed basis r and we move up with a fixed slope α. We found values that guarantee a worst case ratio of at most 9.12725 against the shortest offline path for detecting the ray. By special adjustments we attain the same ratio for a 1.5D terrain search problem which motivates our problem definition. We can also slightly improve the lower bound to 9.06357 shown in the full technical report [17].

Related Version arXiv:2512.16622 [17]

1 Introduction

The so-called cow-path problem is a well-known online search problem for an unknown target point on a line. The searcher starts at some fixed starting point s on the infinite line l and can move along the line until the target t is visited. The distance and location of the target is unknown, the agent has to visit both sides in an alternating way, see Figure 1. Thus any reasonable strategy will be described by an (infinite) sequence of moves $X = (x_1, x_2, x_3, x_4, x_5, \ldots)$, $x_i \in \mathbb{R}$ for $i \in \mathbb{N}$, such that the agent visits one direction up to increasing depth $x_1 < x_3 < x_5 < \ldots$ and in the other direction up to increasing depth $x_2 < x_4 < x_6 < \ldots$ and visits the two directions alternately. For a given strategy X a (local) worst case situation in relative comparison to the unknown shortest path to the target appears, when the agent slightly miss the target at step k at distance x_k, starts a new iteration to the other side with distance x_{k+1} and after turning back, the target is detected closely behind the former visit at some distance $|s\,t| = x_k + \epsilon$ for an arbitrarily small ϵ. We are looking for an efficient *online* strategy under incomplete information against an optimal *offline* solution under full information, in this case the direct shortest path to the goal. One is looking for a sequence X that guarantees that the length of the online search path to a goal is only C times worse than the shortest path to the goal. Such a framework introduced by Sleater and Tarjan [19] is known as *competitive analysis* for *online* algorithms or strategies.

* Thanks for fruitful discussion at the Lorentz Center Workshop on *New challenges in search and patrolling games*

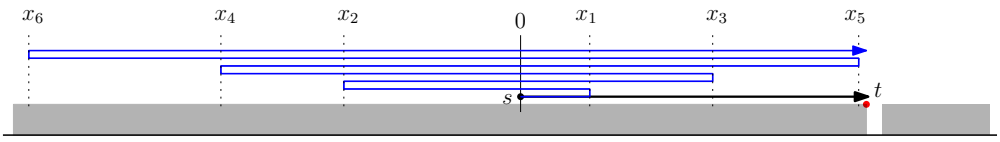


Figure 1 The *cow* is searching for the unknown *hole in the fence* and moves forth and back along the fence. Any reasonable search strategy can be represented by a sequence $X = (x_1, x_2, x_3, x_4, \ldots)$. The cow runs into a local worst case situation in relative comparison to the unknown shortest path to the target, when the target is distance $|s\,t| = x_k + \epsilon$ away from the start (here for $k = 5$).

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

In our case we try to find a sequence such that for a constant $C \geq 1$ we guarantee

$$2\sum_{i=1}^{k+1} x_i + (x_k + \epsilon) \leq C \cdot (x_k + \epsilon) \text{ holds for any } k \in \mathbb{N} \text{ and any } \epsilon > 0\,. \tag{1}$$

We can neglect ϵ and assume that the goal is at least one step away and we are searching for a sequence that guarantees the overall smallest C such that

$$\frac{2\sum_{i=1}^{k+1} x_i + x_k}{x_k} = 1 + 2\frac{\sum_{i=1}^{k+1} x_i}{x_k} \leq C \text{ holds for any } k \in \mathbb{N}\,. \tag{2}$$

Note that it is obvious that the above local worst case guarantee is sufficient to guarantee the approximation ratio C for all other target locations. The problem was first introduced as a linear search problem by Beck [4] in 1964. It is well-known that a doubling heuristic with $x_i = 2^{i-1}$ gives an optimal strategy with $C = 9$. The first proof of optimality was given by Gal [11] in the context of search games. Later the problem was re-invented by Baeza-Yates et. al [3]. The search space is a line (or alternatively s is the start of two infinite rays) so it is a $1D$ search problem. The given search problem was further extended w.r.t. many different aspects (randomization, turn costs, a star of m-rays starting from s, hints etc.) [2, 5, 6, 9, 12, 14, 18]. Extending the search problem to searching for points, lines (or rays) in 2D is an ultimate challenge which is not completely solved ; see also [1, 7, 10, 13, 15, 16].

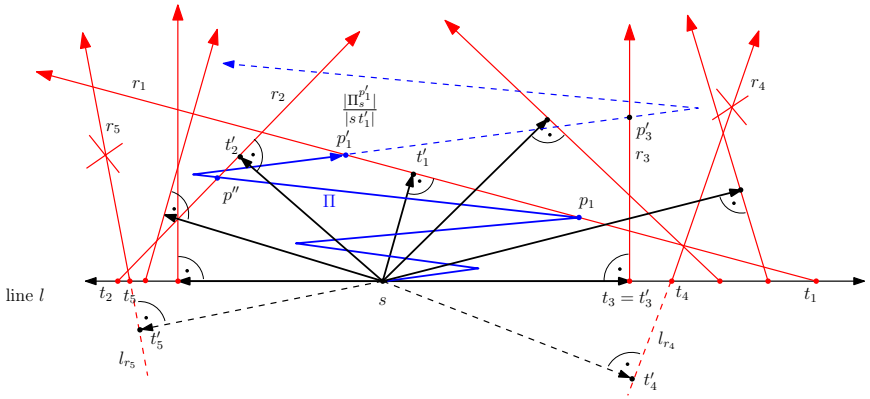


Figure 2 A subset of the rays in a half-plane we are looking for by a cow-path like strategy Π. For a target ray r the shortest path from s to the ray r with source t on l, i.e., a segment st' with $t' \in r$ that runs perpendicular to r, also has to lie in the upper half-plane. For example the rays r_4 and r_5 are not in our focus. A reasonable search path, Π, might be given by a generalized cow-path that also increases its height successively. The path alternate from one side to the other and increases the height with some (fixed) slope.

1.1 Search problem statement and motivation by related work

As an intermediate (1.5D) version we are extend the search problem to searching for (special) rays in a given half-plane indicated by a line l and would still like to apply a cow-path like strategy which means moving from one side to the other but also with a fixed slope for moving up! Generally, an agent moves along a search path Π inside a half-plane that is bounded by a line l and the agent starts at a point s on l. We try to efficiently detect (by

visiting) an arbitrary unknown target ray r that has its source t on l. The rays of interest are restricted to rays that do do not *point away* from the start point s. Assuming w.l.o.g. that l is the X-axis, $s = (0, 0)$ is the origin and searching is done in the upper half-plane, this means that the rays are either vertical or the rays intersect the Y-axis above the X-axis. In our online setting as usual we compete against the length of the shortest path to the ray; see Figure 2. And also local worst case situations occur. Ray r_1 is a local worst case situation for the strategy. The ray is detected at p'_1 and we have *rotated* r_1 around p'_1 such that $s\,t'_1$ gets minimal, in the limit r_1 is just touching the generalized cow-path at p_1 where r_1 was actually slightly missed by the strategy in the iteration before. Thus we have a very natural extension. Note that further restriction to vertical rays directly brings us back to the traditional cow-path problem.

The situation is also motivated by the following considerations in a recent work from S. de Berg et al. (Latin 2024) [8] on an 1.5D terrain T. The upper boundary of T has to be X-monotone or monotone w.r.t. some ground level G that can also be assumed to run in parallel to the X-axis.

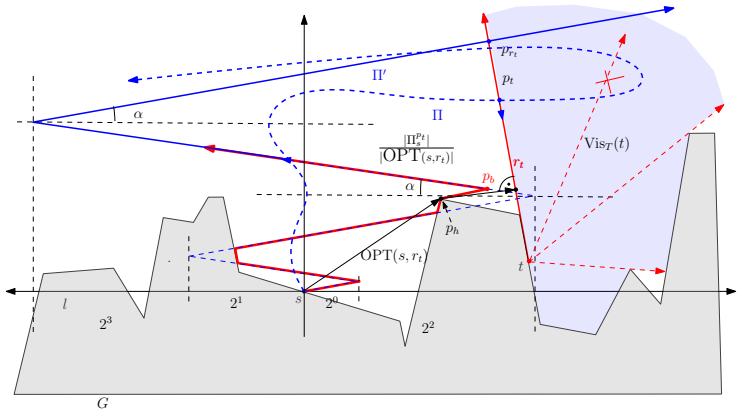


Figure 3 Searching for a target point t on the terrain by a search path Π. The target is detected at p_t when the path enters the visibility region Vis_T of t w.r.t. T at a ray that points toward the target. The authors in [8] suggest to use a fixed zig-zag strategy with $x_i = 2^i$ and a fixed slope given by angle $\alpha \approx 0.231477$ for orientation. The return to the other side will always be performed along the fixed pre-computed path. If the slope direction is free again and the next return segment was not met so far (i.e. at p_h) the agent moves into the slope direction α for a while.

An agent starts somewhere on the terrain and has to keep above the terrain along a search path, Π, in order to find an unknown target t located on the terrain, comparable to a helicopter that is moving above the terrain, see Figure 3. Along the search path Π at some point p_t the target might get into sight for the first time, i.e., the segment $t\,p_t$ does not cross the terrain any more. In this case the target t will be detected. Formally, at p_t the searcher enters the visibility region of the target t, $\mathrm{Vis}_T(t)$, for the first time. By the X-monotonicity of the terrain boundary, this will indeed always occur for a ray r_t that *points* toward the target. Now the searcher competes against the shortest path from the start s that has to keep above the terrain and also visits this ray, i.e., the offline shortest path for detecting the goal, $\mathrm{OPT}_T(s, r_t)$.

S. de Berg et al. (Latin 2024) [8] presented a strategy for the terrain search problem. As a basis they make use of an alternating doubling strategy with $x_i = 2^i$ in the X-direction and a fixed slope given by $\alpha \approx 0.231477363970178$ for any Y-movement, see Figure 3. This indeed gives a generalized cow-path.

If the terrain will be hit by this path in a given direction, the agent has to react, she starts moving along the terrain but will always pick up the fixed pre-computed next return path again, see Figure 3.

Analytically, a competitive ratio of $3\sqrt{19/2} \approx 9.24663$ have been shown in general. The proof directly takes the terrain into account. The authors also show that no strategy can achieve a ratio less than $\sqrt{82} \approx 9.05538$.

2 Better solutions for the pure half-plane search

Different from the above known result we first just concentrate on a better solution for our ray search problem without any terrain. After that we suggest a different adjustment of the path for the terrain in the next section. This finally shows that the terrain actually can only be helpful, if we do the adjustment appropriately, we attain the same better ratio. We also find a better lower bound and further close the gap. All technical details can be found in the full version of this paper; see [17].

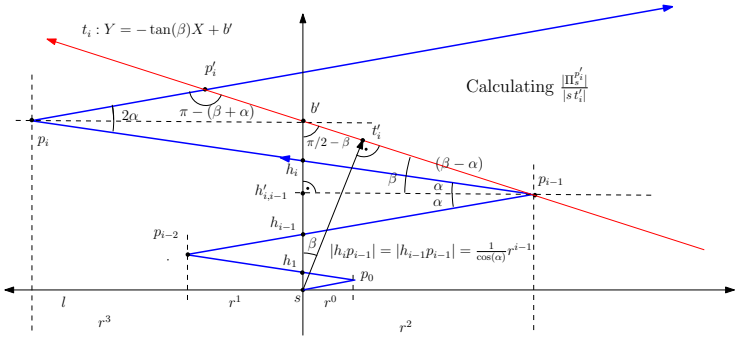


Figure 4 The math for the general local worst case subsumes all situations. Any worst case ray with a slope β will be considered if we consider all $\beta \in [\alpha, \pi/2]$. Such a given ray is part of the line $t_i : Y = -\tan(\beta)X + b'$. Inserting the known point p_{i-1} gives b'. The triangles $s\,t'_i\,b'$ and $p_{i-1}\,h'_{i,i-1}\,b'$ are congruent. Thus we compute st'_i by the law of sine. Similarily the length $|p_i p'_i|$ is computed by the law of sine in the triangle $p_{i-1}\,p'_i\,p_i$.

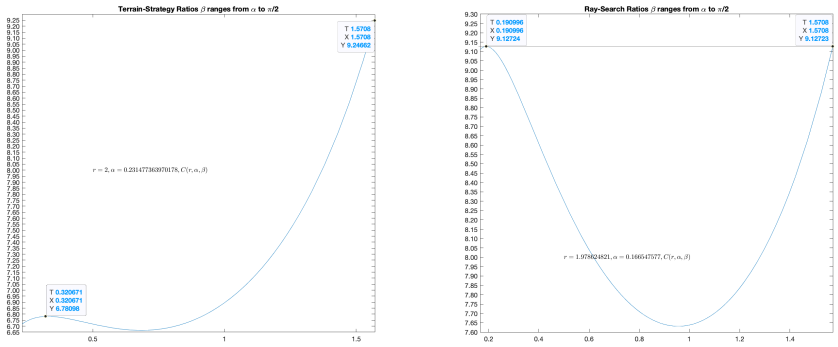
The first point is that we design a cow-path like strategy with X-distances $x_i = r^i$ and a slope α and we leave both parameters open for a while. For this we can analytically do the math for the worst case situation of any ray. By fixing r and α the strategy is fully given and the worst cases appear successively in any iteration on both sides. We detect a ray and the ray was slightly missed at a previous stage of the strategy so that the distance to the ray is minimized. This gives rise to a precise slope β of the corresponding ray and only $\beta \in [\alpha, \pi/2]$ has to be considered. The generic calculations for this situation is shown in Figure 4. Any worst case situation (worst case ray) will be covered and we obtain a ratio formula that depends on the strategy by (r, α) and on the slope β for the worst case ray that is currently met. Note that the worst cases will appear in any iteration as usual in these settings, the ratio will also be achieved asymptotically from below, we finally have

$$\frac{|\Pi_s^{p_i'}|}{|s\,t_i'|} \leq C(r,\alpha,\beta) := \frac{r+1}{\cos(\alpha)} \quad \frac{\frac{r}{r-1} + \frac{\sin(\beta-\alpha)}{\sin(\beta+\alpha)}}{\sin(\beta) + \cos(\beta)\tan(\alpha)\frac{r+1}{r-1}}\,. \tag{3}$$

More formally, the best ratio for a cow-path like strategy is now given by

$$\inf_{(r,\alpha)} \left(\sup_{\beta\in[\alpha,\pi/2]} C(r,\alpha,\beta) \right) \text{ for } r > 1 \text{ and } \alpha \in (0,\pi/2]\,. \tag{4}$$

Obviously, this is a multidimensional optimization problem and not so easy to handle and to solve. But we can make use of our formula for attaining very good results.



i) ii)

Figure 5 i) An example for the case that the boundary maximum $\beta = \pi/2$ subsumes all ratios $C(r,\alpha,\beta)$ with the value $r = 2$ and $\alpha = 0.23147\ldots$ taken from [8]. ii) The nature of the general optimization. We have to find pairs (r,α) such that the local maximum $C(r,\alpha,\beta_{\max})$ for some $\beta_{\max}$ shortly after increasing $\beta \in [\alpha,\pi/2]$ and the boundary maximium $C(r,\alpha,\pi/2)$ coincidence. Using (r,α) with $r = 1.978624821$ and $\alpha = 0.166547577$ gives $C(r,\alpha,\beta) < 9.12725$. The value is (almost) attained by $C(r,\alpha,\beta_{\max})$ with $\beta_{\max} \approx 0.190681180073897$ and also by $C(r,\alpha,\pi/2)$.

2.1 Finding an efficient strategy pair (r,α)

Since we have the formulas in our hand we can make use of a computer algebra system to get further insight and we analyse the behaviour of $C(r,\alpha,\beta)$ for $\beta \in [\alpha,\pi/2]$; see Figure 5. We found out that actually the overall best choice is to balance out two worst case situations for the rays w.r.t. β. One is attained shortly after the next return point of the strategy with a slope slightly larger than α. After a *local* maximum at some $\beta_{\max}$ the ratio will decrease and will attain a local minimum until finally increasing again up to a *boundary* maximum at $\beta = \pi/2$ which means the boundary case of a vertical ray! Any reasonable pair (r,α) has to balance out these two worst case situations. Overall and also by interval arithmetic we found a pair $r = 1.978624821$ and $\alpha = 0.166547577$ that guarantees $C(r,\alpha,\beta) < 9.12725$.

3 Better adjustment for the terrain problem

With our fixed strategy from the previous section we can now consider the terrain problem. We follow our strategy as long as there is no terrain. When we met the terrain we will behave differently from [8]. Instead of simply trying to re-use the fully computed path for the return

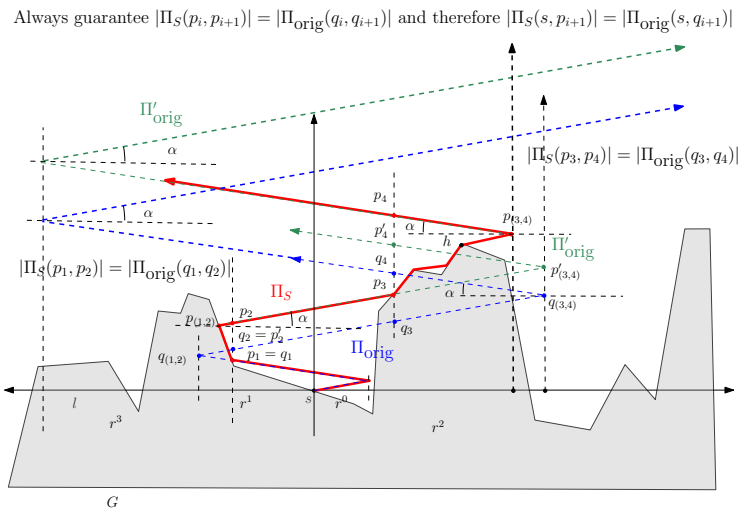


Figure 6 Exemplification of the new zig-zag strategy adaptation for the terrain search problem with X distances r^i and a fixed slope given by angle α. We use our original strategy Π_{orig} (or its copies Π'_{orig}) and in case of collisions with the terrain at some point p_i we use the terrain to climb up. The new strategy Π_S attains the same X-coordinates in the free space back movement, compare point p_4 in comparison to q_4 on Π_{orig}. The idea is that we generally guarantee $|\Pi_S(p_i, p_{i+1})| = |\Pi_{\text{orig}}(q_i, q_{i+1})|$ and therefore $|\Pi_S(s, p_{i+1})| = |\Pi_{\text{orig}}(s, q_{i+1})|$ for such pairs of points. For example when the terrain is visited for the first time on the current intermediate plan at some point p_3 we make use of the budget $|\Pi_{\text{orig}}(q_3, q_4)|$ and climb up along the terrain until we have to follow the α-direction again or we have to return from a point $p_{(3,4)}$ to p_4 in order to locally guarantee that $|\Pi_{\text{orig}}(q_3, q_4)| = |\Pi_S(p_3, p_4)|$. Note that we can always keep track of the rest budget and can compute this return point $p_{(3,4)}$ adequately during the movement. Then we move back into α-direction in the free space and reach p_4 as intended with the overall guarantee $|\Pi_{\text{orig}}(s, q_4)| = |\Pi_S(s, p_4)|$!

we lift our strategy up. By our adjustment we can also analytically guarantee that the ratio will not get worse. For convenience the new adjustment is fully explained in Figure 6.

4 The terrain only improves the ratio

For the guarantee that the visit of the terrain will now help to improve the ratios we consider our two worst case situations exemplified and sketched by the situation for the rays r_1 and r_2 in Figure 7. The first observation is that it is sufficient to consider only vertical barriers. If we locally move along the terrain (into one direction), there will always be a final point $h = (x_h, y_h)$ on the terrain of current maximal height that is visited last before we move in the free space and finally make use of a return point to move back. A simple vertical segment barrier B from $(x_h, 0)$ to h will be the best (lean) obstacle that will be able to decrease the length of optimal shortest paths to rays. The point h can now just be reached by the segment $s\,h$. Additionally, w.r.t. our strategy starting at $p_{(2i-1)}$ in presence of (only) B we will follow the α-direction first and will hit the terrain at $h' \in B$ and then we move vertically up to h. By triangle inequality this local movement has length not smaller than the original movement from $p_{(2i-1)}$ along the terrain to this final height point h and we have to use more from the budget. In other words in the presence of simple barriers B the path length to attain height points h with our strategy S' is also maximized and rays behind h

will potentially be reached earlier by an adversary.

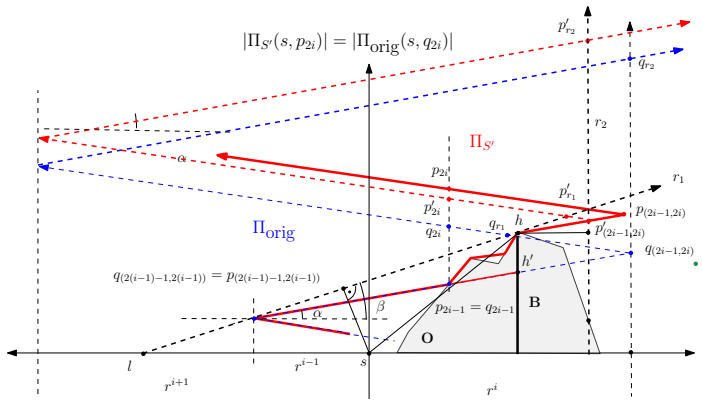


Figure 7 Exchanging obstacles O by single vertical barriers B will cause the strategy to use more from the budget by triangle inequality. The next return point $p'_{(2i-1,2i)}$ will thus get *closer* to the start and the detection of a ray r_1 will be postponed to p'_{r_1}. But in comparison to the original pure strategy the detection of such rays always occurs not later in time (i.e. path length) than the visit at q_{r_1}. For a boundary worst case r_2 related to the corresponding worst case return point $p'_{(2i-1,2i)}$, we can show that the (shortest) path length from s to r_2 in the presence of B is never smaller than the distance r^i. The ray r_2 will also be detected earlier in time at p'_{r_2} in comparison to q_{r_2}.

4.1 Local worst case situation after the return point

If we hit the terrain and move up to some height point h the rays that will be newly detected (slightly) after the next return point will be visited at a higher level. By construction such rays will be detected not later in time (and path length) than before. For example in Figure 7 we consider the first ray r_1 (related with angle β) newly detected after the return. By construction we attain a smaller ratio than the original pure strategy since the path length for the detection of r_1 is smaller. This also holds if β increases for a while.

4.2 Boundary worst case situation

Changing the return point (for example from $q_{(2i-1,2i)}$ to $p'_{(2i-1,2i)}$ in Figure 7) has impact on the boundary worst case. The change of the return point was caused by a terrain visit at a reasonable height. We can show by geometric arguments that the path length to a new (vertical) worst case ray is never smaller than the distance to the vertical boundary worst case ray for the pure strategy in this iteration of the strategy. For example in Figure 7 a boundary ray r_2 that makes use of a new return point $p'_{(2i-1,2i)}$ is detected a bit earlier (at p'_{r_2}) than originally (at q_{r_2}). But we can also show that the shortest distance to r_2 where the terrain is used as an obstacle is never smaller than the distance r^i to the original vertical boundary worst case ray at $q_{(2i-1,2i)}$. This is not so easy to see in general but finally manageable by some nice geometric arguments taking also the values (r, α) into account; see [17].

References

1 Steve Alpern and Shmuel Gal. *The theory of search games and rendezvous*. Springer, 2003.
2 Spyros Angelopoulos. Online search with a hint. *Information and Computation*, 295:105091, 2023.
3 R. Baeza-Yates, J. Culberson, and G. Rawlins. Searching in the plane. *Inform. Comput.*, 106:234–252, 1993.
4 Anatole Beck. On the linear search problem. *Israel Journal of Mathematics*, 2(4):221–228, 1964.
5 Anatole Beck and Donald J Newman. Yet more on the linear search problem. *Israel journal of mathematics*, 8(4):419–429, 1970.
6 Prosenjit Bose, Jean-Lou De Carufel, and Stephane Durocher. Searching on a line: A complete characterization of the optimal solution. *Theoretical Computer Science*, 569:24–42, 2015.
7 Marek Chrobak. A princess swimming in the fog looking for a monster cow. *ACM SIGACT News*, 35(2):74–78, 2004.
8 Sarita de Berg, Nathan van Beusekom, Max van Mulken, Kevin Verbeek, and Jules Wulms. Competitive searching over terrains. In *Latin American Symposium on Theoretical Informatics*, pages 254–269. Springer, 2024.
9 Erik D Demaine, Sándor P Fekete, and Shmuel Gal. Online searching with turn cost. *Theoretical computer science*, 361(2-3):342–355, 2006.
10 Steven R Finch. The logarithmic spiral conjecture. *arXiv preprint math/0501133*, 2005.
11 Shmuel Gal. *Search Games*, volume 149 of *Mathematics in Science and Engeneering*. Academic Press, New York, 1980.
12 Christoph Hipke, Christian Icking, Rolf Klein, and Elmar Langetepe. How to find a point on a line within a fixed distance. *Discrete Applied Mathematics*, 93(1):67–73, 1999.
13 Artur Jeż and Jakub Łopuszański. On the two-dimensional cow search problem. *Information Processing Letters*, 109(11):543–547, 2009.
14 Ming-Yang Kao, John H Reif, and Stephen R Tate. Searching in an unknown environment: An optimal randomized algorithm for the cow-path problem. *Information and computation*, 131(1):63–79, 1996.
15 Elmar Langetepe. On the optimality of spiral search. In *Proceedings of the twenty-first annual ACM-SIAM symposium on Discrete Algorithms*, pages 1–12. SIAM, 2010.
16 Elmar Langetepe. Searching for an axis-parallel shoreline. *Theoretical Computer Science*, 447:85–99, 2012.
17 Elmar Langetepe and Florian Gans. Online competitive searching for rays in the half-plane, 2025. URL: `https://arxiv.org/abs/2512.16622`, `arXiv:2512.16622`.
18 Alejandro López-Ortiz and Sven Schuierer. The ultimate strategy to search on m rays? *Theoretical Computer Science*, 261(2):267–295, 2001.
19 Daniel D. Sleator and Robert E. Tarjan. Amortized efficiency of list update and paging rules. *Communications of the ACM*, 28(2):202–208, 1985.

Wataridori is NP-Complete

Suthee Ruangwises[1]

1 **Department of Computer Engineering, Faculty of Engineering, Chulalongkorn University, Bangkok, Thailand**
suthee@cp.eng.chula.ac.th

Abstract

Wataridori is a pencil puzzle that involves drawing paths in a rectangular grid to connect circles into pairs while satisfying several constraints. In this paper, we prove that deciding whether a given Wataridori puzzle has a solution is NP-complete via a reduction from Numberlink, another pencil puzzle that has previously been proved NP-complete.

1 Introduction

Wataridori is a pencil puzzle introduced by Nikoli [25], a Japanese publisher that developed many famous pencil puzzles including Sudoku, Kakuro, and Slitherlink. The puzzle consists of a rectangular grid partitioned into polyominoes called *regions*. (A polyomino is a shape formed by one or more unit squares connected orthogonally.) Some cells contain a circle, and some of these circles contain a number.

The objective of Wataridori is to connect all circles into pairs using paths that go between horizontally or vertically adjacent cells. Paths cannot cross or share a cell with one another. The two circles connected by a path must contain the same number, which must be equal to the total number of regions the path passes through. A path may enter and exit a region at most once. Circles without numbers are treated as wildcards and may represent any number. See Figure 1. Note that it is not required for all cells in the grid to be covered by paths.

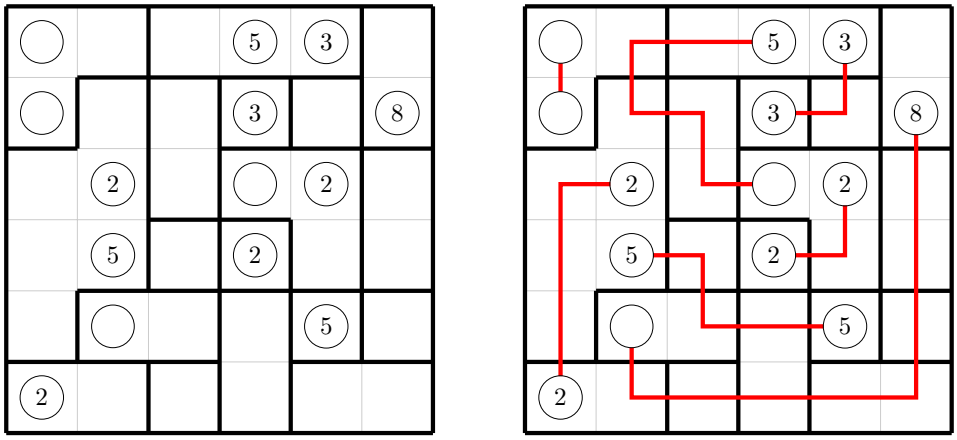


Figure 1 An example of a 6×6 Wataridori puzzle (left) and its solution (right)

1.1 Related Work

Many pencil puzzles have been proved NP-complete, where the decision problem asks whether a given instance has a valid solution. Examples include Chained Block [18], Choco Banana

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

[23], Dosun-Fuwari [16], Fillmat [35], Five Cells [20], Goishi Hiroi [5], Hashiwokakero [4], Herugolf [15], Heyawake [12], Juosan [17], Kakuro [36], Kurodoko [26], Kurotto [17], LITS [6], Makaro [15], Moon-or-Sun [21], Nagareru [21], Nondango [30], Nonogram [34], Norinori [6], Nurikabe [11], Nurimeizu [21], Nurimisaki [19], Nuritwin [22], Pencils [28], Ripple Effect [33], Roma [9], Sashigane [19], Shakashaka [8], Shikaku [33], Slitherlink [36], Sto-Stone [3], Sudoku [36], Suguru [29], Sumplete [10], Tatamibari [2], Tilepaint [36], Toichika [32], Usowan [14], Yin-Yang [7], Yosenabe [13], and Zeiger [31].

The vast majority of NP-completeness proofs for pencil puzzles are obtained via reductions from SAT-type problems, such as 3SAT, 1-in-3 SAT, Planar SAT, and Circuit SAT. Some proofs instead use reductions from Hamiltonian cycle problems. In contrast, direct reductions from one pencil puzzle to another are rare, as the rules of different puzzles typically differ significantly, making such reductions difficult to construct.

1.2 Our Contribution

In this paper, we prove that deciding solvability of a given Wataridori puzzle is NP-complete via direct reductions from Numberlink, another pencil puzzle that has already been proved to be NP-complete.

2 Numberlink

Numberlink (also known as *Arukone*) is a well-known pencil puzzle. While it was popularized by Nikoli, its origins date back to at least 1932 [24]. Despite its appearance being similar to Wataridori, its rules are significantly different.

The puzzle consists of a rectangular grid in which some cells contain numbers, with each number appearing exactly twice. The objective is to connect every pair of cells with the same number by a path that goes between horizontally or vertically adjacent cells. Paths cannot cross or share cells with one another. See Figure 2. Note that, unlike Wataridori, the numbers in Numberlink have no quantitative meaning and merely serve as labels to distinguish pairs of cells (in some Numberlink mobile apps, letters or colors are used instead of numbers).

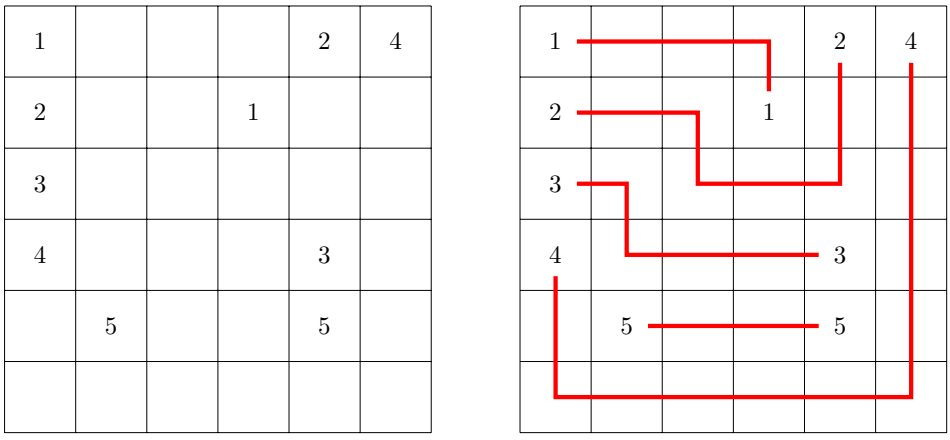


Figure 2 An example of a 6×6 Numberlink puzzle (left) and its solution (right)

There are two main variants of Numberlink: one that requires all cells in the grid to be covered by paths and one that does not. Both variants have been proved to be NP-complete [1, 27].

3 Reduction from Numberlink to Wataridori

As Wataridori clearly belongs to NP, the nontrivial part is to prove its NP-hardness. We do so by constructing a reduction from Numberlink to Wataridori. We consider the variant of Numberlink in which it is not required for all cells in the grid to be covered by paths.

3.1 Constructing a Block Representing Each Cell

Given an $m \times n$ Numberlink puzzle G, we transform it into a $(4k+5)m \times (4k+5)n$ Wataridori puzzle H, where k will be defined later. For each cell of G, we construct a $(4k+5) \times (4k+5)$ block in H. A cell containing a number is represented by a *number block*, while an empty cell is represented by an *empty block*. Examples of a number block and an empty block for the case $k = 2$ are shown in Figures 3 and 4.

The blocks are placed in H in the same arrangement as the corresponding cells in G, resulting in a $(4k+5)m \times (4k+5)n$ Wataridori puzzle.

3.2 Idea of Reduction

Suppose G contains p pairs of numbers. We construct H so that the p paths in a solution of G correspond to p paths in H that pass through different numbers of regions. This property allows us to label the p pairs of endpoints in H with distinct numbers, thereby forcing the pairings in H to match exactly those in G.

Consider the main path in Figure 3 (the one connecting to the center circle labeled x). There are at least three ways for the path to perform a "zig-zag" before exiting the block on the right side, as shown in Figure 5. In these three cases, the path passes through six, eight, and ten distinct regions, respectively. The rightmost region is the region in which the path will remain while traversing empty blocks, until it enters the number block at the other endpoint. This region is also the same as the outermost region of the number block at that endpoint.

For general k, there are at least $k+1$ different ways for the path to perform a "zig-zag" before exiting the block on each side. In these cases, the path passes through $2k+2, 2k+4, 2k+6, \ldots, 4k+2$ different regions, respectively.

For each solution path in G, its corresponding path in H can perform a "zig-zag" in the number blocks at both endpoints. Hence, the total number of regions it passes through can be $4k+3, 4k+5, 4k+7, \ldots, 8k+3$, giving $2k+1$ possible values.

Finally, we choose $k = \lceil (p-1)/2 \rceil$, so that $2k+1 \geq p$. Therefore, we can assign p distinct numbers from the set $\{4k+3, 4k+5, 4k+7, \ldots, 8k+3\}$ to the center circles of the p pairs of number blocks in H corresponding to the p pairs of numbered cells in G. Specifically, if the numbers in G are $1, 2, \ldots, p$, then we assign the number $4k+2i+1$ to the center circle of a number block in H corresponding to a cell labeled i in G ($i = 1, 2, \ldots, p$). This forces the pairings in H to be exactly the same as in G.

Since $2p \leq mn$ (the total number of cells in G), we have $k \leq p/2 \leq mn$, so the size of H is polynomial in m and n. Therefore, the transformation can be performed in polynomial time.

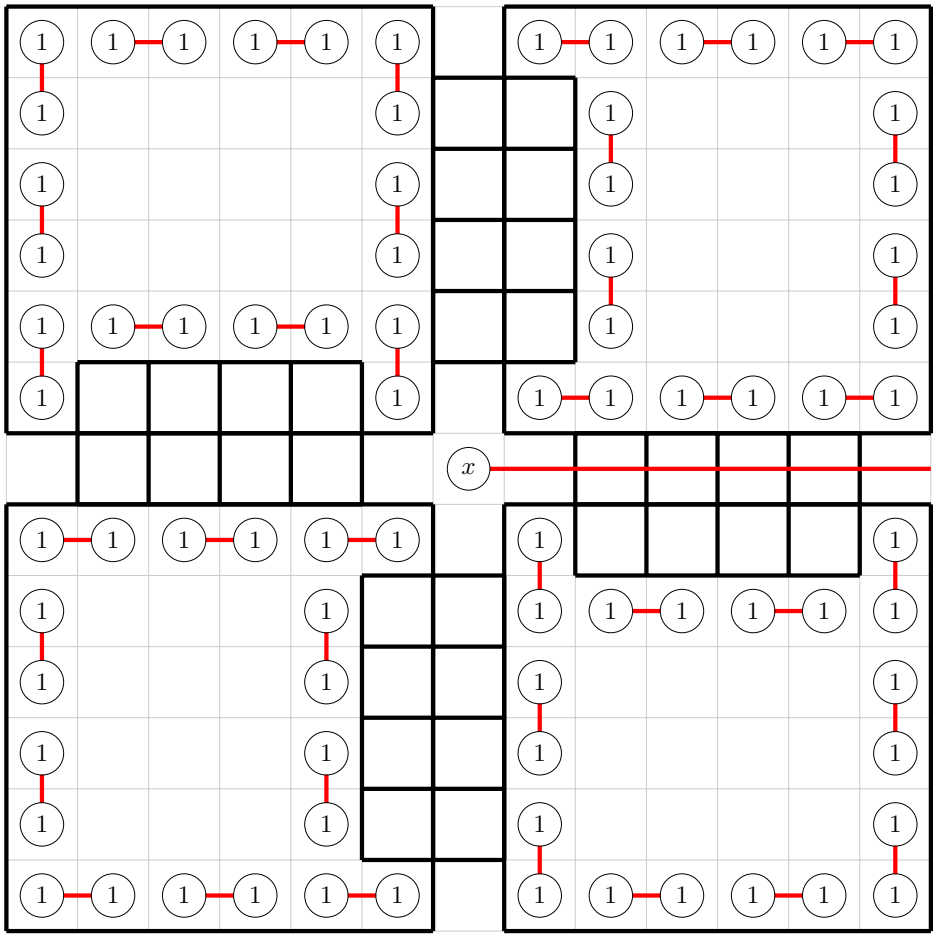


Figure 3 A 13×13 number block in the Wataridori puzzle H for $k = 2$ (bottom), representing a cell labeled x in the Numberlink puzzle G (top). Red lines showing one possible solution corresponding to a path going to the right of the cell in G.

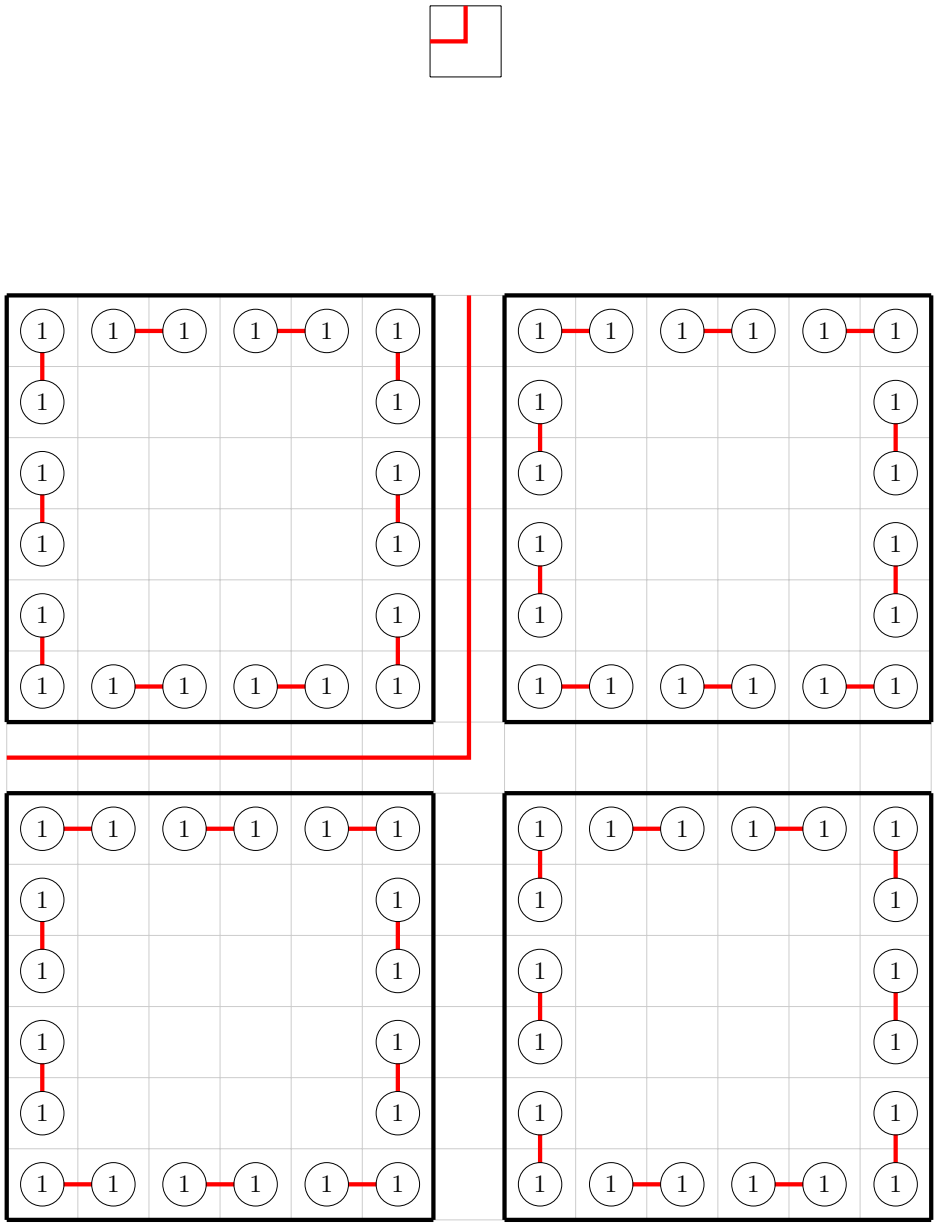


Figure 4 A 13×13 empty block in the Wataridori puzzle H for $k = 2$ (bottom), representing an empty cell in the Numberlink puzzle G (top). Red lines showing one possible solution corresponding to a path entering from the left, turning left, and exiting at the top of the cell in G.

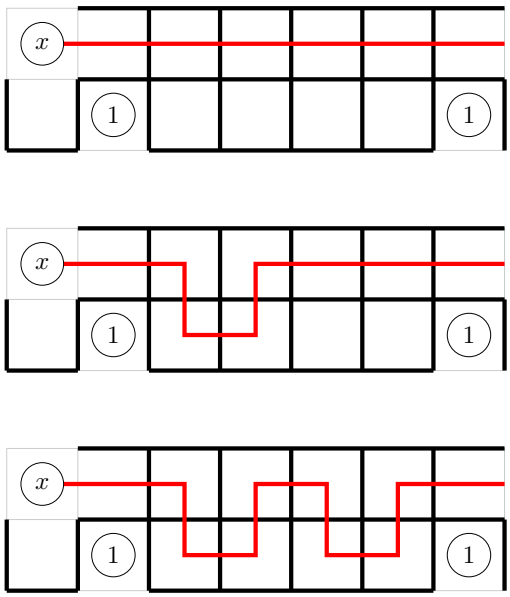


Figure 5 Three ways for the path to exit the block on the right side, passing through six, eight, and ten distinct regions, respectively

3.3 Proof of Correctness

▶ **Theorem 3.1.** *G has a solution if and only if H has a solution.*

Proof. Suppose G has a solution S, with paths $s_1, s_2, \ldots, s_p$ connecting the pairs of numbers $1, 2, \ldots, p$, respectively. For each $i = 1, 2, \ldots, p$, we draw a path in H connecting the pair of circles labeled $4k + 2i + 1$, following the corresponding path in S and performing exactly $i - 1$ "zig-zags" in the number blocks at the two endpoints combined.

The direct path with no "zig-zag" passes through $4k + 3$ regions, and each additional "zig-zag" increases the number of regions by two. Hence, the resulting path passes through exactly $4k + 2i + 1$ regions, satisfying the puzzle constraint. Therefore, H has a solution.

Conversely, suppose H has a solution T. In each number block, there must be a path connecting the center circle that exits the block through one of its four sides. In each empty block, at most one path can enter and exit through two of the four sides (or no path passes through the block). Consequently, the p paths in T induce p paths in G connecting the pairs of numbers $1, 2, \ldots, p$. Therefore, G has a solution. ◀

Hence, Wataridori is NP-complete.

4 Future Work

We proved the NP-completeness of Wataridori via a reduction from another puzzle, Numberlink. Although the reduction is intuitive, it is nontrivial. Future work includes applying similar techniques to prove the NP-completeness of other puzzles via direct reductions from puzzles already known to be NP-complete.

In particular, one possible candidate is *Mintonette*, a puzzle that also involves drawing paths to connect circles in a rectangular grid into pairs under different constraints. It would

be interesting to investigate whether there exists a direct reduction from Numberlink or Wataridori to Mintonette.

Another interesting direction is to study the solution structure of Wataridori. In particular, it remains open whether Wataridori is ASP-complete, that is, whether the problem of deciding if another solution exists given one solution is NP-complete.

Acknowledgement

This work was supported by grants for development of new faculty staff, Ratchadaphiseksomphot Fund, Chulalongkorn University.

References

1 A. Adcock, E.D. Demaine, M.L. Demaine, M.P. O'Brien, F. Reidl, F.S. Villaamil and B.D. Sullivan. Zig-Zag Numberlink is NP-Complete. *Journal of Information Processing*, 23(3): 239–245 (2015).

2 A. Adler, J. Bosboom, E.D. Demaine, M.L. Demaine, Q.C. Liu and J. Lynch. Tatamibari Is NP-Complete. In *Proceedings of the 10th International Conference on Fun with Algorithms (FUN)*, pp. 1:1–1:24 (2020).

3 A. Allen and A. Williams. Sto-Stone is NP-Complete. In *Proceedings of the 30th Canadian Conference on Computational Geometry (CCCG)*, pp. 28–34 (2018).

4 D. Andersson. Hashiwokakero is NP-complete. *Information Processing Letters*, 109(9): 1145–1146 (2009).

5 D. Andersson. HIROIMONO Is NP-Complete. In *Proceedings of the 4th International Conference on Fun with Algorithms (FUN)*, pp. 30–39 (2007).

6 M. Biro and C. Schmidt. Computational complexity and bounds for Norinori and LITS. In *Proceedings of the 33rd European Workshop on Computational Geometry (EuroCG)*, pp. 29–32 (2017).

7 E.D. Demaine, J. Lynch, M. Rudoy and Y. Uno. Yin-Yang Puzzles are NP-complete. In *Proceedings of the 33rd Canadian Conference on Computational Geometry (CCCG)*, pp. 97–106 (2021).

8 E.D. Demaine, Y. Okamoto, R. Uehara and Y. Uno. Computational Complexity and an Integer Programming Model of Shakashaka. *IEICE Trans. Fundamentals*, 97.A(6): 1213–1219 (2014).

9 K. Goergen, H. Fernau, E. Oest and P. Wolf. All Paths Lead to Rome. In *Proceedings of the 24th Japan Conference on Discrete and Computational Geometry, Graphs, and Games (JCDCG^3)*, pp. 228–256 (2026).

10 K. Hatsugai, S. Ruangwises, K. Asano and Y. Abe. NP-Completeness and Physical Zero-Knowledge Proofs for Sumplete, a Puzzle Generated by ChatGPT. *New Generation Computing*, 42(3): 429–448 (2024).

11 M. Holzer, A. Klein and M. Kutrib. On The NP-Completeness of The Nurikabe Pencil Puzzle and Variants Thereof. In *Proceedings of the 3rd International Conference on Fun with Algorithms (FUN)*, pp. 77–89 (2004).

12 M. Holzer and O. Ruepp. The Troubles of Interior Design–A Complexity Analysis of the Game Heyawake. In *Proceedings of the 4th International Conference on Fun with Algorithms (FUN)*, pp. 198–212 (2007).

13 C. Iwamoto. Yosenabe is NP-complete. *Journal of Information Processing*, 22(1): 40–43 (2014).

14 C. Iwamoto and M. Haruishi. Computational Complexity of Usowan Puzzles. *IEICE Trans. Fundamentals*, 101.A(9): 1537–1540 (2018).

15 C. Iwamoto, M. Haruishi and T. Ibusuki. Herugolf and Makaro are NP-complete. In *Proceedings of the 9th International Conference on Fun with Algorithms (FUN)*, pp. 24:1–24:11 (2018).

16 C. Iwamoto and T. Ibusuki. Dosun-Fuwari is NP-complete. *Journal of Information Processing*, 26: 358–361 (2018).

17 C. Iwamoto and T. Ibusuki. Polynomial-Time Reductions from 3SAT to Kurotto and Juosan Puzzles. *IEICE Trans. Inf. & Syst.*, 103.D(3): 500–505 (2020).

18 C. Iwamoto and T. Ide. Chained Block is NP-Complete. *IEICE Trans. Inf. & Syst.*, 107.D(3): 320–324 (2024).

19 C. Iwamoto and T. Ide. Computational Complexity of Nurimisaki and Sashigane. *IEICE Trans. Fundamentals*, 103.A(10): 1183–1192 (2020).

20 C. Iwamoto and T. Ide. Five Cells and Tilepaint are NP-Complete. *IEICE Trans. Inf. & Syst.*, 105.D(3): 508–516 (2022).

21 C. Iwamoto and T. Ide. Moon-or-Sun, Nagareru, and Nurimeizu are NP-Complete. *IEICE Trans. Fundamentals*, 105.A(9): 1187–1194 (2022).

22 C. Iwamoto and A. Shibata. Nuritwin is NP-complete. In *Proceedings of the 27th China-Japan Conference on Discrete and Computational Geometry, Graphs, and Games (CJCDCG^3)*, pp. 34–25 (2025).

23 C. Iwamoto and T. Tokunaga. Choco Banana is NP-Complete. *IEICE Trans. Fundamentals*, 107.A(9): 1488–1491 (2024).

24 A. Janko and O. Janko. Arukone, Number Link. `https://www.janko.at/Raetsel/Arukone/index.htm`

25 A. Janko and O. Janko. Wataridori. `https://www.janko.at/Raetsel/Wataridori/index.htm`

26 J. Kölker. Kurodoko is NP-Complete. *Journal of Information Processing*, 20(3): 694–706 (2012).

27 J.F. Lynch. The equivalence of theorem proving and the interconnection problem. *ACM SIGDA Newsletter*, 5(3): 31–36 (1975).

28 D. Packer, S. White and A. Williams. A Paper on Pencils: A Pencil and Paper Puzzle - Pencils is NP-Complete. In *Proceedings of the 30th Canadian Conference on Computational Geometry (CCCG)*, pp. 35–41 (2018).

29 L. Robert, D. Miyahara, P. Lafourcade, L. Libralesso and T. Mizuki. Physical zero-knowledge proof and NP-completeness proof of Suguru puzzle. *Information and Computation*, 285(B): 104858 (2022).

30 S. Ruangwises. Nondango is NP-Complete. In *Proceedings of the 40th European Workshop on Computational Geometry (EuroCG)*, pp. 1:1–1:10 (2024).

31 S. Ruangwises. NP-Completeness and Physical Zero-Knowledge Proofs for Zeiger. In *Proceedings of the 19th International Conference and Workshops on Algorithms and Computation (WALCOM)*, pp. 312–325 (2025).

32 S. Ruangwises. Toichika is NP-Complete. In *Proceedings of the 25th Indonesia-Japan Conference on Discrete and Computational Geometry, Graphs, and Games (IJCDCG^3)*, p. 96 (2023).

33 Y. Takenaga, S. Aoyagi, S. Iwata and T. Kasai. Shikaku and Ripple Effect are NP-Complete. *Congressus Numerantium*, 216: 119–127 (2013).

34 N. Ueda and T. Nagao. NP-completeness Results for NONOGRAM via Parsimonious Reductions. Technical Report TR96-0008, Department of Computer Science, Tokyo Institute of Technology (1996).

35 A. Uejima and H. Suzuki. Fillmat is NP-Complete and ASP-Complete. *Journal of Information Processing*, 23(3): 310–316 (2015).

36 T. Yato and T. Seta. Complexity and Completeness of Finding Another Solution and Its Application to Puzzles. *IEICE Trans. Fundamentals*, 86.A(5): 1052–1060 (2003).

Simplicial Approximation to CW Complexes with Spherical Delaunay Triangulations

Raphaël Tinarrage[1]

1 **Institute of Science and Technology Austria, Klosterneuburg, Austria**
raphael.tinarrage@ist.ac.at

Abstract

Simplicial approximation provides a framework for constructing simplicial complexes that are homotopy equivalent to a given manifold, provided a CW structure is explicitly known. However, its conventional implementation quickly becomes intractable on a computer: barycentric subdivision produces poorly shaped simplices, and the star condition introduces many vertices. To address these limitations, this article develops a subdivision scheme based on spherical Delaunay triangulations, which attains better refinement properties than barycentric subdivisions. Moreover, the star condition is reframed as two independent problems, one geometric and the other combinatorial, respectively tackled in the language of locally equiconnected spaces and the list homomorphism problem, allowing an exponential reduction in the number of vertices. Via a prototype implementation, we obtain simplicial complexes homotopy equivalent to Grassmannians and Stiefel manifolds up to dimension 5.

Related Version *Full version:* https://arxiv.org/abs/2112.07573

1 Introduction

Topology software. Over the past two decades, software for computations on simplicial complexes has advanced substantially. Existing libraries span a wide range of goals, from algebra to geometry and data analysis. They have underpinned numerous concrete advances: proofs and counterexamples in 3-manifold topology with `Regina` and `SnapPy` [18, 22]; large-scale enumerations that tested conjectures with `BISTELLAR` and `Twister` [10, 8]; new homotopy and cohomology computations in `Kenzo` and `HAP` [23, 26]; and persistent homology pipelines that surfaced new properties of datasets with `GUDHI`, `Ripser`, or `TTK` [45, 7, 52] among others.

On the other hand, computational topology lacks *explicit* examples of triangulated manifolds, as noted in [9, 32, 6, 51]. By explicit, we mean stored on a computer as a list of simplices, or obtainable in reasonable time by an implemented algorithm. This is especially striking for the Grassmannians $\mathcal{G}(d,n)$, the Stiefel manifolds $\mathcal{V}(d,n)$, and the special orthogonal groups $\mathrm{SO}(n)$. All known triangulations are "accidental", i.e., arising from special homeomorphisms with known spaces, such as $\mathcal{G}(1,n) \cong \mathbb{R}P^{n-1}$, the projective space.

Related work. Traditionally, explicit triangulations of manifolds are obtained by two main approaches. The first is combinatorial and relies on specific descriptions of the manifolds under consideration. For example, small triangulations of $\mathbb{R}P^n$ have recently been constructed by exploiting its realization from a symmetric polytope [4]. Topological properties can also guide the enumeration of combinatorial manifolds, as in the work of Lutz [42, 50]. However, combinatorial complexity grows rapidly with the dimension. Thus, in dimensions 3 and 4, more structured constructions are preferred, based on layered triangulations [38, 37, 39], Dehn fillings [24], Heegaard diagrams [27, 36, 28], and Kirby diagrams [16, 17]. The present work follows this perspective by exploiting the *CW structure* of the spaces involved.

Sampling-based methods offer a different viewpoint. A large body of work studies how to reconstruct an embedded manifold $\mathcal{M} \subset \mathbb{R}^n$ from a finite sample $X \subset \mathbb{R}^n$. The goal is usually

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

not to recover a triangulation of $\mathcal{M}$ itself, but rather a simplicial complex with the same homotopy type. A standard construction is the Čech complex. Recovery is guaranteed when the sampling density is sufficiently fine relative to an appropriate condition number, typically the reach of $\mathcal{M}$ [46]. Quantitative refinements use less restrictive geometric quantities, such as the μ-reach [19, 40], weak feature size [20], local feature size [21], convexity defects [5], and convexity radius [30]. In higher dimensions, however, the Čech complex may be prohibitively expensive to compute, and one may instead use the Vietoris–Rips complex, which depends only on pairwise distances. Following the foundational results of Hausmann and Latschev [35, 41], quantitative guarantees have been established in terms of the reach [43, 44], the μ-reach [40], and convexity defects [5]. Other approaches rely on Delaunay complexes [12, 13, 11, 1], witness complexes [49, 33, 14, 15], and metric thickenings [2, 3].

These methods nevertheless remain computationally demanding. Vietoris–Rips complexes are the cheapest to compute, but their guarantees are weaker and they typically require many vertices. Čech complexes present the same difficulty, as they may contain a large number of simplices, potentially of high dimension. In contrast, more refined constructions such as Delaunay complexes require substantial computation time and scale poorly with dimension.

Contributions. In Algorithm 1 we present an implementation of *simplicial approximation to CW complexes*, a framework well established in theory yet overlooked in practice. Given a CW structure on a topological space, the algorithm outputs a *homotopy equivalent* simplicial complex. Although not homeomorphic, it still suffices for many of the applications mentioned.

We implemented two versions of the algorithm, using either global or local refinement. Only the first is currently proven to terminate, but the second often produces smaller complexes. In both cases, when the algorithm terminates, the output is correct.

▶ **Theorem 1.1.** *With global refinement, Algorithm 1 terminates and produces a simplicial complex homotopy equivalent to the given CW complex. With local refinement, if it terminates, then the output is also homotopy equivalent to the input CW complex.*

The software is available online.[1] The repository includes simplicial complexes homotopy equivalent to $\mathcal{G}(2,4)$ and $\mathcal{V}(2,4)$ (of dimensions 4 and 5, respectively), which were not available prior to this work. We intend to address higher-dimensional examples in future work by increasing computational resources; the current results were obtained on a personal laptop.

2 Overview and discussion of the construction

CW complexes. The theoretical background can be found in Hatcher's book [34, Section 2.C]. A *CW complex* of dimension n is a topological space X endowed with a decomposition $X = X^n \supset X^{n-1} \supset \cdots \supset X^0$ where X^0 is finite and each X^d is equal to a disjoint union

$$X^d = X^{d-1} \coprod_{1 \le i \le m(d)} e_i^d,$$

where each e_i^d, called a *d-cell*, is homeomorphic to an open ball of dimension d, and $m(d)$ is their number. It is required that the homeomorphism extends to the closed ball B^d, yielding a map $\Phi_i^d \colon B^d \to \bar{e}_i^d \subset X^d$, called the *characteristic map*. Its restriction to the boundary of

[1] Fully implemented prototype in Python: `https://github.com/raphaeltinarrage/cw2simp`

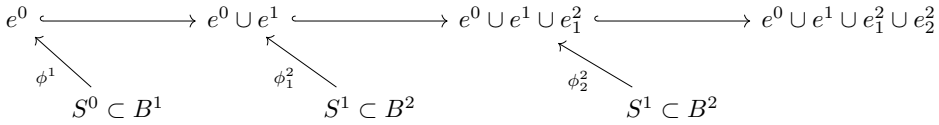
B^d—i.e., the sphere S^{d-1}—is called the *gluing map* and is denoted $\phi_i^d \colon S^{d-1} \to X^{d-1}$. In other words, X^d is homeomorphic to the gluing of d-balls along their boundary on X^{d-1}:

$$X^d \cong X^{d-1} \cup_{\phi_1^d} B^d \cup_{\phi_2^d} B^d \cup \cdots \cup_{\phi_{m(d)}^d} B^d.$$

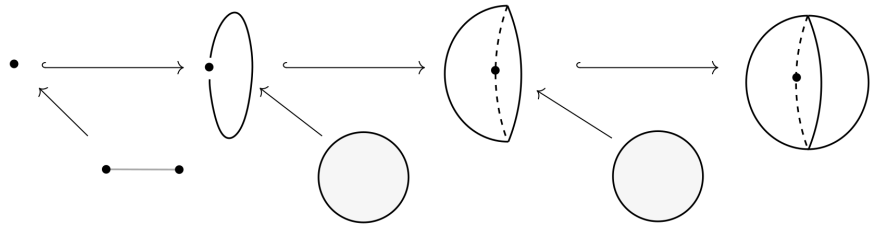
Sketch of the approach. One can "convert" a CW complex into a simplicial complex by gluing *simplicial balls* instead of cells; the idea is sketched in Figure 1. The construction proceeds inductively on the dimension. Suppose we have already built a simplicial complex L^d homotopy equivalent to the d-skeleton X^d; we denote by $|L^d|$ its geometric realization. For each $(d+1)$-cell e_i^{d+1}, we perform the following steps:

- We apply *simplicial approximation* (see below) to the gluing map $\phi_i^{d+1} \colon S^d \to |L^d|$. This yields a triangulation K_i^d of S^d and a simplicial map $g_i^{d+1} \colon K_i^d \to L^d$ homotopic to ϕ_i^{d+1}.
- From K_i^d, we construct a simplicial ball $B(K_i^d)$ with boundary K_i^d and glue it to L^d along this boundary via g_i^{d+1}. This amounts to the simplicial mapping cone of g_i^{d+1}.

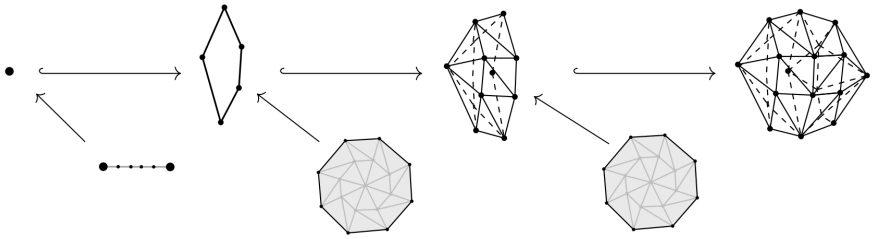
After all $(d+1)$-cells have been attached, the resulting complex L^{d+1} is homotopy equivalent to the skeleton X^{d+1}, and the procedure can be iterated in the next dimension.



(a) Diagram of a CW structure on S^2. If only one d-cell is attached, we omit the subscript in e_i^d.



(b) Geometric visualization of the diagram above.



(c) Simplicial approximation of the diagram.

Figure 1 Schematic view of the simplicial approximation procedure for CW complexes.

Implementation. Conventionally, the simplicial approximation of $\phi_i^{d+1} \colon S^d \to |L^d|$ is obtained as follows: one starts from any triangulation K of S^d and repeatedly applies barycentric subdivision until the triangulation is fine enough. More precisely, one stops when the map satisfies the *star condition*: each closed star of a vertex $v \in K$ is mapped by ϕ_i^{d+1} into the open star of some vertex $w \in L^d$. The assignment of vertices $v \mapsto w$ defines a simplicial map g_i^{d+1} homotopic to ϕ_i^{d+1} (through a homotopy that is linear on each simplex).

In practice, this approach quickly becomes intractable for two reasons. First, barycentric subdivision introduces many vertices: a d-simplex is converted into a complex with $2^{d+1}-1$ vertices. In Section 3, we propose to use instead *Delaunay complexes*. Secondly, the star condition itself is too coarse: each facet of L^d requires $d+1$ vertices of K that are mapped into it. In Section 4 we avoid the star condition altogether by decomposing it into a geometric-combinatorial problem. The full procedure is formalized in Algorithm 1.

Algorithm 1: Simplicial approximation to CW complexes with Delaunay refinement

Input: CW complex X of dimension n with cells e_i^d, attaching maps ϕ_i^d, scalars λ_i
Output: simplicial complex L endowed with a homotopy equivalence $X \to |L|$

$L \leftarrow$ discrete simplicial complex on the 0-cells of X;
for $d = 1, \dots, n$ **do**
  foreach *d-cell e_i^d with attaching map $\phi_i^d \colon S^{d-1} \to X^{d-1}$* **do**
    compose ϕ_i^d with $X^{d-1} \to |L|$ to obtain $f_i^d \colon S^{d-1} \to |L|$;
    use Algorithm 2 (simplicial approximation to a map) to obtain $g_i^d \colon K_i \to L$ homotopic to f_i^d (with global or local refinement, and parameter λ_i);
    construct a simplicial ball $B(K_i)$ and glue it to L along K_i via g_i^d, i.e., let $L \leftarrow L \cup_{g_i^d} B(K_i)$;
    endow $|L \cup_{g_i^d} B(K_i)|$ with an equiconnecting map or a local motion planner;
return L;

3 Successive Refinements of Spherical Delaunay Triangulations

Let X be a finite subset of the unit sphere $S^d \subset \mathbb{R}^{d+1}$. A subset of $d+1$ points has the *empty circle property* if its circumscribing (open) geodesic ball is empty of points of X. The collection of all such subsets forms the facets of an abstract simplicial complex $\mathrm{Del}(X)$, called the *spherical Delaunay complex*. Under the genericity assumption that no subset of $d+2$ points lies on the same geodesic sphere, $\mathrm{Del}(X)$ is naturally embedded in $\mathbb{R}^{d+1}$.

Given an initial Delaunay complex $\mathrm{Del}(X_0)$ on $X_0 \subset S^d$, one obtains a finer triangulation by choosing additional points $Y_0 \subset S^d$ and building the Delaunay complex on $X_1 = X_0 \cup Y_0$. The new points are called *Steiner points* and this procedure is known as a *Delaunay refinement*. This construction can be iterated, yielding a sequence of Steiner points Y_0, Y_1, Y_2, ... and complexes $\mathrm{Del}(X_0)$, $\mathrm{Del}(X_1)$, $\mathrm{Del}(X_2)$, ...

To build complexes with arbitrarily small edge lengths, we introduce several families of refinements (see Figure 2): we let the Steiner points Y_i be the **barycenters** of all simplices of $\mathrm{Del}(X_i)$; the **midpoints** of its edges; the **minicenters** (centers of minimal enclosing balls) of the facets (maximal simplices); or the **centroids** (barycenters) of only the facets.

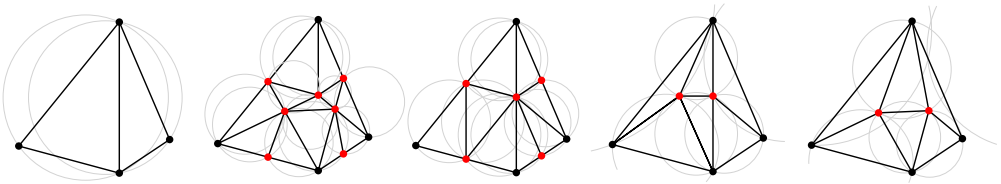
Note that adding a point to a Delaunay triangulation can increase the maximal edge length. As a consequence, it is not immediate that simplices become smaller under refinement. Instead, we show that the *maximal (spherical) circumradius* of $\mathrm{Del}(X_i)$, denoted $\rho_{\mathrm{circ}}(\mathrm{Del}(X_i))$, decreases monotonically to zero. Thus, the maximal diameter of the simplices does too.

▶ **Theorem 3.1.** *Assume $X_0 \subset S^d$ is sufficiently dense so that the maximal circumradius satisfies $\rho_{\mathrm{circ}}(\mathrm{Del}(X_0)) \leq \pi/3$. Then the n^{th} iteration of Delaunay refinement satisfies*

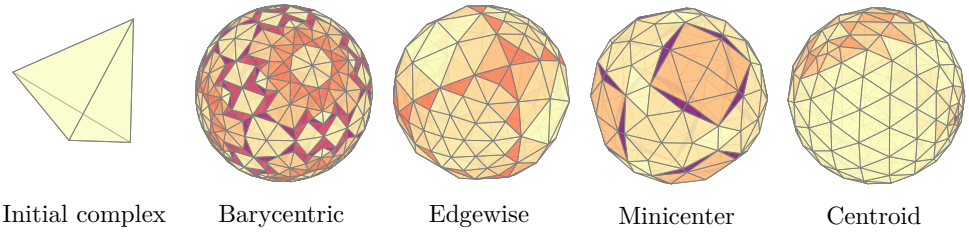
$$\rho_{\mathrm{circ}}(\mathrm{Del}(X_n)) < \alpha^n \rho_{\mathrm{circ}}(\mathrm{Del}(X_0)),$$

where $\alpha = \alpha' / \cos(\rho_{\mathrm{circ}}(\mathrm{Del}(X_0)))$, and α' depends on the chosen refinement, as given below:

Refinement	Edgewise	Minicenter	Centroid
α'	$1/\sqrt{2}$	$1/\sqrt{2}$	$d/(d+1)$



(a) Local picture: A Delaunay complex on a set of four points in $\mathbb{R}^2$ and the four refinements proposed.



(b) Global picture: Spherical Delaunay complexes after three or four refinements of a triangulation of S^2.

Figure 2 We employ Delaunay refinement as a subdivision scheme for triangulations of S^d.

4 Simplicial Approximation as a List Homomorphism Problem

Given a continuous map $f\colon S^d \to |L|$, we seek a triangulation K of S^d and simplicial map $g\colon K \to L$ such that $g(v) \in \mathrm{carr}(f(v))$ (the *carrier* of $f(v)$) for all vertices. This is a purely combinatorial question that we address as a constraint satisfaction problem. Jointly, we ensure the existence of a homotopy between f and $|g|$, in the framework of motion planners.

Constructing the homotopy. Following Farber [29], a *local motion planner* on a space Y is a pair (U, Π) where $U \subset Y \times Y$ is open and $\Pi\colon U \times [0,1] \to Y$ is a continuous map such that $\Pi(x,y,0) = x$ and $\Pi(x,y,1) = y$ for all $(x,y) \in U$. We make here an additional requirement: the set U contains the diagonal in $Y \times Y$. Planners allow us to test the homotopy between maps. Indeed, two maps $a, b\colon X \to Y$ are homotopic whenever $(a(x), b(x)) \in U$ for all $x \in X$; a homotopy is given by $H(x,t) = \Pi(a(x), b(x), t)$. We say that the maps are U*-close*.

The spaces $Y = |L|$ considered in this article are obtained as simplicial mapping cones $L^d = L^{d-1} \cup_{g_i^d} B(K_i)$; see Section 2. We endow them recursively with a local motion planner.

▶ **Theorem 4.1.** *Let $g\colon K \to L$ be a simplicial map where K is a triangulation of S^d and $|L|$ is endowed with a local motion planner. Then $|L \cup_g B(K)|$ admits a local motion planner.*

Planner condition. We now consider a complex $L = L_0 \cup_{g_0} B(K_0)$ endowed with a planner (U, Π). Given a triangulation K of S^d, we ask whether it is fine enough so that the planner can be applied. More precisely, for each facet $\sigma = [v_0, \dots, v_d] \in K$ we look for an assignment $v_i \mapsto g(v_i)$ such that, on $|\sigma|$, the continuous map f and the linear map $|g|$ are U-close.

Let us assume for simplicity that $f(|\sigma|)$ is contained in the last cell $B(K_0)$ of $L = L_0 \cup_{g_0} B(K_0)$; the general case is treated recursively along the filtration. During the proof

of Theorem 4.1, we describe U explicitly. In particular, for points x, y in the ball $|B(K_0)|$, the pair lies in U whenever x and y are sufficiently close to the origin or when they are not antipodal ($x/\|x\| \neq -y/\|y\|$). We guarantee this condition through the following estimate.

▶ **Lemma 4.2.** *Let $\sigma = [v_0, \dots, v_d]$ be a simplex in $\mathbb{R}^n$ and $f\colon |\sigma| \to \mathbb{R}^m$ a λ-Lipschitz map. For a nonempty subset $I \subset \{0, \dots, d\}$, $f(|\sigma|)$ is included in the closed ball $B(c_I, \lambda r_I)$ where*

$$c_I = \frac{1}{|I|} \sum_{i \in I} f(v_i) \quad \text{and} \quad r_I = \max_{0 \le k \le d} \frac{1}{|I|} \sum_{i \in I} \|v_k - v_i\|.$$

If σ is small enough, Lemma 4.2 and the description of U ensure that the required assignment g exists; see Figure 3. We refer to this requirement on σ as the *planner condition*.

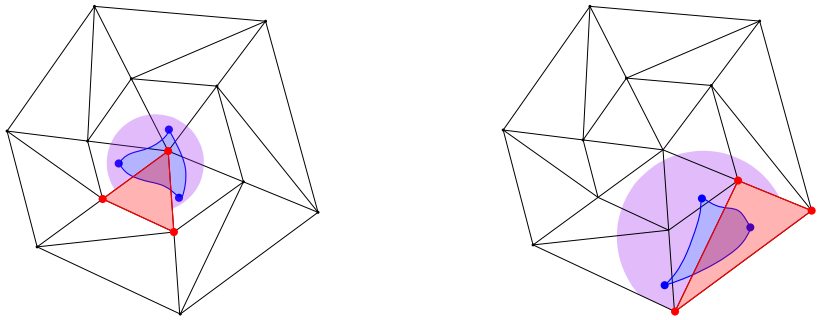


Figure 3 For every facet $\sigma \in K$, the *planner condition* checks whether the image $f(|\sigma|)$ (in blue) is U-close to a simplex τ of L (in red). Lemma 4.2 allows us to enclose $f(|\sigma|)$ in a ball (in purple).

List homomorphism problem. Once the planner condition is satisfied, we obtain for each facet $\sigma \in K$ a collection of admissible simplices $\tau_1, \tau_2, \dots$ of L. From these, we derive for each vertex $v \in V(K)$ a set $\mathcal{L}(v) \subset V(L)$ of admissible images, defining a map $\mathcal{L}\colon V(K) \to \mathcal{P}(V(L))$ to the power set. Given these lists of candidates, the remaining task is to find a simplicial map g with $g(v) \in \mathcal{L}(v)$ for every vertex v. The planner ensures that g is homotopic to f.

In this form, the problem is essentially an instance of the *list homomorphism problem*, denoted `LHom` [31, 25]. We treat it as a constraint satisfaction problem and solve it using the `CP-SAT` solver from the `OR-Tools` suite [47, 48]. If the resulting problem is unsatisfiable, we refine the triangulation K and repeat the procedure, as formalized in Algorithm 2.

Algorithm 2: Simplicial approximation with Delaunay refinement and simplification

Input: map $f\colon S^d \to |L|$, triangulation K of S^d, planner (U, Π) on L, scalar $\lambda > 0$
Output: refinement of K and simplicial map $g\colon K \to L$ homotopic to f

```
repeat
    compute admissible lists L(v) ⊂ V(L) via the planner condition;
    solve LHom with a SAT solver to obtain g;
    if no solution then
        apply Delaunay refinement on K (global, or only facets blamed by LHomDrop);
until a solution g is found;
apply Delaunay simplification to (K, g);
return (K, g);
```

Local Delaunay refinements. A drawback of this method is that it refines the entire complex, even when the obstruction is localized on a few facets. To address this, we introduce a *local refinement* strategy. If the solver reports that the instance is infeasible, we instead consider a weaker problem that we call `LHomDrop`: determine a minimal set of facets $\sigma_0, \dots, \sigma_k$ of K whose removal makes the instance feasible. We then perform a local refinement by inserting Steiner points (see Section 3) only on the facets blamed by the solver. In the same spirit, when enforcing the planner condition, we refine only those facets on which the condition fails.

References

1 Eddie Aamari and Clément Levrard. Stability and minimax optimality of tangential Delaunay complexes for manifold reconstruction. *Discrete & Computational Geometry*, 59(4):923–971, 2018. doi:10.1007/s00454-017-9962-z.

2 Michał Adamaszek, Henry Adams, and Florian Frick. Metric reconstruction via optimal transport. *SIAM Journal on Applied Algebra and Geometry*, 2(4):597–619, 2018. doi:10.1137/17M1148025.

3 Henry Adams and Joshua Mirth. Metric thickenings of Euclidean submanifolds. *Topology and its Applications*, 254:69–84, 2019. doi:10.1016/j.topol.2018.12.014.

4 Karim Adiprasito, Sergey Avvakumov, and Roman Karasev. A subexponential size triangulation of $\mathbb{R}P^n$. *Combinatorica*, 42(1):1–8, 2022. doi:10.1007/s00493-021-4602-x.

5 Dominique Attali, André Lieutier, and David Salinas. Vietoris–Rips complexes also provide topologically correct reconstructions of sampled shapes. *Computational Geometry*, 46(4):448–465, 2013. 27th Annual Symposium on Computational Geometry (SoCG 2011). doi:10.1016/j.comgeo.2012.02.009.

6 Rubén Ballester, Ernst Röell, Daniel Bīn Schmid, Mathieu Alain, Sergio Escalera, Carles Casacuberta, and Bastian Rieck. MANTRA: The Manifold Triangulations Assemblage. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2025. URL: https://openreview.net/pdf?id=X6y5CC44HM.

7 Ulrich Bauer. Ripser: efficient computation of Vietoris–Rips persistence barcodes. *Journal of Applied and Computational Topology*, 5(3):391–423, June 2021. URL: http://dx.doi.org/10.1007/s41468-021-00071-5, doi:10.1007/s41468-021-00071-5.

8 Mark Bell, Tracy Hall, and Saul Schleimer. Twister (computer software), 2014. URL: https://github.com/MarkCBell/twister.

9 Bruno Benedetti and Frank H Lutz. Random discrete Morse theory and a new library of triangulations. *Experimental Mathematics*, 23(1):66–94, 2014. doi:10.1080/10586458.2013.865281.

10 Anders Björner and Frank H Lutz. Simplicial manifolds, bistellar flips and a 16-vertex triangulation of the Poincaré homology 3-sphere. *Experimental Mathematics*, 9(2):275–289, 2000. doi:10.1080/10586458.2000.10504652.

11 Jean-Daniel Boissonnat, Ramsay Dyer, and Arijit Ghosh. Delaunay triangulation of manifolds. *Foundations of Computational Mathematics*, 18(2):399–431, 2018. doi:10.1007/s10208-017-9344-1.

12 Jean-Daniel Boissonnat and Arijit Ghosh. Manifold reconstruction using tangential Delaunay complexes. In *Proceedings of the Twenty-Sixth Annual Symposium on Computational Geometry*, SoCG '10, page 324–333, New York, NY, USA, 2010. Association for Computing Machinery. doi:10.1145/1810959.1811013.

13 Jean-Daniel Boissonnat and Arijit Ghosh. Manifold reconstruction using tangential Delaunay complexes. *Discrete & Computational Geometry*, 51(1):221–267, 2014. doi:10.1007/s00454-013-9557-2.

14 Jean-Daniel Boissonnat, Leonidas J. Guibas, and Steve Y. Oudot. Manifold reconstruction in arbitrary dimensions using witness complexes. In *Proceedings of the Twenty-Third Annual Symposium on Computational Geometry*, SCG '07, page 194–203, New York, NY, USA, 2007. Association for Computing Machinery. doi:10.1145/1247069.1247106.

15 Jean-Daniel Boissonnat, Leonidas J. Guibas, and Steve Y. Oudot. Manifold reconstruction in arbitrary dimensions using witness complexes. *Discrete & Computational Geometry*, 42(1):37–70, July 2009. doi:10.1007/s00454-009-9175-1.

16 Rhuaidi Burke. Practical software for triangulating and simplifying 4-manifolds. *Journal of Computational Geometry*, 16(2):109–144, Jul. 2025. doi:10.20382/jocg.v16i2a4.

17 Rhuaidi Antonio Burke, Benjamin A. Burton, and Jonathan Spreer. Small triangulations of 4-manifolds and the 4-manifold census. In Oswin Aichholzer and Haitao Wang, editors, *41st International Symposium on Computational Geometry (SoCG 2025)*, volume 332 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 28:1–28:16, Dagstuhl, Germany, 2025. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.SoCG.2025.28.

18 Benjamin A. Burton, Ryan Budney, William Pettersson, et al. Regina: Software for low-dimensional topology, 2021. URL: http://regina-normal.github.io/.

19 Frédéric Chazal, David Cohen-Steiner, and André Lieutier. A sampling theory for compact sets in Euclidean space. *Discrete & Computational Geometry*, 41(3):461–479, 2009. doi:10.1007/s00454-009-9144-8.

20 Frédéric Chazal and André Lieutier. Stability and computation of topological invariants of solids in $\mathbb{R}^n$. *Discrete & Computational Geometry*, 37(4):601–617, 2007. doi:10.1007/s00454-007-1309-8.

21 Frédéric Chazal and André Lieutier. Smooth manifold reconstruction from noisy and non-uniform approximation with guarantees. *Computational Geometry*, 40(2):156–170, 2008. doi:10.1016/j.comgeo.2007.07.001.

22 Marc Culler, Nathan M. Dunfield, Matthias Goerner, and Jeffrey R. Weeks. SnapPy: A computer program for studying the geometry and topology of 3-manifolds, 2021. URL: http://snappy.computop.org.

23 X. Dousson, J. Rubio, F. Sergeraert, and Y. Siret. *The Kenzo program*. Institut Fourier, Grenoble, 1999. URL: http://www-fourier.ujf-grenoble.fr/~sergerar/Kenzo/.

24 Nathan M. Dunfield. A census of exceptional Dehn fillings. In *Characters in Low-Dimensional Topology*, volume 760 of *Contemporary Mathematics*, pages 143–155. American Mathematical Society, 2020. doi:10.1090/conm/760/15289.

25 László Egri, Andrei Krokhin, Benoit Larose, and Pascal Tesson. The complexity of the list homomorphism problem for graphs. *Theory of Computing Systems*, 51(2):143–178, 2012. doi:10.1007/s00224-011-9333-8.

26 Graham Ellis. *HAP: Homological Algebra Programming, Version 1.70*, 2025. URL: https://gap-packages.github.io/hap.

27 Henrique Ennes and Clément Maria. Hardness of computation of quantum invariants on 3-manifolds with restricted topology. In *33rd Annual European Symposium on Algorithms (ESA 2025)*, volume 351 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 37:1–37:16, Dagstuhl, Germany, 2025. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.ESA.2025.37.

28 Henrique Ennes and Clément Maria. Compressed data structures for Heegaard splittings, 2025. arXiv:2507.11406.

29 Michael Farber. Topological complexity of motion planning. *Discrete & Computational Geometry*, 29(2):211–221, 2003. doi:10.1007/s00454-002-0760-9.

30 Brittany Terese Fasy, Rafal Komendarczyk, Sushovan Majhi, and Carola Wenk. On the reconstruction of geodesic subspaces of $\mathbb{R}^n$. *International Journal of Computational Geometry & Applications*, 32(01n02):91–117, 2022. doi:10.1142/S0218195922500066.

31 Tomás Feder, Pavol Hell, and Jing Huang. Bi-arc graphs and the complexity of list homomorphisms. *Journal of Graph Theory*, 42(1):61–80, 2003. doi:10.1002/jgt.10073.

32 Marek Filakovský and Lukáš Vokřínek. Are two given maps homotopic? An algorithmic viewpoint. *Foundations of Computational Mathematics*, 20(2):311–330, 2020. doi:10.1007/s10208-019-09419-x.

33 Leonidas J Guibas and Steve Y Oudot. Reconstruction using witness complexes. *Discrete & Computational Geometry*, 40(3):325–356, 2008. doi:10.1007/s00454-008-9094-6.

34 Allen Hatcher. *Algebraic Topology*. Cambridge University Press, 2002. URL: https://pi.math.cornell.edu/~hatcher/AT/AT.pdf.

35 Jean-Claude Hausmann. *On the Vietoris-Rips complexes and a Cohomology Theory for metric spaces*, page 175–188. Princeton University Press, December 1996. doi:10.1515/9781400882588-013.

36 Alexander He, James Morgan, and Em K. Thompson. An algorithm to construct one-vertex triangulations of Heegaard splittings. *Journal of Computational Geometry*, 16(1):635–693, Nov. 2025. doi:10.20382/jocg.v16i1a18.

37 William Jaco, Hyam Rubinstein, and Stephan Tillmann. Minimal triangulations for an infinite family of lens spaces. *Journal of Topology*, 2(1):157–180, 2009. doi:10.1112/jtopol/jtp004.

38 William Jaco and J. Hyam Rubinstein. Layered-triangulations of 3-manifolds, 2006. arXiv:math/0603601.

39 William Jaco, J Hyam Rubinstein, and Stephan Tillmann. Coverings and minimal triangulations of 3–manifolds. *Algebraic & Geometric Topology*, 11(3):1257–1265, 2011. doi:10.2140/agt.2011.11.1257.

40 Jisu Kim, Jaehyeok Shin, Frédéric Chazal, Alessandro Rinaldo, and Larry Wasserman. Homotopy Reconstruction via the Čech Complex and the Vietoris-Rips Complex. In *36th International Symposium on Computational Geometry (SoCG 2020)*, volume 164 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 54:1–54:19, Dagstuhl, Germany, 2020. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.SoCG.2020.54.

41 Janko Latschev. Vietoris-Rips complexes of metric spaces near a closed Riemannian manifold. *Archiv der Mathematik*, 77(6):522–528, 2001. doi:10.1007/PL00000526.

42 Frank H. Lutz. Combinatorial 3-manifolds with 10 vertices. *Beiträge Algebra Geom.*, 49(1):97–106, 2008. URL: https://ftp6.gwdg.de/pub/EMIS/journals/BAG/vol.49/no.1/5.html.

43 Sushovan Majhi. Demystifying Latschev's Theorem: Manifold Reconstruction from Noisy Data. In Wolfgang Mulzer and Jeff M. Phillips, editors, *40th International Symposium on Computational Geometry (SoCG 2024)*, volume 293 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 73:1–73:16, Dagstuhl, Germany, 2024. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.SoCG.2024.73.

44 Sushovan Majhi. Demystifying Latschev's theorem: manifold reconstruction from noisy data. *Discrete & Computational Geometry*, 74(2):544–568, 2025. doi:10.1007/s00454-024-00655-9.

45 Clément Maria, Jean-Daniel Boissonnat, Marc Glisse, and Mariette Yvinec. The GUDHI library: Simplicial complexes and persistent homology. In *International Congress on Mathematical Software*, pages 167–174. Springer, 2014. doi:10.1007/978-3-662-44199-2_28.

46 Partha Niyogi, Stephen Smale, and Shmuel Weinberger. Finding the homology of submanifolds with high confidence from random samples. *Discrete & Computational Geometry*, 39(1):419–441, 2008. doi:10.1007/s00454-008-9053-2.

47 Laurent Perron and Frédéric Didier. *CP-SAT*. Google, February 2025. Version 9.12. URL: https://developers.google.com/optimization/cp/cp_solver/.

48 Laurent Perron and Vincent Furnon. *OR-Tools*. Google, February 2025. Version 9.12. URL: https://developers.google.com/optimization/.

49 Vin de Silva and Gunnar Carlsson. Topological estimation using witness complexes. In Markus Gross, Hanspeter Pfister, Marc Alexa, and Szymon Rusinkiewicz, editors, *SPBG'04 Symposium on Point - Based Graphics 2004*. The Eurographics Association, 2004. doi:10.2312/SPBG/SPBG04/157-166.

50 Thom Sulanke and Frank H. Lutz. Isomorphism-free lexicographic enumeration of triangulated surfaces and 3-manifolds. *European J. Combin.*, 30(8):1965–1979, 2009. doi:10.1016/j.ejc.2008.12.016.

51 Lev Telyatnikov, Guillermo Bernardez, Marco Montagna, Mustafa Hajij, Martin Carrasco, Pavlo Vasylenko, Mathilde Papillon, Ghada Zamzmi, Michael T. Schaub, Jonas Verhellen, Pavel Snopov, Bertran Miquel-Oliver, Manel Gil-Sorribes, Alexis Molina, Victor Guallar, Theodore Long, Julian Suk, Patryk Rygiel, Alexander Nikitin, Giordan Escalona, Michael Banf, Dominik Filipiak, Max Schattauer, Liliya Imasheva, Alvaro Martinez, Halley Fritze, Marissa Masden, Valentina Sánchez, Manuel Lecha, Andrea Cavallo, Claudio Battiloro, Matt Piekenbrock, Mauricio Tec, George Dasoulas, Nina Miolane, Simone Scardapane, and Theodore Papamarkou. TopoBench: A framework for benchmarking topological deep learning, 2025. arXiv:2406.06642.

52 Julien Tierny, Guillaume Favelier, Joshua A. Levine, Charles Gueunet, and Michael Michaux. The topology toolkit. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):832–842, 2017. doi:10.1109/TVCG.2017.2743938.

Which Vertical Graphs are Non VPHT Reconstructible?

Jette Gutzeit[1], Kalani Kistler[2], Tim Ophelders[3], and Anna Schenfisch[4]

1 **Department of Mathematics, University of Zurich**
`jette.gutzeit@uzh.ch`
2 **Department of Mathematics, ETH Zürich**
`kkistler@student.ethz.ch`
3 **Department of Mathematics and Computer Science, TU Eindhoven**
`t.a.e.ophelders@tue.nl`
4 **Department of Mathematics, KTH Royal Institute of Technology**
`schenf@kth.se`

Abstract

The verbose persistent homology transform (VPHT) is a topological summary of shapes in Euclidean space. Assuming general position, the VPHT is injective, meaning shapes can be reconstructed using only the VPHT. In this work, we investigate cases in which the VPHT is not injective, focusing on a simple setting of degeneracy; graphs whose vertices are all collinear. We identify both necessary properties and sufficient properties for non-reconstructibility of such graphs, bringing us closer to a complete classification.

Related Version https://arxiv.org/abs/2603.07809

1 Introduction

Persistence diagrams are a topological tool for data analysis. The persistent homology transform (PHT) is the collection of these diagrams corresponding to height filtrations of some shape in every possible direction. First introduced in [11], the PHT was shown to be injective for simplicial complexes embedded in general position in dimensions $d = 2, 3$, meaning that we can use it to reconstruct them up to subdivision. Injectivity was later shown for general d in various settings [7, 8, 1, 3, 5, 2, 9, 10, 12]. In this work, we take a rather opposite approach, and instead investigate which simplicial complexes are *not* reconstructible by homology transforms. In the following, we consider the more discriminative *verbose* variant of persistence diagrams, which give rise to the *verbose* PHT (VPHT), described in Section 2. While [8] identifies families of functions not reconstructible by the related Euler characteristic transform, we are not aware of other work specifically focused on non-reconstructibility of simplicial complexes using the (V)PHT. It is easy to find simplicial complexes that are non PHT-reconstructible, but it is less straightforward to do so for the more discriminative VPHT. Furthermore, as we already know the VPHT is injective for simplicial complexes whose vertices are in general position [5], we turn to a simple degenerate setting as our starting place, namely, *vertical graphs*, whose vertices are all collinear. Our main results identify families and characterizations of non VPHT-reconstructible vertical graphs. We additionally observe that our results inform the related question of choosing directions for more general application settings; directions that see vertices and edges of a general graph in the same order as a non-reconstructible vertical graph will not be able to effectively determine the edge set on their own.

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

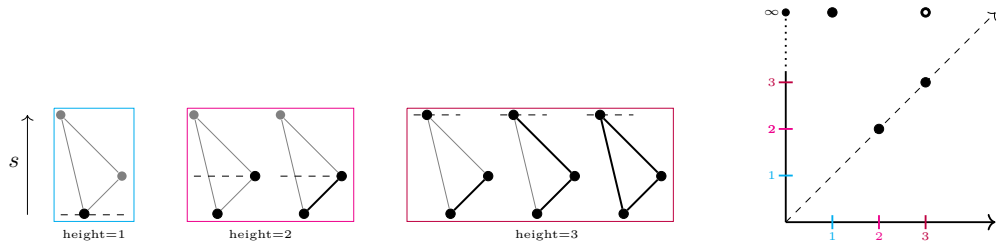


Figure 1 The lower-star filtration of this graph with respect to the vertical up direction s has a step for each vertex height, adding; (1) the bottom vertex (which creates a connected component), (2) the middle vertex and first edge (no effect on homology), and (3) the top vertex and remaining edges (creating a cycle). A corresponding index filtration is shown in the boxes, adding vertices and edges individually. This causes instantaneous births/deaths of components, which appear as on-diagonal points in the verbose diagram (right). The circle denotes the cycle born at height (3) and the black dots denote connected components.

We define vertical graphs and related concepts in Section 2. In Section 3, we state our theoretical results on the structures: omitted proofs can be found in the full version. Our computational results are discussed in Section 4. We close this paper with a discussion of future work in Section 5.

2 Preliminaries

We first discuss the VPHT and related concepts. We consider graphs $G = (V, E)$ in $\mathbb{R}^2$, but note that all definitions generalize to simplicial complexes in $\mathbb{R}^d$ (see, e.g., [5]). Given a unit vector s, the height of a vertex v with respect to s is the dot product $h_s(v) = s \cdot v$. We extend this to edges as $h_s(v, w) = \max\{h_s(v), h_s(w)\}$. Then, the *lower-star filtration of G with respect to s* is the nested inclusion of subgraphs of G with increasing maximum height.

Associated to every lower-star filtration are compatible total orderings of $V \cup E$, where, for $\sigma_i, \sigma_j \in V \cup E$, the relation $\sigma_i \prec \sigma_j$ implies $h_s(\sigma_i) \leq h_s(\sigma_j)$, i.e., total orders that respect the height relation. From such orders, we build *compatible index filtrations*, or nested inclusions of subgraphs of G with increasing maximum value in the total order. See Fig. 1.

Finally, we introduce *persistence diagrams*, multisets of points in $(\mathbb{R} \cup \{\infty\})^2$, where a point (a, b) represents the lifespan of a connected component or cycle. Each feature is *born* at its first appearance, a in the filtration. A connected component *dies* at time b when it merges with an older component. If this never occurs, $b = \infty$. Cycles always live to infinity. Persistence diagrams that arise from a lower-star filtration are examples of (standard) *concise diagrams*, and persistence diagrams that arise from compatible index filtrations are called *verbose diagrams*. See Fig. 1, right. Verbose diagrams differ from their concise counterparts only in that they have additional points on the diagonal. The verbose persistence diagram is well-defined, see, e.g., [5, Appendix B].

We utilize the following important observation about verbose persistence diagrams.

▶ **Observation 2.1.** *There is a one-to-one correspondence between points in the verbose diagram recording a connected component and vertices of the graph, with birth coordinates equal to the heights of vertices. That is, vertices are* creators. *Additionally, there is a one-to-one correspondence between points in the verbose diagram recording cycles and points recording connected components with finite lifespans, with cycle births and component deaths*

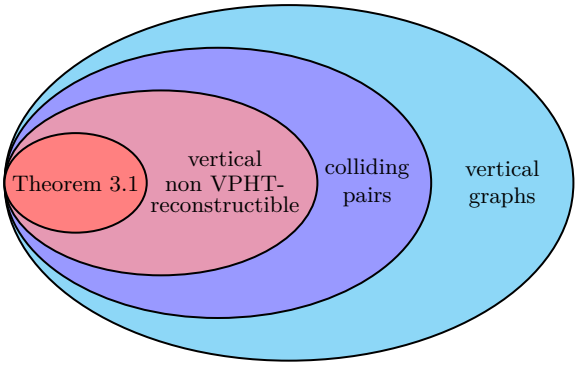


Figure 2 The above Venn diagram shows the classification of different graphs that will be discussed in the following pages. All inclusions are strict, where Lemma 3.4 distinguishes the pink class from the red and Fig. 5 distinguishes the purple class from the pink.

equal to heights of edges. That is, edges are either creators of cycles or destructors *of connected components.*

We now define the verbose persistent homology transform, specific to graphs:

▶ **Definition 2.2** (Verbose Persistent Homology Transform)**.** Given a graph G with vertices in $\mathbb{R}^2$, the verbose persistent homology transform (VPHT) of G, written $VPHT(G)$, is the set of verbose diagrams corresponding to filtrations in all directions, parameterized by direction.

In subsequent sections, we discuss graphs with multi-edges, which are themselves not simplicial complexes. We also discuss directed graphs. However, we use multi- and directed edges only as a tool to understand pairs of simple undirected graphs through their disjoint unions, and we only consider the VPHT of undirected simple graphs.

Finally, we come to reconstructibility.

▶ **Definition 2.3** (VPHT-Reconstructible)**.** We say a graph G_1 with an embedded vertex set is *VPHT reconstructible* if $VPHT(G_1) = VPHT(G_2)$ implies $G_1 = G_2$. That is, the VPHT of G_1 is unique, and could be used to reconstruct G_1. Otherwise, G_1 is *non VPHT-reconstructible*.

Graphs in general position are always VPHT-reconstructible [1, Secs 4 and 5]. We therefore focus on the simplest graphs breaking general position, graphs whose vertices are collinear and, without loss of generality, aligned with the y-axis.

▶ **Definition 2.4** (Vertical Graph)**.** A *vertical graph* G is a graph whose vertices $(v_1, \ldots, v_n)$ are distinct points on a vertical line, such that $i < j$ implies that v_j is above v_i, which we may also write as $v_i < v_j$. We write the height of v_i in the up direction as i.

Note that in this setting, every direction encounters the vertices in one of three orders; bottom-up, top-down, or simultaneously. The diagrams corresponding to directions that see the vertices in one of these orders are related. Directions that see vertices in the same total order produce nearly identical persistence diagrams, differing only by the exact heights of events, but *not* in the order or type of events. Diagrams for the two directions that see vertices simultaneously shift the order of events to the extreme, with every event born at the same height and dying either instantaneously or at infinity. These observations are more succinctly stated below, and allow us to greatly simplify our proofs.

▶ **Observation 2.5.** *Let G be a vertical graph. Then we can obtain $VPHT(G)$ from the verbose diagrams corresponding to only two directions, e.g., up and down along the vertical axis, as they can be used to obtain the verbose diagrams for any other direction.*

To understand relationships between pairs of graphs G_1 and G_2, we often use their *disjoint union*, $G_1 \sqcup G_2$, by which we mean the ordinary union of their vertices and disjoint union of their edges (possibly leading to multi-edges). We utilize this idea in the following definition.

▶ **Definition 2.6** (Simple Colliding Pair)**.** Vertical graphs G_1 and G_2 form a *simple colliding pair* if orienting all edges of G_1 up and all edges of G_2 down creates a cycle in $G = G_1 \sqcup G_2$ that alternates between up and down edges, i.e., creates an *alternating cycle*. See Fig. 3.

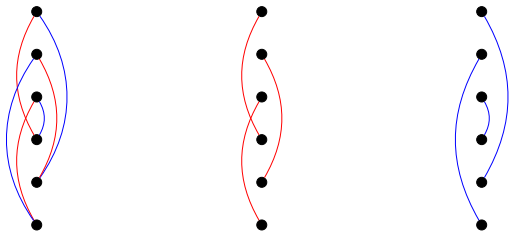


Figure 3 The disjoint union (left) of a simple colliding pair (middle and right). We observe the alternating cycle if we consider all red edges to be oriented up (without loss of generality) and all blue edges to be oriented down.

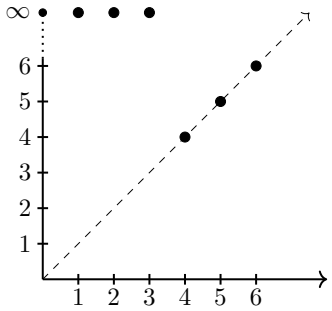


Figure 4 Verbose diagram for the red and blue graphs of Fig. 3 when filtering in the up direction.

▶ **Definition 2.7.** A graph of *type* $\mathcal{G}$ is a vertical graph, where the edges can be partitioned into alternating cycles $G^1, \ldots, G^k$. Furthermore, each G^i can each be partitioned into a simple colliding pair (G_1^i, G_2^i). The two subgraphs on the edges $\bigsqcup_{i\in[k]} G_1^i$, resp. $\bigsqcup_{i\in[k]} G_2^i$, are called graphs of *type* $\mathcal{G}_1$, resp. $\mathcal{G}_2$. The pair $(\mathcal{G}_1, \mathcal{G}_2)$ is a *colliding pair*.

Note that the choice of cycle partitioning $\mathcal{G}$ into some $\mathcal{G}_1$ and $\mathcal{G}_2$ is not necessarily unique.

▶ **Definition 2.8.** For a vertical graph $G = (V, E)$, a fixed up direction, and vertex $v \in V$, we define the number of *down-edges of* v as $\text{ldeg}_G(v) := |\{w \in V \mid (v, w) \in E, w < v\}|$. Similarly, the number of *up-edges of* v is $\text{hdeg}_G(v) := |\{w \in V \mid (v, w) \in E, v < w\}|$.

3 Theoretical Results for Vertical Graphs

Next, we classify families of vertical graphs that are non VPHT-reconstructible, summarized in Fig. 2. Omitted proofs can be found in our full version. We first observe that some colliding

pairs are VPHT-reconstructible, e.g., the graphs in Fig. 5. In addition to having distinct diagrams in the up direction (Fig. 6), we computationally checked that no other graph has the same VPHT. The following gives a family of colliding pairs that are *not* reconstructible.

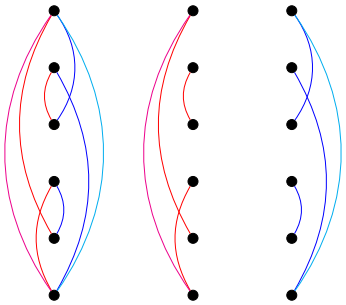


Figure 5 On the left $\mathcal{G}$, made up of two vertical graphs with alternating cycles; one includes all red and blue edges, the other the cyan and magenta edge. On the right the corresponding $\mathcal{G}_1$ and $\mathcal{G}_2$.

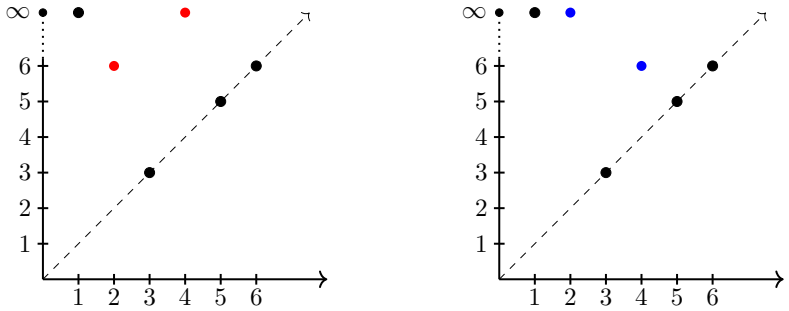


Figure 6 Verbose persistence diagrams corresponding to $\mathcal{G}_1$ and $\mathcal{G}_2$ of Fig. 5 and the up direction; black dots appear in both diagrams, red only in $\mathcal{G}_1$, blue ones only in $\mathcal{G}_2$.

▶ **Theorem 3.1** (Special Colliding Pairs are Non Reconstructible). *Consider a type $\mathcal{G}$ graph, such that each vertex has no more than two lower neighbors and no more than two upper neighbors. Then, for any choice of alternating cycles $C^1, \ldots, C^k$ partitioning G, the resulting graphs $\mathcal{G}_1$ and $\mathcal{G}_2$ are non VPHT-reconstructible.*

Such a colliding pair can be seen in Fig. 3 and an associated verbose diagram in Fig. 4.

Proof sketch. We first consider the verbose diagrams for $\mathcal{G}_1$ and $\mathcal{G}_2$ in the up direction. We can show that each vertex v has at most $\text{ldeg}_{\mathcal{G}_i}(v) = 1$ for $i = 1, 2$, so both verbose diagrams only have non-trivial zeroth homology. Further, notice that no connected component can die at a point different to its birth height or infinity, since otherwise two non-trivial connected components must be joined at some vertex v, but then $\text{ldeg}_{\mathcal{G}_1}(v) \geq 2$, a contradiction. If the connected component created by a vertex is killed instantly, that vertex is the higher vertex of some edge. If it lives forever, it is not. Because $(\mathcal{G}_1, \mathcal{G}_2)$ forms a colliding pair, vertices and their down degrees are equivalent in $\mathcal{G}_1$ and $\mathcal{G}_2$, and thus the verbose diagrams are also equivalent. By a symmetric argument in the down direction, we find $\text{VPHT}(\mathcal{G}_1)$=$\text{VPHT}(\mathcal{G}_2)$. ◀

Next, we identify properties of non-reconstructible graphs. In order to prove Theorem 3.3, we first establish the following fact about degrees of vertices in type $\mathcal{G}$ graphs.

▶ **Lemma 3.2** (Type $\mathcal{G}$ Equivalence). *A vertical graph $G = (V, E)$ is of type $\mathcal{G}$ if and only if for all $v \in V$, $\mathrm{ldeg}_G(v)$ and $\mathrm{hdeg}_G(v)$ are even.*

This allows us to state the following theorem about non-reconstructibility.

▶ **Theorem 3.3** (Non-Reconstructible implies Colliding Pair). *If a pair of vertical graphs (G_1, G_2) is not a colliding pair, then $VPHT(G_1)$ and $VPHT(G_2)$ are distinct.*

Proof. We consider the verbose diagrams corresponding to the up direction. If G_1 and G_2 are not a colliding pair, then, by Lemma 3.2, there exists some vertex v_i that (without loss of generality) $\mathrm{ldeg}(v_i)$ is odd and $\mathrm{ldeg}_{G_1}(v_i) > \mathrm{ldeg}_{G_2}(v_i)$.

By Observation 2.1, edges incident to and below v_i in G_1 are in bijection with points in the verbose diagram marking zero-dimensional deaths at height i and one-dimensional births at height i. Then, because the number of such edges in G_1 and G_2 differ, the number of such points in the diagrams for G_1 and G_2 must also differ. Thus, $VPHT(G_1) \neq VPHT(G_2)$. ◀

Finally, given a colliding pair, we can extend it so that it remains (non-) reconstructible.

▶ **Lemma 3.4.** *Let $(\mathcal{G}_1, \mathcal{G}_2)$ be a colliding pair. Let $\mathcal{C} = \{C^1, \ldots, C^k\}$ be a partition of G into alternating cycles. Then adding another copy of a cycle $C \in \mathcal{C}$ to $\mathcal{G}$ does not change the non VPHT-reconstructibility of the resulting colliding pair.*

Proof. We investigate how adding a cycle copy impacts the verbose diagrams with respect to the up direction. Adding a copy of an existing cycle cannot connect previously disconnected components, meaning there is no change to the zero-dimensional homology. We obtain a new cycle for every new double edge, born at the height of the higher vertex of the edge. But since we added an alternating cycle, a vertex v is the higher vertex for an edge in $\mathcal{G}_1 \sqcup C$ if and only if it is the higher vertex for an edge in $\mathcal{G}_2 \sqcup C$, since that means we have a sequence v_i, v, v_j in the cycle, with $v_i, v_j < v$, thus the edge (v_i, v) must be in $\mathcal{G}_1 \sqcup C$ and (v, v_j) in $\mathcal{G}_2 \sqcup C$. As this argument applies to both copies of the added cycle, double edges with upper vertex v appear in both of the new graphs in the new pair (obtained after partitioning the new G with added cycle C), thus the same one-cycles are born at the same times, and thus the changes to both verbose diagrams are identical. An identical argument holds for the down direction. Thus, if the VPHTs were identical (or distinct), they remain identical (or distinct), maintaining the (non-) reconstructibility of the resulting graphs. ◀

4 Computational Approach

Our theoretical part is complemented by a publicly accessible[1] interactive computational tool we developed. Its functions are described in the associated repository. In addition to building intuition, we used it to confirm all results above for all simple vertical graphs on up to seven vertices. Furthermore, we observed the following:

▶ **Lemma 4.1** (Independence up to 7 Vertices). *All pairs of vertical graphs with up to seven vertices that share a VPHT form colliding pairs, and there exists an alternating cycle partition in which all edges common to both graphs lie on cycles of length two.*

That is, removing all common edges, we still obtain a decomposition into alternating cycles. We conjecture that an analogue to Lemma 4.1 holds for an arbitrary number of vertices.

[1] Web App: `https://kalokak.github.io/vpht-algo/`

5 Discussion and Conclusion

We presented a partial classification of non VPHT-reconstructible vertical graphs (Fig. 2). We are still interested in a complete classification in terms of easily checkable conditions.

In our work, we only consider vertical graphs. For general graphs, one can reconstruct any edge whose supporting line contains no vertices other than its two endpoints [1]. We conjecture that if a particular edge of a graph G is not reconstructible, then the set of edges of G with the same supporting line themselves form a (rotated) vertical graph that is non VPHT-reconstructible. If true, this conjecture implies that any non VPHT-reconstructible graph decomposes into reconstructible edges, and non VPHT-reconstructible vertical graphs.

While theoretical work on reconstructibility often involves carefully chosen directions, current applications of directional transforms typically use random or evenly-spaced directions, the effect of which was explored in [4]. Our results give insight to directions that are potentially unhelpful to choose in such settings. Namely, if projecting a graph onto some one-dimensional subspace produces one of our non VPHT-reconstructible vertical graphs, we know a direction orthogonal to such a subspace is not able to uniquely reconstruct the edges. For instance, this is exactly the case for the graphs in [6, Figure 8], where the authors identify all directions in which the graphs project to a version of Figure 3.

References

1 Robin Lynne Belton, Brittany Terese Fasy, Rostik Mertz, Samuel Micka, David L. Millman, Daniel Salinas, Anna Schenfisch, Jordan Schupbach, and Lucia Williams. Reconstructing embedded graphs from persistence diagrams. *Computational Geometry*, 90:101658, October 2020. URL: `https://www.sciencedirect.com/science/article/pii/S0925772120300523`, `doi:10.1016/j.comgeo.2020.101658`.

2 Leo M Betthauser. *Topological reconstruction of grayscale images*. PhD thesis, University of Florida, 2018.

3 Justin Curry, Sayan Mukherjee, and Katharine Turner. How many directions determine a shape and other sufficiency results for two topological transforms. *Transactions of the American Mathematical Society, Series B*, 9(32):1006–1043, 2022.

4 Brittany Terese Fasy, Maksym Makarchuk, Samuel Micka, and David L. Millman. Too many or too few? sampling bounds for topological descriptors, 2025. URL: `https://arxiv.org/abs/2511.12059`, `arXiv:2511.12059`.

5 Brittany Terese Fasy, Samuel Micka, David L. Millman, Anna Schenfisch, and Lucia Williams. A Faithful Discretization of Verbose Directional Transforms. *Discrete & Computational Geometry*, November 2025. `doi:10.1007/s00454-025-00791-w`.

6 Brittany Terese Fasy, David L Millman, and Anna Schenfisch. How small can faithful sets be? ordering topological descriptors. In *Canadian Conference on Computational Geometry*, 2024.

7 Robert Ghrist, Rachel Levanger, and Huy Mai. Persistent homology and euler integral transforms. *Journal of Applied and Computational Topology*, 2(1):55–60, 2018.

8 Mattie Ji. On the injectivity of Euler integral transforms with hyperplanes and quadric hypersurfaces. *Mathematics*, 12(15):2339, 2024.

9 Samuel Adam Micka. *Searching and reconstruction: algorithms with topological descriptors*. Montana State University, 2020.

10 Adam Onus, Nina Otter, and Renata Turkes. Shoving tubes through shapes gives a sufficient and efficient shape statistic. *arXiv preprint arXiv:2412.18452*, 2024.

11 Katharine Turner, Sayan Mukherjee, and Doug M Boyer. Persistent homology transform for modeling shapes and surfaces. *Information and Inference: A Journal of the IMA*, 3(4):310–344, 2014.

12 Katharine Turner, Vanessa Robins, and James Morgan. The extended persistent homology transform of manifolds with boundary. *Journal of Applied and Computational Topology*, 8(7):2111–2154, 2024.

Topologically Stable Hough Transform*

Stefan Huber[1], Kristóf Huszár[2], Michael Kerber[2], and Martin Uray[1]

1 **Josef Ressel Centre for Intelligent and Secure Industrial Automation, Salzburg University of Applied Sciences, Austria**
{stefan.huber,martin.uray}@fh-salzburg.ac.at
2 **Institute of Geometry, Graz University of Technology, Austria**
{kristof.huszar,kerber}@tugraz.at

Abstract

We propose an alternative formulation of the well-known Hough transform to detect lines in point clouds. Replacing the discretized voting scheme of the classical Hough transform by a continuous score function, its persistent features in the sense of persistent homology give a set of candidate lines. We also devise and implement an algorithm to efficiently compute these candidate lines.

Related Version *Full Version:* https://arxiv.org/abs/2603.08245 [8]

1 Introduction

The *Hough transform* is a classical method in Computer Vision to detect lines (or more complicated geometric shapes) in partially-sampled and noisy sceneries, dating back to the 1960s [9]. Informally, lines are detected by a voting scheme where every sample point p votes for lines that contain p or are close-by, and those lines that accumulate many votes are considered to be part of underlying scenery. To specify the meaning of "close-by," the space of lines is parameterized by $\mathbb{R}^2$ and discretized into pixels. A sample point transforms into a planar curve in that line space and votes for those lines whose pixel it traverses, see Figure 1.

The above voting scheme suffers from two major problems: first, noise in the sample can lead to a collection of neighboring pixels that all get large vote counts. If the selection rule is to chose the k lines with largest votes, this may result in a collection of lines that are very close to each other, which might be not the desired outcome. Second, the voting scheme based on the binary decision whether a pixel is traversed or not is unstable: even translating the grid (i.e., choosing a different origin) may result in drastically different outcomes.

Contributions. We propose a simple and efficient framework that addresses both of the shortcomings. First, instead of voting on pixels, the sample points continuously vote on all lines in line space by scoring each of them. The score is determined by the Euclidean distance of the point and the line and some kernel function, e.g., a linear or a Gaussian curve. This way, every candidate line receives a score from the sample points, yielding a total score for any candidate line. The resulting *score function* is continuous over the parameter space of lines, and is stable under perturbations of the point sample, meaning that small changes in the sample will lead to controlled point-wise changes in the score function. The authors of [10] also follow a continuous treatment, but for a hypothesis testing framework.

The second idea concerns the selection of the most important lines based on the score function. We suggest a natural approach to select the lines corresponding to the local

* The financial support by the Austrian Federal Ministry of Economy, Energy and Tourism, the National Foundation for Research, Technology and Development and the Christian Doppler Research Association is gratefully acknowledged.

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

maxima of the score function as they received the highest score in a neighborhood. To sort these (potentially many) local maxima by importance, we pick those maxima with the largest *persistence* in the sense of 0-homology for the super-levelset filtration of the score function [6].[1] Using persistence prevents that two close-by lines are both selected, except when they are separated by a significant valley in the score function. This addresses the problem of selecting two many similar lines. A similar approach has been already applied to clustering [3]. While the number of local maxima is not stable under perturbations, the number of *persistent* (or prominent) local maxima turns out to be stable, and their locations also remain stable under certain conditions, see Theorem 3.2.

We also describe an efficient algorithm that approximates the score function via a quadtree subdivision. It guarantees that each local maximum with significant persistence has a representative line in the output, and that no line with insignificant persistence appears.

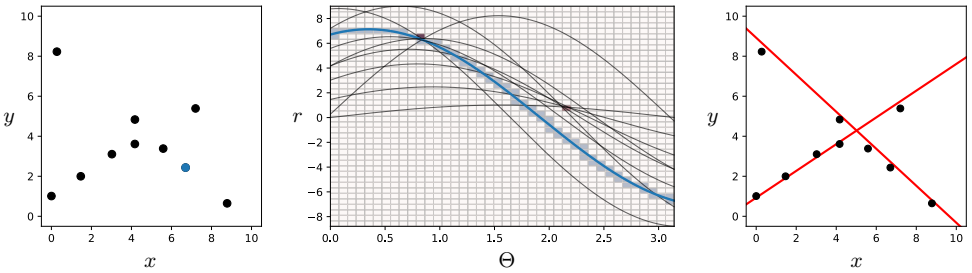


Figure 1 Left: Ten points sampled from two lines. Middle: The classical Hough transform passes to the space of lines in which every input point dualizes into a sinusoidal curve. The blue point on the left, for instance, is transformed to the blue curve in the middle. The pixels that the blue curve votes for are marked. In that example, there are two pixels (red) that receive five votes. Right: The centers of the high-count pixels are two lines in primal space that correspond to the two drawn lines.

2 The score function

Line parameterization. Set $M := \mathbb{R} \times [0, \pi]$. For $(r, \Theta) \in M$, we define

$$\ell(r, \Theta) := \{(x, y) \in \mathbb{R}^2 \mid r = x \cos \Theta + y \sin \Theta\}.$$

For $p = (r \cos \Theta, r \sin \Theta)$, $\ell(r, \Theta)$ is the line through p that is perpendicular to the vector $\overrightarrow{p}$. Note that the set of all lines in $\mathbb{R}^2$ is in bijection with M, except that $\ell(r, 0) = \ell(-r, \pi)$, as the space of lines is homeomorphic to the open Möbius strip. For simplicity, we will first work with M as a planar domain and discuss the twisted glueing to a Möbius strip later.

Score of a line. For a point p and a line ℓ in the Euclidean plane, write $\Delta(p, \ell)$ for the orthogonal distance of p from ℓ. We want that the score of ℓ by p equals 1, if $\Delta(p, \ell) = 0$, and otherwise it decreases as $\Delta(p, \ell)$ grows. For that, we choose a monotonously decreasing kernel function $\kappa : [0, \infty) \to [0, 1]$ with $\kappa(0) = 1$ and $\kappa(x) \to 0$ for $x \to \infty$. Two examples are

$$\kappa_{\text{hat}}(x) := \max\left\{0, 1 - \frac{x}{\sigma}\right\} \qquad \text{and} \qquad \kappa_{\text{RBF}}(x) := \exp\left(-\frac{x^2}{2\sigma^2}\right),$$

[1] This idea of using persistence has already been proposed in a preliminary work [7]. However, it lacked the crucial idea of a continuous treatment via smooth voting functions, but relied on the original discretized setting, preventing a stability result as we now obtained in this work.

referred to as *hat* and *Gauss* kernel, respectively. Now, for a finite set $P \subset \mathbb{R}^2$ we define

$$\mathrm{S}(p,\ell) := \kappa(\Delta(p,\ell)) \qquad \text{and} \qquad \mathrm{S}(\ell) := \mathrm{S}_P(\ell) := \frac{1}{|P|}\sum_{p\in P} \mathrm{S}(p,\ell).$$

We refer to $\mathrm{S}\colon M \to \mathbb{R}, (r,\Theta) \mapsto \mathrm{S}(\ell(r,\Theta))$ as the *score function*. Clearly, $0 \le \mathrm{S}(\ell) \le 1$. If $\kappa(x) = 1$ only for $x = 0$, then $\mathrm{S}(\ell) = 1$ implies that ℓ contains all points of P.

The lines $\ell(r,\Theta)$ containing a fixed $p = (x_0, y_0)$ are given by $r = x_0 \cos\Theta + y_0 \sin\Theta$, whose solutions (r,Θ) form a sinusoidal curve γ, along which $\mathrm{S}(p,\ell(r,\Theta)) = 1$. Moreover, fixing Θ and varying r, the distance of p to the line $\ell(r,\Theta)$ changes linearly with r. Hence, the graph of $\mathrm{S}(p,\cdot)$ is obtained by sliding the function graph of the kernel function κ along γ. The graph of the score function S is obtained by a normalized sum of $|P|$ such graphs.

Stability. It is evident by construction that, if the kernel κ is continuous/smooth, then so is the score function S. Moreover, the fact that the Euclidean distance function to a fixed line is 1-Lipschitz immediately yields the stability of the score function.

▶ **Lemma 2.1.** *Let κ be Lipschitz-continuous with constant λ. Let P' denote an ϵ-perturbation of P (i.e., every point is replaced by a point in its ϵ-ball). Then $\|\mathrm{S}_P - \mathrm{S}_{P'}\|_\infty \le \lambda\epsilon$.*

3 Persistence-based selection

For fixed $h_0 \ge 0$, let $\mathrm{S}^{\ge h_0} := \{\ell \in M \mid \mathrm{S}(\ell) \ge h_0\}$ denote the *super-levelset* of S. By varying h_0 from $+\infty$ to 0, we "sweep out" the graph of S and obtain the super-levelset filtration. We consider the 0-*dimensional persistent homology* of that filtration and sort the local maxima of S according to the persistence of their homology class. For readers not familiar with persistent homology, we offer an elementary explanation. Assume for simplicity that all local maxima of S are isolated and no two maxima have the same height. Fix such a local maximum x with height h. Call x *dominant* in $\mathrm{S}^{\ge h'}$ if the connected component of $\mathrm{S}^{\ge h'}$ that contains x has x as its highest point. Now, the persistence of x is the supremum over all $t \ge 0$ such that x is dominant in $\mathrm{S}^{\ge h-t}$. We call h the *birth* level and $h - t$ the *death* level of this component. Lemma 3.1 is a reformulation of the stability result of persistence [4].

▶ **Lemma 3.1.** *Let $\mathrm{S}_P, \mathrm{S}_{P'}$ be two score functions over M with $\|\mathrm{S}_P - \mathrm{S}_{P'}\|_\infty = \epsilon$. Then, there exists a partial matching of local maxima of S_P to local maxima of $\mathrm{S}_{P'}$ such that all maxima of either score function with persistence greater than ϵ are matched, and the persistence of two matched local maxima differs by at most 2ϵ.*

▶ **Theorem 3.2.** *Let S_P be a score function with local maximum $x = (r,\Theta)$ and $h = \mathrm{S}_P(x)$ of persistence α. Let $\mathrm{S}_{P'}$ be a score function with $\|\mathrm{S}_P - \mathrm{S}_{P'}\|_\infty = \epsilon < \alpha/2$. Then, $\mathrm{S}_{P'}$ contains a local maximum x' of persistence at least $\alpha - 2\epsilon$ such that $|\mathrm{S}_P(x') - h| \le 2\epsilon$ and x' lies in the connected component of x of the set $\mathrm{S}_P^{\ge h-\alpha}$.*

Proof. By definition, x is dominant in a connected component C of $\mathrm{S}_P^{\ge h-\alpha}$. Note that $\mathrm{S}_{P'}^{\ge h-\alpha+\epsilon} \subseteq \mathrm{S}_P^{\ge h-\alpha}$, so $C \cap \mathrm{S}_{P'}^{\ge h-\alpha+\epsilon}$ is a connected component of $\mathrm{S}_{P'}^{\ge h-\alpha+\epsilon}$ (it is non-empty because of x). Its maximum x' is easily seen to satisfy the three postulated properties. ◀

Theorem 3.2 tells us that for $\alpha \gg \epsilon$, the persistent local maxima of $\mathrm{S}_{P'}$ approximate those of S_P in the sense that for every persistent local maximum of S_P, we can find a persistent local maximum of $\mathrm{S}_{P'}$ with almost the same score and being in the same super-levelset.

4 Computation

We devise a simple approximation scheme of S_P based on a quad-tree subdivision that yields a $\tilde{S}_P$ with $\|S_P - \tilde{S}_P\|_\infty \leq \epsilon$ for a fixed $\epsilon > 0$ and is constant on each quad-tree cell.[2] We first determine some $r_0 \in \mathbb{R}$ such that $S_P \leq \epsilon$ on $M \setminus [-r_0, r_0] \times [0, \pi]$, and set $\tilde{S}_P$ zero there.

The main predicate for the subdivision part is a Lipschitz predicate that, given a box $B := [a, b] \times [c, d] \subset M$, yields a $\lambda \in \mathbb{R}$ such that S_P, restricted to B, is continuous with Lipschitz constant λ. Writing h for the value of S_P at the midpoint of B, it follows that the S_P at any point in B differs from h by at most $\lambda \cdot \operatorname{diam}(B)/2$, where $\operatorname{diam}(B)$ denotes the diameter of B. With that predicate, we simply subdivide the initial box $[-r_0, r_0] \times [0, \pi]$ of the quad-tree and apply it until a subdivided box satisfies $\lambda \cdot \operatorname{diam}(B)/2 \leq \epsilon$. In that case, we set $\tilde{S}_P$ within that box B to be the value of S_P at the midpoint of B.

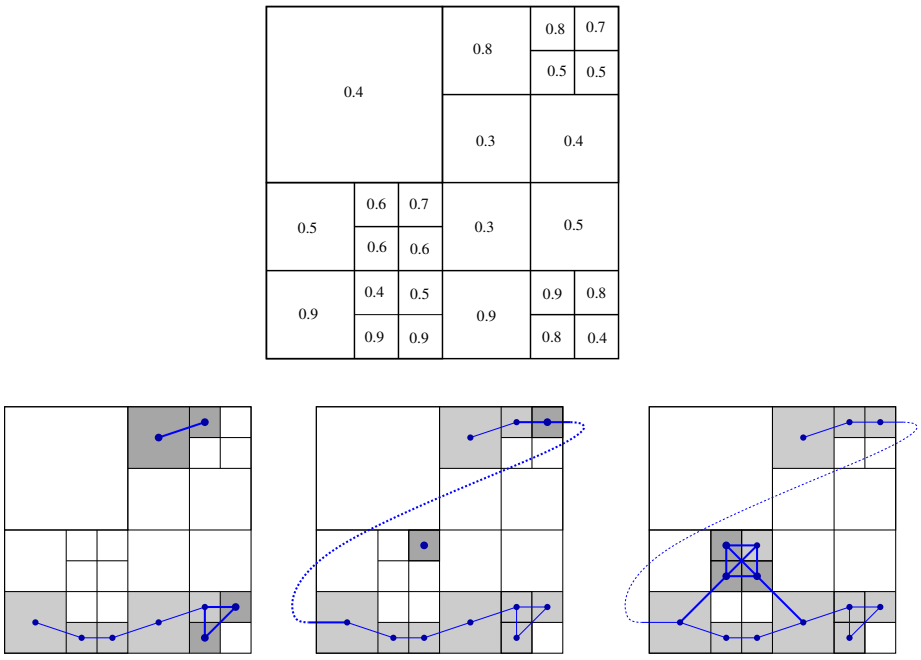


Figure 2 Top: the subdivided $\tilde{S}_P$. Bottom: The super-levelsets for threshold 0.8, 0.7, and 0.6, and the nerve of the boxes (the graph structure in blue). Vertices and edges that enter the graph at this threshold are drawn thicker. We observe that for 0.8, a new connected component is formed in the upper region. For 0.7, another isolated vertex is formed; moreover, the upper-right box connects with the lower-left box because of the Möbius geometry, hence, the component formed at 0.8 merges with another one, yielding a persistence of 0.1 for the component. At 0.6, the component formed at 0.7 also gets merged, yielding a persistence of 0.1 as well. Note that the graph is connected for 0.6.

How to realize such a Lipschitz predicate? Elementary calculus reveals that if the kernel function κ has Lipschitz constant λ_κ and all input points are in the unit disk, S_P has a global Lipschitz constant of $\lambda_\kappa\sqrt{2}$. However, using this global bound will yield a uniform

[2] The non-continuity of $\tilde{S}_P$ does not cause a problem because $\tilde{S}_P$ can be converted to a continuous score function with the same persistent homology. Without going into further detail, one triangulates each quad with a central Steiner vertex, considers a barycentric refinement and then constructs a PL-function. We omit the details of this step and refer to [6, sec. VI.3].

(and inefficient) subdivision. We obtain a better Lipschitz constant based on the locality of the box. We refer to the full version [8] for more details.

Computing the persistent homology (in dimension 0) of $\tilde{S}_P$ is done via standard methods illustrated in Figure 2. Via the Nerve Theorem [1], the problem can be recast into tracking the connected components of a graph that expands in terms of vertices and edges, and that problem can be solved in almost linear time via the union-find data structure [5]. Note that in this step, we remember the twisted identification of the boundary of M (the domain of $\tilde{S}_P$), effectively computing the persistent homology over the Möbius strip.

There is a choice how to ultimately select the local maxima , and hence the lines in primal space, from the persistence information. A simple choice is to just select the k local maxima with largest persistence, where k is a given constant. Another option is to fix a threshold α and pick those local maxima with persistence at least α. Here, α may be chosen a-priori or a-posteriori – we demonstrate the latter in Section 5.

5 A Use-case

We implemented a prototype of our method in Python, available at `https://github.com/JRC-ISIA/TopologicalHoughTransformation`. Figure 3 shows an exemplary run of our method on a point cloud sampled with noise from three lines (top-left).

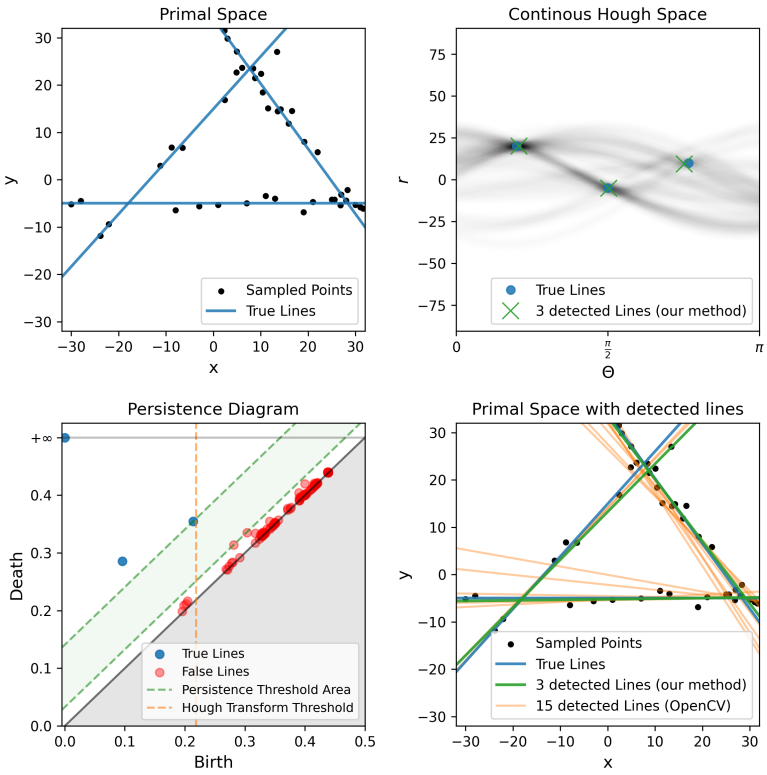


Figure 3 Illustration of the proposed method.

Notably, each line is sampled with a different number of points, resulting in different heights for the three distinguished local maxima in the score surface. Nevertheless, the score

function has three distinguished "bumps" (top-right) and the persistence homology reveals this fact: the *persistence diagram* (bottom-left) represents every local maximum by its height (birth) and the threshold when it ceases to be dominant (death); its persistence is then the distance to the diagonal. With this interpretation, we observe three clearly distinguished local maxima, and selecting them yields three lines that closely approximate the ground truth (bottom-right).

In comparison, we display the outcome of the Hough transform as computed by the `OpenCV` package [2]. In there, the specified threshold filters local maxima by height, not by persistence. Consequently, depending on how we choose the threshold, we either miss one of the ground truth lines altogether, or the method returns many local maxima (of low persistence) that all approximate the line with the densest sampling because it creates a larger bump in dual space. It would then require further post-processing to remove these duplicates while this filtering step is elegantly integrated in our framework already.

We ran more tests on instances similar to the one in Figure 3 and did some statistical analysis on the detected lines, showing that the above situation is not cherry-picked but seems typical if one deals with lines of different densities. See the full version [8] for details.

6 Discussion

We have demonstrated that simple tools from computational geometry and topology have the potential to increase the quality of line detection via the Hough transform. Although the experiments presented are limited to line detection, the proposed method generalizes to arbitrary shapes by varying the shape parametrization and the corresponding parameter space. We are planning more extensive tests, including the use of different kernel functions, evaluation on the effect of the kernel size, real image data, and the comparison with state-of-art line detection methods. A preliminary evaluation can be found in the full version [8]. As a prerequisite, we will first implement a fast version of our prototype – we are optimistic that the combination of adaptive subdivisions and the fact that persistence in dimension 0 can be computed in almost linear time will allow us to apply our method to large image data sets.

References

1 Ulrich Bauer, Michael Kerber, Fabian Roll, and Alexander Rolle. A unified view on the functorial nerve theorem and its variations. *Expositiones Mathematicae*, 41(4), 2023. doi:10.1016/j.exmath.2023.04.005.

2 Gary Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

3 Frédéric Chazal, Leonidas J. Guibas, Steve Y. Oudot, and Primoz Skraba. Persistence-based clustering in Riemannian manifolds. *Journal of the ACM*, 60(6):1–38, November 2013. doi:10.1145/2535927.

4 David Cohen-Steiner, Herbert Edelsbrunner, and John Harer. Stability of persistence diagrams. *Discrete & Computational Geometry*, 37:103–120, 2007. doi:10.1007/s00454-006-1276-5.

5 Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms*. The MIT Press, 2nd edition, 2001.

6 Herbert Edelsbrunner and John Harer. *Computational Topology. An Introduction.* American Mathematical Society, 2010. URL: https://bookstore.ams.org/mbk-69.

7 Johannes Ferner, Stefan Huber, Saverio Messineo, Angel Pop, and Martin Uray. Persistence-based Hough Transform for Line Detection. In Peter Haber, Thomas J. Lampoltshammer, and Manfred Mayr, editors, *Data Science—Analytics and Applications*, Cham, 2026. Springer Nature Switzerland. (Accepted Paper). URL: http://arxiv.org/abs/2504.16114.

8 Stefan Huber, Kristóf Huszár, Michael Kerber, and Martin Uray. Topologically stable Hough transform, 2026. 11 pages, 7 figures. arXiv:2603.08245.

9 Priyanka Mukhopadhyay and Bidyut B. Chaudhuri. A survey of Hough Transform. *Pattern Recognition*, 48(3):993–1010, March 2015. doi:10.1016/j.patcog.2014.08.027.

10 John Princen, John Illingworth, and Josef Kittler. Hypothesis testing: A framework for analyzing and optimizing Hough transform performance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(4):329–341, 1994. doi:10.1109/ICCV.1990.139564.

A set of convex prototiles that allows only countably many different tilings

Aleksa Džuklevski

Department of Applied Mathematics, Faculty of Mathematics and Physics, Charles University, Czech Republic
aleksa@matfyz.cz

Abstract

We say that a tile is σ-morphic if it tiles the plane in exactly $\aleph_0$ many noncongruent ways, where two tilings are said to be congruent if there's an isometry mapping one to the other. It is an unsolved problem of whether a σ-morphic tile exists in the plane. In this note we present a construction of a set of convex tiles that is σ-morphic. The result is interesting since all the constructions of σ-morphic sets of tiles that arise in the literature make use of bumps and nicks, which necessarily make the tiles non-convex. We construct our set by cleverly dividing the tiles of the set of tiles discovered by Schmitt into convex tiles so that they behave in the same manner.

Related Version arXiv:2507.20867

1 Introduction

The theory of tilings is a relatively young research area in discrete geometry and there are many yet unsolved problems concerning various elementary notions regarding tiles. Formally, a tiling $\mathcal{T}$ of the Euclidean plane is a set of elements T_i, $i \in \mathbb{N}$, which are closed topological discs in $\mathbb{E}^2$ such that (i) $\bigcup_{i\in\mathbb{N}} T_i = \mathbb{E}^2$ and (ii) $\operatorname{int}(T_i) \cap \operatorname{int}(T_j) = \varnothing$ for all $i \neq j$. If all the elements of $\mathcal{T}$ are congruent to some tile T, we say that T *tiles the plane*, or that it *admits a tiling* of the plane. The set of distinct tiles that appear in a tiling is called the *protoset* and its elements the *prototiles*. Amongst the unsolved problems in tiling theory there are those that are more combinatorial in nature, and perhaps the best example is the problem of the so called *polymorphic* tiles.

We say that a tile is *m-morphic* if it tiles the plane in exactly m non-congruent ways, where two tilings are congruent if there is an isometry that maps one to the other [4]. In 1977, Grünbaum and Shephard asked [3] whether for each integer m there exists an m-morphic tile, and the problem is still open. The record holder is an 11-morphic polyomino discovered by Myers [6].

Grünbaum and Shephard then focused on tiles that tile the plane in infinitely many ways and proceeded to make a fascinating observation: each tile they found, which tiles the plane in *infinitely* many ways, did so in $\mathfrak{c}$ many ways! For instance, taking a regular tiling by unit squares, we can observe that we get $\mathfrak{c}$ many tilings by shifting one row for $\alpha \in [0, \frac{1}{2}]$ along the x-axis (each corresponding to a different α). Some tiles, however, exibit a richer combinatorial structure to their tilings, as is the case with the tile shown in Figure 1. Note that the blue outline is the same as the yellow one, allowing the orange tiles to be oriented in a different direction, without requiring any change in the rest of the tiling. This freedom of choice at every layer gives rise to $\mathfrak{c}$ many tilings. See the arXiv version of this paper for a more thorough discussion [2].

A very natural question was then asked, concerning the existence of a tile that tiles the plane in exactly $\aleph_0$ many distinct ways, which was dubbed σ-morphic, and whose existence is still an open problem.

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

One of the main ideas behind "forcing" the tiles to behave in the way one would like them to, and which stands behind a lot of constructions seen in the literature (the construction of σ-morphic protosets by Schmitt [7, 8] or of a σ-morphic tile with generalized matching rules by Bašić, Džuklevski and Slivková [1]) is to use bump and nicks on the tiles, which then force specific edges of tiles to meet in a tiling (cf. Figure 2). Thus the problem of finding a protoset of convex prototiles that is σ-morphic is an interesting one and in this paper we show how it can be constructed.

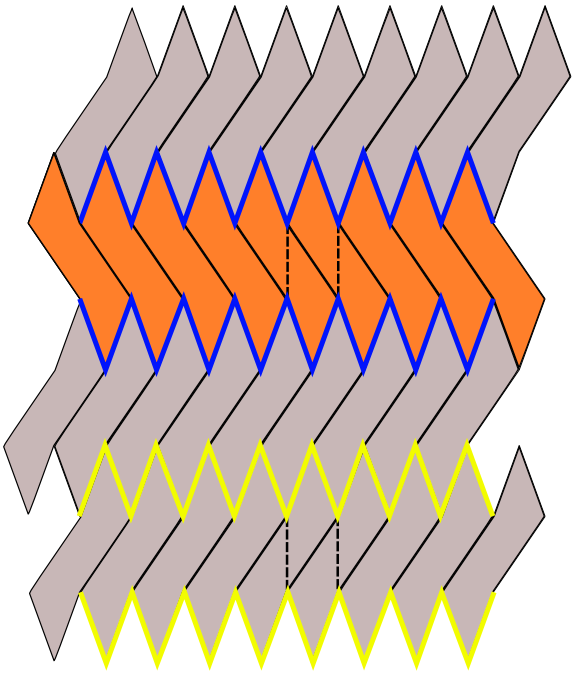


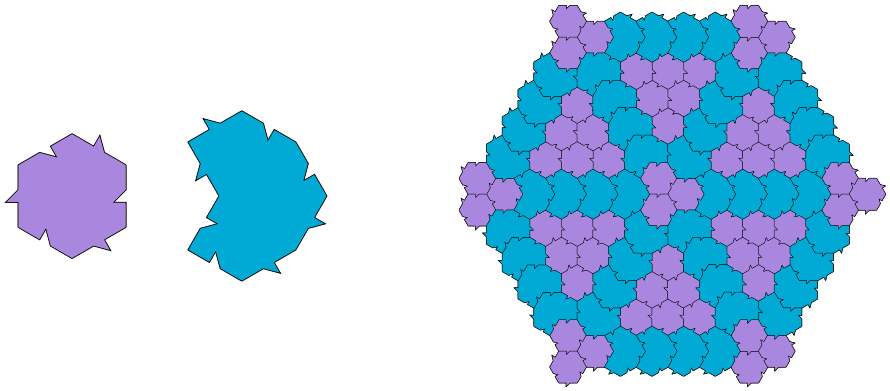
Figure 1 A $\mathfrak{c}$-morphic tile.

2 Construction of the σ-morphic convex protoset

As mentioned, restricting to convex tiles has often been done in the literature. For instance, all the convex tiles that tile the plane have been classified (see the full version of this paper for details [2]). To name a few others, the problem of the existence of a convex tile whose Heesch number is larger than 1 yet finite is still open [5]. Results of Ammann, Danzer, and Grünbaum and Shephard [4], page 549, are, however, much more in line with what we are trying to achieve. They managed to construct sets of tiles which have 3, 9 and 6 convex polygons, respectively, which are aperiodic. This brings us our main theorem:

▶ **Theorem 2.1.** *There exists a protoset $\mathcal{P}$ consisting of convex prototiles that is σ-morphic.*

Proof. We will show that the protoset consisting of distinct polygons shown in Figure 4 has the desired property. The main idea is to choose the angles and sides of the convex polygons in such a way that the only way in which any particular vertex of any of those tiles could be surrounded would be to force the configurations from Figure 4. For instance, the angles of the lowermost three tiles into which the right tile from Figure 4 is divided are such that the



(a) (b)

Figure 2 (a) The σ-morphic protoset discovered by Schmitt [8] and (b) one of the tilings it admits.

only way in which the angle of tile 4 that is incident to the vertex Q can be surrounded fully is in the way that is shown in the figure.

The configurations of tiles shown in Figure 4 behave exactly as the protoset discovered by Schmitt (we describe this below) and this in turn forces $\mathcal{P}$ to be σ-morphic: In any tiling by $\mathcal{P}$ the tiles can be grouped into the two configurations from Figure 4 and so we obtain a tiling by Schmitt's protoset. Therefore, there are at most $\aleph_0$ many tilngs by this protoset, and obtaining $\aleph_0$ many tilings is achieved by dividing each tile in any tiling by Schmitt's protoset into the prototiles of $\mathcal{P}$ exactly as in Figure 4.

Since we are aiming to construct the smallest protoset with this property, we will obtain our protoset by dividing the protoset of just 2 prototiles, those from Figure 2. The first step is to modify the bumps and nicks present at the original tiles in a manner shown in Figure 4. To do this, we exchange the edge with a bump on it with the two edges that are making an angle α above that original edge and that are protruding away from it, and similarly for the nicks; cf. Figure 3. We have to make sure that they are still asymmetric and that the bump

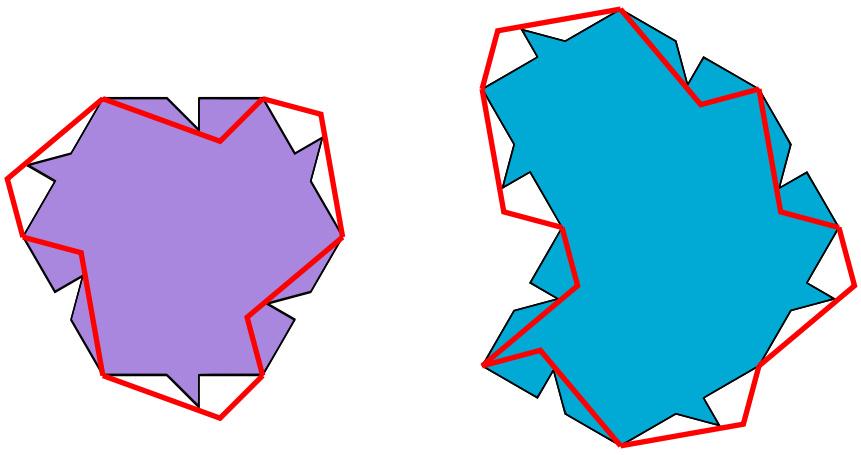


Figure 3 The first step in the construction of $\mathcal{P}$.

can only be accommodated by a nick and vice versa. The way we do this is to take the angle α of the new bumps and nicks to be unique among the angles (and their sums) of tiles in $\mathcal{P}$. We will have some additional requirements on α that will come from the forcing of the desired configurations that we mentioned, but we will focus on those later.

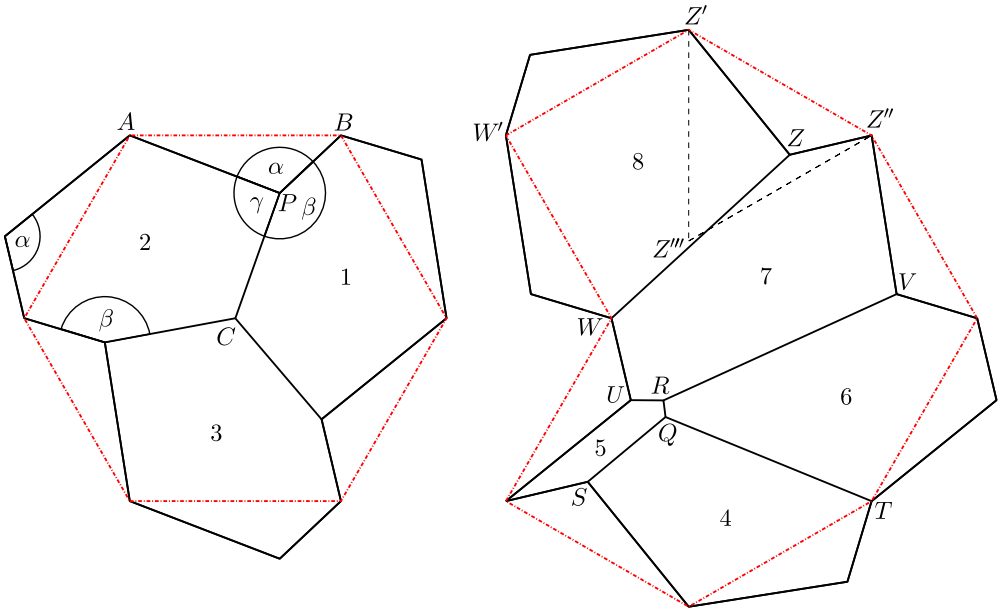


Figure 4 Prototiles of the protoset $\mathcal{P}$. There are four distinct convex hexagons in $\mathcal{P}$ and two distinct convex pentagons. The original protoset of Schmitt that we are subdividing to obtain $\mathcal{P}$ is shown in red (without the bumps and nicks) and is referred to as the original protoset in the proof.

In the following, we ensure that no two consecutive edges of the obtained tiles, which are not the edges of the original tiles (but with modified bumps and nicks), are of the same length (with the exception of tile 1, which we will consider separately). Also, we don't want any number of angles to sum up to θ or $180° - \theta$, for any inner angle θ of any of the tiles in $\mathcal{P}$. The reason for introducing these constraints is that they force the configurations around vertices to be equal to those shown in Figure 4. As long as at least one tile is present, we have to prove that the right tiles (with adequate angles at the vertex) will be present around the vertex, that they will be in the same order around the vertex as shown, and that no tile will be of the opposite orientation to that shown in Figure 4 (i.e. flipped). We first have to make sure that the correct tiles appear around the vertex, which we do either by letting the angles be such that they only partake in one sum that equates to 360°, or by a bit more complicated argument, but we'll leave that for later. Having forced the correct tiles to appear around each vertex, to see why the order of the tiles will have to coincide with that shown in Figure 4, take again, for example, vertex Q from Figure 4. The same analysis will work for all the vertices in the interior of the original tiles (which are the only ones with which we are concerned) since they are all 3-valent. Note that the edge QS is shorter than the edge QT, and note also that a similar statement holds for other vertices since no two consecutive inner edges of any tile are of equal length. Therefore, among the edges QR, QS and QT, there are no two of the same length (for otherwise some tile would have two consecutive edges of the same length). Assuming now the contrary, that the tiles could be in some other order around this vertex (the other one since there are 2 in total once the tiles that participate in

the surrounding are fixed), we observe that the tile 6 has to touch tile 5 along its edge QR, since otherwise the edges of tiles 5 and 6 that are touching would be of different length. But since QS, QT and QR are all of different lengths, there would be an angle equal to $180° - \theta$, which cannot be filled due to our choice of angles. Now, once the order of the tiles is forced, we have to show that each tile will be in the exact orientation as shown in Figure 4. But this easily follows from the lengths of consecutive inner edges being different and the requirement on angles, using an analogous argument as before.

Let us now describe the way in which we ensure these two conditions (on the angles and lengths of edges). We first describe the way in which the convex polygons are obtained and then show that they indeed fulfill the two conditions. Focusing on the left original tile first, we divide it by connecting some of the vertices of the original tile with the center of the red hexagon, cf. Figure 4. The angle α of the bumps and nicks is chosen such that the only sum that equals 360° and that contains α, β or γ in it is exactly $\alpha + \beta + \gamma = 360°$. Also, the edges ZW and ZZ' (look at the tile on the right) have to be of different lengths (note here that the position of vertex Z relative to the vertices of the original tile – including W – is determined by the angle α), which we will show later. For now, we suppose that α has been chosen and proceed in our discussion, and will retroactively explain all the details. As for dividing the right tile from Figure 4, we connect points W and Z and we place two points Q and R such that the angles that the vertices make with points S, T and U, V, respectively, are unique among the angles, fulfill the criterion, can only sum up to 360° if they are all used in the surrounding of a vertex, and are such that the tiles obtained by connecting Q to S and T and R to U and V are convex, and no two consecutive sides which they do not share with the original tile are of equal length. We will note later that the angles $\angle Z'ZW$ and $\angle Z''ZW$ are determined by the choice of α, so we will consider that they were fixed prior to the placement of QR and "tweak" the angles at Q and R accordingly. That this can be achieved is again fairly obvious since there are continuum many placements of the segment QR that fulfill the convexity criterion (which must lie in the intersection of some halfplanes) so there exists one that fulfills the requirements on the angles since the set of forbidden angles is finite. This finishes the construction of the prototiles of $\mathcal{P}$.

Next we show that indeed the prototiles from $\mathcal{P}$ can be surrounded only in the way they are surrounded in Figure 4. Let us start from tile 1. Tile 1 is different from the others in that it actually has two consecutive edges it does not share with the original tile that are of the same length. This, however, will not be important since the forcing of the placement of tiles will happen along other angles (for instance, angle β). Note that the angle β of each tile can be surrounded only with angles α and γ and exactly in that order and orientation (note here that several tiles have angles equal to α and that therefore this surrounding is not unique; but each tile that has angle α as one of its inner angles will have the two edges incident to it of the same length as all the other such tiles and then the orientation of the tile makes sense). In particular, although, as mentioned, this will not entirely force a unique surrounding, it will force the placement of tile 2 as shown in Figure 4. This tile will in turn force the placement of tile 3 from the same figure and the desired surrounding is obtained. Arguing in a similar manner, we see that any of the tiles $4, 5$ and 6 will force the rest through the angles at Q, and similarly any of tiles $5, 6$ and 7 will force the rest along the angles at vertex R. We are left to show that tiles 7 and 8 force the placement of each other. If the angles at Z are distinct from those at P, we are done. Otherwise, the angles have to be equal. It then suffices to show that the edge ZW is of different length than any of the edges AP, BP and CP. To show this, we will show that $|WZ| > |Z'Z''|$. Note that the point W is obtained by rotating Z' around W' for 90° in the clockwise direction, while point Z''' is

obtained by rotating point Z' around W' by 60° in the same direction, meaning that W is outside the triangle $Z'Z''Z'''$. Furthermore, we claim that $\angle Z'Z'''W$ is obtuse. To see why, observe that $\angle W'Z'''W$ is equal to 75° and that $\angle W'Z'''Z' = 60°$, making $\angle Z'Z'''W$ equal to 135°. This means that $|Z'W| > |Z'Z'''|$ and so the complement of the circle centered at W and of radius $Z'Z'''$ with respect to the triangle $Z'Z''Z'''$ is nonempty and thus the original point Z could have been chosen in that region, making sure that ZW is longer than $Z'Z'''$ which is in turn longer than both ZZ' and ZZ'' since they are contained in the triangle $Z'Z''Z'''$. As $Z'Z \cong AP$ and $Z''Z \cong BP$, and as $\triangle ABC \cong \triangle Z'Z''Z'''$, we have shown that WZ is longer than each of AP, BP and CP. We next proceed to use the argument that we have used when discussing the orientations of the tiles to show that indeed tiles 7 and 8 have to meet as is shown in Figure 4.

As now the protoset $\mathcal{P}$ behaves in the same way as the pair of prototiles discovered by Schmitt (see Figure 2) we have shown that $\mathcal{P}$ is indeed σ-morphic.

◀

Note that the left original prototile had to be divided in a symmetric manner: were it otherwise, we could at each place in the tiling choose a different orientation of the tile, which would lead to $\mathfrak{c}$ many tilings.

Acknowledgments

The author is supported by the grant GAČR grant 23-04949X, GAUK grant VV–2025–2608 22 and by Višegrad Fund. The author would also like to thank Tomaš Valla for attracting his attention to the problem discussed in this paper.

References

1 Bojan Bašić, Aleksa Džuklevski, and Anna Slivková, Solutions to Seven and a Half Problems on Tilings. Electron. J. Comb., 30, 2023, https://doi.org/10.37236/11813

2 Aleksa Džuklevski, A σ-morphic convex protoset, (2025). Arxiv: https://doi.org/10.48550/arXiv.2507.20867

3 Branko Grünbaum and Geoffrey C. Shephard, Patch-determined tilings, *Math. Gaz.* **61** (1977), 31–38.

4 Branko Grünbaum and Geoffrey C. Shephard, *Tilings and Patterns*, W. H. Freeman and Company, New York (1987).

5 Casey Mann, Problems involving simple shapes and small protosets, in: *School and Workshop: Combinatorics on Words and Tilings,* Centre de recherches mathématiques, Montréal, Canada, March 27 – April 7, 2017.

6 Joseph S. Myers, Polyomino, polyhex and polyiamond tiling, https://www.polyomino.org.uk/mathematics/polyform-tiling/.

7 Peter Schmitt, Pairs of tiles which admit finitely or countably infinitely many tilings, *Geom. Dedicata* **20** (1986), 133–142. https://doi.org/10.1007/BF00164394

8 Peter Schmitt, σ-morphic sets of prototiles, *Discrete Comput. Geom.* **2** (1987), 271–295. https://doi.org/10.1007/BF02187884

Lower Bounding the Number of Triangulations as a Function of the Convex Hull Size

Justin Dallant[1]

1 **Institute of Theoretical Computer Science, Technische Universität Dresden, Germany**
justin.dallant@tu-dresden.de

Abstract

For a planar set P of n points in general position, h of those being extremal, let $t(P)$ denote the number of geometric triangulations P admits. It is conjectured that for any n, $t(P)$ is minimized when P is a so-called double circle configuration (which has half the points on the convex hull). We prove lower bounds on $t(P)$, expressed as functions of both n and h. Our bounds improve asymptotically on the best known bounds for $h \leq 0.64 \cdot n$. In particular we show that, for large enough n,

- if $h = O(1)$, then $t(P) \geq 2.72^n$, improving the bounds of 2.63^n by McCabe and Seidel (EWCG 2004) and 2.631^n by Aichholzer et al. (SoCG 2016);
- if $h \approx n/2$ (which is the regime of the family of configurations conjectured to minimize the number of triangulations) then $t(P) \geq 2.93^n$, improving the bound of 2.83^n by Sharir, Sheffer and Welzl (J. Comb. Theory. 2011);
- regardless of h, $t(P) \geq 2.637^n$, slightly improving the aforementioned bound of 2.631^n (which also holds regardless of h) by Aichholzer et al.

1 Introduction

Given a planar set of $n \geq 3$ points P in general position, h of those extremal, a triangulation of P is a maximal geometric (i.e. straight-edge) plane (i.e. crossing-free) graph with vertex set P. In this work we are interested in improved lower bounds on $t(P)$, the number of different triangulations of P, expressed as functions of n and h (or n alone).

In general, no provably tight bounds on $t(P)$ are known. Over the years, different works [4, 16, 5, 12, 11, 15, 13] have gradually improved the upper bounds, from the seminal result in [4] showing that $t(P) \leq 10^{13n}$ up to the currently best known bound $t(P) \leq 30^n$ by Sharir and Sheffer [13]. For some time, the known family of configurations with the most triangulations asymptotically was given by the double chains [7] (which have $\tilde{\Theta}(8^n)$ triangulations[1]), until a family with $\Omega(8.48^n)$ triangulations was constructed in [2]. This was later improved to $\Omega(8.65^n)$ [6], and most recently to $\Omega(9.08^n)$ [10].

Turning our attention to the minimum number of triangulations a set of points must have, the first non-trivial lower bound was given by Aichholzer, Hurtado and Noy [3], who showed that $t(P) \geq \Omega(2.33^n)$. This was improved to $t(P) \geq \Omega(2.4317^n)$ by Sharir, Sheffer, and Welzl [14], then $t(P) \geq \Omega(2.631^n)$ by the authors of [1]. On the other hand, it is conjectured that the double circle configuration [11] (which has $h = \lfloor \frac{n}{2} \rfloor$ extremal points and $\tilde{\Theta}(12^{n/2}) \approx 3.464^n$ triangulations) is the minimizing configuration for all $n \geq 3$. This has been verified computationally up to $n = 15$ [1].

[1] Throughout the paper, we use the notations $\tilde{\Theta}, \tilde{O}, \tilde{\Omega}$ to signify that polynomial factors are ignored.

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

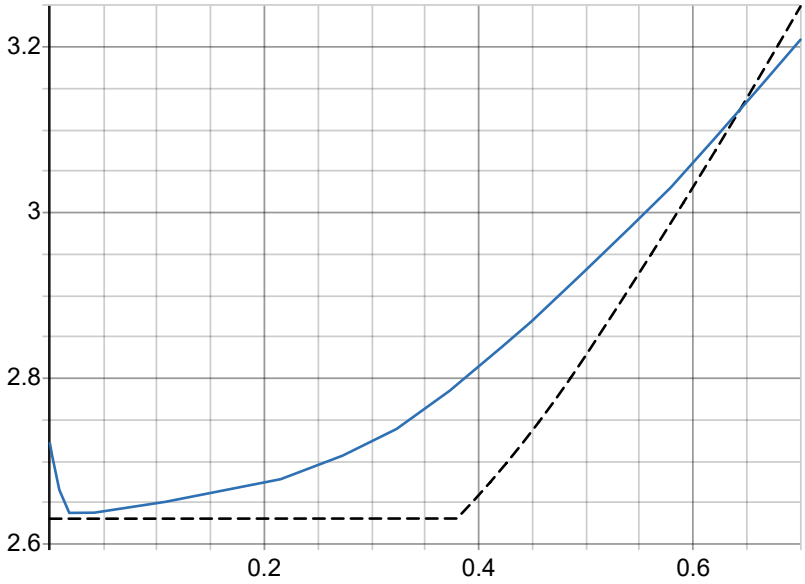


Figure 1 Plot of functions $f(x)$ such that the number of triangulations of a set of n points, h among them being extremal, is at least $f(h/n)^n$ for n large enough. In solid blue, the bounds obtained in this paper. In dotted black, the previous best bounds.

Taking h into account, McCabe and Seidel [8] showed that $t(P) \geq \Omega(2.7272^h \cdot 2.2^{n-h})$ and that $t(P) \geq \Omega(2.63^n)$ when $h \leq O(1)$ (this was later superseded by the aforementioned bound of [1] which holds for all h). Sharir, Sheffer, and Welzl [14] showed that if $h \geq n/2$, then $t(P) \geq \tilde{\Omega}(2^{n+h})$ and if $h \leq n/2$, then

$$t(P) \geq \tilde{\Omega}\left(2^{n+h} \cdot 5^{\frac{4n-3h}{4}} \cdot (1-h/n)^{n-h} \cdot (h/n)^{\frac{h}{4}} \cdot (4-3h/n)^{-\frac{4n-3h}{4}}\right).$$

When $n = h$ (i.e. the points are in convex position), the number of triangulations of P is well known to be given exactly by $C_{n-2} = \tilde{\Theta}(4^n)$, where C_k denotes the kth Catalan number.

Contributions.
In this work, we improve the known asymptotic lower bounds on $t(P)$ in terms of n and h for all $h \leq 0.67 \cdot n$, and slightly improve on the best known lower bound in terms of n alone. Our bounds (together with the best known previous bounds) are illustrated in Figure 1, in the form of the plot of a function $f(x)$ such that $t(P) \geq \Omega(f(\frac{h}{n})^n)$.

Some conventions and notations.
Throughout the paper, we consider only finite point sets in the plane and in general position. If P is a set of points, we let $\text{ext}(P)$ denote the set of its extremal points and $\text{int}(P)$ denote the set of its interior (i.e. non-extremal) points. We will generally use the notation $n = |P|$ and $h = |\text{ext}(P)|$. When using asymptotic (big-Oh) notation for functions of both n and h, this is to be understood as to be with respect to n (the relation should still hold for any constant h, and not only for h large enough).

We let $t(n)$ denote the minimum number of triangulations a set of n points in general position may have. Similarly, we let $t(n,h)$ (resp. $t(n,h^+)$) denote the minimum number of

triangulations a set of n points in general position with h extremal points (resp. at least h extremal points) may have. We let $C_k = \frac{1}{k+1}\binom{2k}{k} = \tilde{\Theta}(4^k)$ denote the kth Catalan number.

Due to space constraints, some proofs and details are missing from this extended abstract, and will appear in the full version of this work.

2 Prime triangulations

The following type of triangulation will serve as useful building blocks in our constructions, and are at the heart of our improvements for point sets with few extremal points.

▶ **Definition 2.1.** A triangulation T of a set of points P is *prime* if $|P| > 3$ and every triangle of T bounds a face of T.
We let $t'(P)$ denote the number of prime triangulations of P.
We define $t'(n,h)$ for prime triangulations as we did $t(n,h)$ for arbitrary triangulations.

Lower bounding the number of prime triangulations.

Let us consider bounds on the number of prime triangulations a set of points must have.

▶ **Definition 2.2.** Call a set of n points P *special* if $n \neq 4$, $|\operatorname{ext}(P)| = 3$ and there is a point $p \in \operatorname{ext}(P)$ such that $P \setminus \{p\}$ is in convex position.

We have the following.

▶ **Theorem 2.3.** *Any non-special set of $n \geq 4$ points has at least one prime triangulation, and*
- $t'(n,n) = C_{n-2}$,
- $t'(n,n-1) = C_{n-2} - 2C_{n-3}$,
- *if $n \geq 7$ and $3 \leq h < n$, then $t'(n,h) \geq t'(n-1,h)$.*

In particular, we have $t'(n,h) \geq 1$ for $h \geq 4$ and $t'(n,h) \geq \tilde{\Omega}(4^h)$.

For sets of up to $n = 11$ points, we have computed the exact values of $t'(n,h)$ by exhaustively considering all combinatorially distinct (generalized) sets of n points, using software by Rote [9]. For larger values of n, we can use these values together with the previous theorem to recursively compute lower bounds on $t'(n,h)$.

From prime triangulations to all triangulations.

We now show how to exploit bounds on the number of prime triangulations to obtain bounds on the overall number of triangulations.

▶ **Definition 2.4.** Let P be a set of n points, and S a subset of $\operatorname{int}(P)$. If T is a triangulation of $P \setminus S$, we let $\mathrm{r}(S,P,T)$ denote the number of different triangulations of P which can be obtained by adding edges to T.
We let $\mathrm{r}(S,P)$ denote the minimum of $\mathrm{r}(S,P,T)$ over all triangulations T of $P \setminus S$.
For $3 \leq h \leq n$ and $0 \leq k \leq n-h$, we let $\mathrm{r}_h(k,n)$ denote the quantity

$$\min_{\substack{k_1,\dots,k_{2(n-k-1)-h} \geq 0 \\ \sum k_j = k}} \left\{ \prod_{j=1}^{2(n-k-1)-h} t(k_j+3,3). \right\}$$

▶ **Lemma 2.5.** *Let P be a set of n points, and S a subset of $\operatorname{int}(P)$. Let $h = |\operatorname{ext}(P)|$ and $k = |S|$. Then $\mathrm{r}_h(k,n) \leq \mathrm{r}(S,P)$.*

The following follows directly from the definition.

▶ **Lemma 2.6.** *Let $\sigma > 0$ and $\tau \geq 1$. If $t(n,3) \geq \sigma\tau^{n-3}$ for all $n \geq 3$, then*

$$\mathrm{r}_h(k,n) \geq \sigma^{2(n-k-1)-h} \cdot \tau^k.$$

▶ **Lemma 2.7.** *Let P be a set of n points with h extremal points. Then,*

$$t(P) \geq \sum_{S \subset \mathrm{int}(P)} t'(P \setminus S) \cdot \mathrm{r}(S,P).$$

Thus, by Lemma 2.5,

$$t(n,h) \geq \sum_{j=0}^{n-h} \binom{n-h}{j} \cdot t'(n-j,h) \cdot \mathrm{r}_h(j,n). \tag{1}$$

For $h = 3$, the bound on $t(n,h)$ in the previous lemma reduces to $t(n,h) \geq (n-3) \cdot r_h(n-4,n)$ as $t'(n,3) = 0$ for all $n \neq 4$. For asymptotic results, we will use the following.

▶ **Lemma 2.8.** *Let $\sigma > 0$ and $\tau > 1$. If $t(n,3) \geq \sigma\tau^{n-3}$ for all $n \geq 3$, then*

$$t(n,h) \geq \tilde{\Omega}\left(\min\{4^n, (\tau+\sigma^2)^{n-h}(4\sigma)^h\}\right).$$

3 Point sets with triangular convex hulls

We now focus on specializing the method of Lemma 2.7 to sets of points with a convex hull of size 3. In order to do so, it will be useful for the sum to be over individual points, rather than over subsets of points.

▶ **Definition 3.1.** A *valid weight function* is a function which to any pair (P,p), where P is a set of points with $|\mathrm{ext}(P)| = 3$, and p an interior point of P, assigns a non-negative value $w_P(p)$, such that for all P,

$$\sum_{p \in \mathrm{int}(P)} w_P(p) \leq t'(P).$$

Given P and a valid weight function w, we define the *value* of an interior point p of P (with respect to w) as

$$v_P^w(p) = \sum_{S \subset \mathrm{int}(P \setminus \{p\})} w_{P \setminus S}(p) \cdot \mathrm{r}(S,P).$$

▶ **Lemma 3.2.** *Let P be a set of n points and w a valid weight function. Then,*

$$t(P) \geq \sum_{p \in \mathrm{int}(P)} v_P^w(p).$$

There are two challenges to exploiting this bound for the purpose of lower bounding $t(n,3)$: the need for a (good) valid weight function, and the need to minimize the bound over all (combinatorially different) sets P of n points with 3 extremal points. In order to tackle both, we will follow the following strategy:

- Associate to every interior point a so-called *signature* (which will be a constant-sized vector of integers).
- Express a valid weight function as a function of the signature alone.

- Find conditions that must hold for every set of signatures induced by the interior points of a set P of n points with 3 extremal points.
- Minimize the bound over all sets of signatures for which these conditions hold.

With the right conditions, the last step of this process will hopefully be computationally easier to carry out than minimizing over sets of points. In the full version of this work, we exploit this approach for different types of signatures to obtain lower bounds on $t(n, h)$.

4 Improved bounds for larger convex hulls and asymptotic bounds

We now turn our attention towards improving some of the bounds used in [3] (also in [1]) for sets of points with large convex hulls. We do this by refining the bounds to be expressed in terms of $t(n, h^+)$ rather than $t(n)$ (see Appendix **??**).

In fact, this can also be exploited to obtain asymptotic bounds as follows.

▶ **Theorem 4.1.** *Let $\sigma \geq 0$, $\tau \geq 1$ and $\mu \geq 1$ be real values. Let $a \geq 3$ be an integer, and $n_0 = 2a + \left\lceil \frac{6}{\sigma\tau^2\mu^2} \right\rceil$. Suppose that for all $a \leq n < n_0$ and all $3 \leq h \leq n$, $t(n, h) \geq \sigma\tau^n\mu^h$. Then $t(n, h) \geq \sigma\tau^n\mu^h$ for all $3 \leq h \leq n$. If $a = 3$, the result holds already for $n_0 = \left\lceil \frac{6}{\sigma\tau^2\mu^2} \right\rceil$.*

Compare this to the mechanism used by [3, 1], which only allows for values of σ of the form $1/k$ (for some integer $k \geq 1$), has generally worse dependencies in σ and a, and does not yield bounds which take h into account.

It remains to find a good induction base to apply Theorem 4.1 to. For all values of $h \leq n$ up to $n = 13$, $t(n, h)$ was computed by the authors of [1], and $t(n, h)$ was also computed for some other values of $h \leq n$ up to $n = 16$. For values where this is not the case, we use the bounds we establish, as well as bounds from [1] and [14] to compute bounds on $t(n, h)$ up to $n = 1500$. Details on which bounds are used depending on the values of n and h will be given in the full version of this work. Applying Theorem 4.1 for different values of σ, τ, μ we get the following.

▶ **Theorem 4.2.** *For all $n \geq h \geq 3$, the following bounds hold:*

$$t(n,h) \geq 0.0004992 \cdot 2.6302^n \cdot 1.0761^h, \qquad t(n,h) \geq 0.0004803 \cdot 2.6243^n \cdot 1.0995^h,$$
$$t(n,h) \geq 0.0004190 \cdot 2.5758^n \cdot 1.1994^h, \qquad t(n,h) \geq 0.0003884 \cdot 2.5396^n \cdot 1.2635^h,$$
$$t(n,h) \geq 0.0003381 \cdot 2.4569^n \cdot 1.3998^h, \qquad t(n,h) \geq 0.0003176 \cdot 2.4119^n \cdot 1.4712^h,$$
$$t(n,h) \geq 0.0003054 \cdot 2.3741^n \cdot 1.5242^h, \qquad t(n,h) \geq 0.0002965 \cdot 2.3316^n \cdot 1.5750^h,$$

We additionally compute bounds on $t(n)$ up to $n = 50000$, and apply Theorem 4.1 (with $\mu = 1$) to get the following.

▶ **Theorem 4.3.** *For all $n \geq 3$,*

$$t(n) \geq 0.00001725 \cdot 2.6377^n$$

McCabe and Seidel [8] use the fact that if $t(n, 3) \geq \sigma\tau^{n-3}$ for all $3 \leq n \leq \lceil \sigma^{-2}\tau \rceil$, then $t(n, 3) \geq \sigma\tau^{n-3}$ for all $n \geq 3$ to obtain the bound $t(n, 3) \geq \Omega(2.63^n)$. Applying the same argument with the bounds on $t(n, 3)$ we computed up to $n = 2000$ we get

▶ **Theorem 4.4.** *For all $n \geq 3$,*

$$t(n,3) \geq 0.03690 \cdot 2.7207^n,$$
$$t(n,3) \geq 0.06601 \cdot 2.7161^n,$$
$$t(n,3) \geq 0.21590 \cdot 2.6460^n.$$

Finally, using these bounds together with Lemma 2.8 we get the following.

▶ **Theorem 4.5.** *We have,*

$$t(n,h) \geq \tilde{\Omega}\left(2.7220^{n-h} \cdot 0.1476^h\right),$$
$$t(n,h) \geq \tilde{\Omega}\left(2.7204^{n-h} \cdot 0.264^h\right),$$
$$t(n,h) \geq \tilde{\Omega}\left(2.6926^{n-h} \cdot 0.8636^h\right).$$

In particular, if $h \leq O(1)$, $t(n,h) \geq \tilde{\Omega}\left(2.722^n\right)$.

References

1. Oswin Aichholzer, Victor Alvarez, Thomas Hackl, Alexander Pilz, Bettina Speckmann, and Birgit Vogtenhuber. An improved lower bound on the minimum number of triangulations. In *32nd International Symposium on Computational Geometry: SoCG'16, June 14-17, 2016, Boston, MA, USA*, volume 51, pages 7:1–7:16. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2016.
2. Oswin Aichholzer, Thomas Hackl, Clemens Huemer, Ferran Hurtado, Hannes Krasser, and Birgit Vogtenhuber. On the number of plane geometric graphs. *Graphs and Combinatorics*, 23(Suppl 1):67–84, 2007.
3. Oswin Aichholzer, Ferran Hurtado, and Marc Noy. A lower bound on the number of triangulations of planar point sets. *Computational Geometry*, 29(2):135–145, 2004.
4. Miklós Ajtai, Vašek Chvátal, Monroe M Newborn, and Endre Szemerédi. Crossing-free subgraphs. In *Theory and Practice of Combinatorics.*, volume 60 of *North-Holland Mathematics Studies*, pages 9–12. Elsevier, 1982.
5. Markus Denny and Christian Sohler. Encoding a triangulation as a permutation of its point set. In *CCCG*, volume 97, pages 39–43, 1997.
6. Adrian Dumitrescu, André Schulz, Adam Sheffer, and Csaba D Tóth. Bounds on the maximum multiplicity of some common geometric graphs. *SIAM Journal on Discrete Mathematics*, 27(2):802–826, 2013.
7. Alfredo Garcıa, Marc Noy, and Javier Tejel. Lower bounds on the number of crossing-free subgraphs of kn. *Computational Geometry*, 16(4):211–221, 2000.
8. Paul McCabe and Raimund Seidel. New lower bounds for the number of straight-edge triangulations of a planar point set. In *Proc. 20th European Workshop on Computational Geometry (EWCG 2004)*, pages 175–176, 2004.
9. Günter Rote. Numpsla–an experimental research tool for pseudoline arrangements and order types. *arXiv preprint arXiv:2503.02336*, 2025.
10. Daniel Rutschmann and Manuel Wettstein. Chains, koch chains, and point sets with many triangulations. *Journal of the ACM*, 70(3):1–26, 2023.
11. Francisco Santos and Raimund Seidel. A better upper bound on the number of triangulations of a planar point set. *Journal of Combinatorial Theory, Series A*, 102(1):186–193, 2003.
12. Raimund Seidel. On the number of triangulations of planar point sets. *Combinatorica*, 18(2):297–299, 1998.

13 Micha Sharir and Adam Sheffer. Counting triangulations of planar point sets. *The Electronic Journal of Combinatorics*, pages P70–P70, 2011.

14 Micha Sharir, Adam Sheffer, and Emo Welzl. On degrees in random triangulations of point sets. *Journal of Combinatorial Theory, Series A*, 118(7):1979–1999, 2011.

15 Micha Sharir and Emo Welzl. Random triangulations of planar point sets. In *Proceedings of the twenty-second annual symposium on Computational geometry*, pages 273–281, 2006.

16 Warren Douglas Smith. *Studies in computational geometry motivated by mesh generation*. Ph.D. Thesis, Princeton University, 1989.

Geometric Give and Take*

Oswin Aichholzer[1], Katharina Klost[2,3], Kristin Knorr[2], Viola Mészáros[4], and Josef Tkadlec[5]

1 **Graz University of Technology**
`oswin.aichholzer@tugraz.at`
2 **Freie Universität Berlin**
`kathklost@inf.fu-berlin.de, knorrkri@inf.fu-berlin.de`
3 **Universität Tübingen**
4 **Bolyai Institute, University of Szeged**
`meszaros.viola@gmail.com`
5 **Charles University**
`josef.tkadlec@iuuk.mff.cuni.cz`

Abstract

We consider a special, geometric case of a balancing game introduced by Spencer in 1977. Consider any arrangement $\mathcal{L}$ of n lines in the plane, and assume that each cell of the arrangement contains a box. Alice initially places pebbles in each box. In each subsequent step, Bob picks a line, and Alice must choose a side of that line, remove one pebble from each box on that side, and add one pebble to each box on the other side. Bob wins if any box ever becomes empty. We determine the minimum number $f(\mathcal{L})$ of pebbles, computable in polynomial time, for which Alice can prevent Bob from ever winning, and we show that $f(\mathcal{L}) = \Theta(n^3)$ for any arrangement $\mathcal{L}$ of n lines in general position.

Related Version http://arxiv.org/abs/2603.08074

1 Introduction

Inspired by the following game that appeared in the shortlist of problems for the International Math Olympiad in 2019 [4], we consider a special case of the balancing game introduced by Spencer [6] in 1977. The game [4] is played with n boxes that are placed in a row and each box can contain a number of pebbles. There are two players: Alice and Bob. Initially, Alice places p_i pebbles into box i. Then the game proceeds in rounds: In each round, first Bob picks a line ℓ_i with $1 \leq i < n$ that separates boxes $1, \dots, i$ from boxes $i+1, \dots, n$. Second, Alice chooses one side of the line, removes one pebble from each box on that side, and adds one pebble to each box on the opposite side. (So, if Alice picked the left side of line ℓ_i, then she removes one pebble from each box $1, \dots, i$ and adds one pebble to boxes $i+1, \dots, n$ each.) The game ends with a win for Bob once any of the boxes becomes empty. Alice's goal is to prevent Bob from winning. The problem asks to determine the smallest total number of pebbles that Alice initially needs to put in the boxes, so that she can prevent Bob from winning. Denoting this number by $f_{1D}(n)$, it can be shown that $f_{1D}(n) = \frac{1}{4}n(n+4)$ if n is even and $f_{1D}(n) = \frac{1}{4}(n+1)(n+3) - 1$ if n is odd, see [4]. In particular, $f_{1D}(n) = \Theta(n^2)$.

In our work, we generalize the game to a 2-dimensional setting as follows: Let $\mathcal{L} = \{\ell_1, \dots, \ell_n\}$ be a set of n lines in the plane. Those lines split the plane into up to $\binom{n+1}{2} + 1$ regions. Each region contains one box. As in the 1-dimensional problem, Alice first places pebbles in the boxes. The game then proceeds in rounds with Bob always picking a line

* J.T. supported by Charles Univ. projects UNCE 24/SCI/008 and PRIMUS 24/SCI/012, and by GAČR project 25-17221S.

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

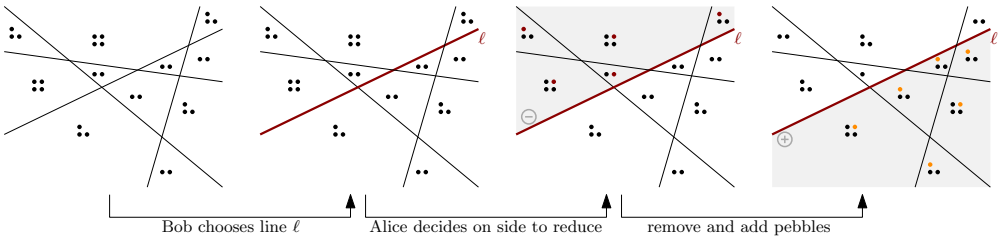


Figure 1 Example for one round of the 2-dimensional game

from $\mathcal{L}$, and Alice picking a side, removing one pebble from each box on that side and adding a pebble to each box on the other side, see Fig. 1 for an example. Given a line arrangement $\mathcal{L}$, let $f(\mathcal{L})$ be the smallest number of pebbles such that Alice can distribute them to the boxes and prevent Bob from winning. The 1-dimensional setting corresponds to the case in which $\mathcal{L}$ consists of $n-1$ parallel lines.

As our main result, we determine $f(\mathcal{L})$ for any line arrangement $\mathcal{L}$. A key notion here is measuring how (un)evenly the regions in the line arrangement are split by the respective lines. More formally, given a line ℓ in a line arrangement $\mathcal{L}$, we define the *rank* of ℓ, denoted c_ℓ, to be the number of regions on the side of ℓ that contains at most half of all the regions.

▶ **Theorem 1.1.** *Let $\mathcal{L}$ be a line arrangement that splits the plane into R regions. Then $f(\mathcal{L}) = R + \sum_{\ell\in\mathcal{L}} c_\ell$.*

We also show that $f(\mathcal{L})$ is cubic, provided that the line arrangement is in general position.

▶ **Theorem 1.2.** *Let $\mathcal{L}$ be a line arrangement with n lines in general position. Then $f(\mathcal{L}) \in \Theta(n^3)$.*

Our results are organized as follows. In Section 2 we introduce the notion of autopilot strategies for Alice, and we prove that one of those autopilot strategies guarantees a win for Alice, provided that she is allowed to initially distribute at least $R + \sum_{\ell\in\mathcal{L}} c_\ell$ pebbles. This establishes the "$\leq$" inequality in Theorem 1.1.

In Section 3 we prove that with fewer than $R + \sum_{\ell\in\mathcal{L}} c_\ell$ pebbles, Bob can force a win in a finite number of moves. In high level terms, Bob's strategy proceeds in stages, where in each stage either the total number of pebbles decreases, or the pebbles are shuffled from "more important" boxes to the "less important" boxes. Using a suitable monovariant, we show that eventually one box becomes empty. This establishes the "$\geq$" inequality in Theorem 1.1.

Finally, in Section 4 we prove that as long as the arrangement of n lines is in general position, the quantity $\sum_{\ell\in\mathcal{L}} c_\ell$ is cubic in n. The argument uses the $\leq k$-level Theorem [1].

Claims whose proofs appear only in the full version are labeled with an asterisk ($\star$).

Related work

The game considered in this work is a special case of a so-called *balancing game* introduced in full generality by Spencer [6]. There, the game is determined by a convex body $K \in \mathbb{R}^d$, a set $V \subseteq \mathbb{R}^d$ of d-dimensional vectors, and a starting position $\xi \in K$. In the i-th round, one player (the "pusher") picks a vector $v \in V$ and the other player (the "chooser") moves from the current position ξ_{i-1} to either $\xi_i = \xi_{i-1} + v$ or to $\xi_i = \xi_{i-1} - v$. The pusher wins if the current position ever leaves K, and the chooser wins otherwise.

To see why the game considered here is a special case of a balancing game, fix any order over the $d \leq \binom{n+1}{2} + 1$ boxes, and represent the current configuration $(x_1, \ldots, x_d)$ of pebbles

as a point $(x_1, \ldots, x_d) \in \mathbb{R}^d$. Each line $\ell_i \in \mathcal{L}$ gives rise to a single vector $v_i \in \{-1, +1\}^d$, and the convex region $K = K^+$ is the d-dimensional "quadrant" with all coordinates positive.

The balancing game was studied in several other works, see e.g. [2] and references in [5]. To our knowledge, the question of finding the smallest $\sum_i x_i$ for which the chooser wins with $K = K^+$ has not been studied before.

2 Autopilot strategy for Alice

In this section we describe the *autopilot strategy* for Alice that, given a line arrangement $\mathcal{L}$, allows her to initially place pebbles, and then play so as to prevent Bob from ever emptying any box. The main idea is that for every line in the arrangement, Alice first assigns a side where she will remove the pebbles when the line is first played. After that for every line she alternates the side from which she removes the pebbles. At the beginning, she assigns just enough pebbles to each box to guarantee that if every line is played exactly once, regardless of the order in which they are played, no box becomes empty. We show that when Alice follows this strategy and updates the sides of the lines accordingly, the distribution of the pebbles will always maintain this property and thus Bob cannot win. The upper bound on $f(\mathcal{L})$ then follows from taking the assignment of the sides that leads to the smallest number of pebbles.

Formally, let $\mathcal{L} = \{\ell_1, \ldots, \ell_n\}$ be a line arrangement and let B be the set of boxes defined by $\mathcal{L}$. We call a function $p : B \to \mathbb{N}$ that assigns pebbles to the boxes a *pebble distribution*. Consider the two half-planes defined by ℓ and let $\sigma(\ell)$ be a function that assigns an orientation "+" to one of the half planes, and the orientation "−" to the other. Let $\tau_\sigma \colon B \to 2^{\mathcal{L}}$ be the function that assigns to each box b the set of all lines for which b lies in the side labeled "−".

Initially, Alice chooses a certain orientation σ and for each box b she sets

$$p(b) = |\tau_\sigma(b)| + 1. \tag{1}$$

We call pebble distributions defined with respect to an orientation of the lines *autopilot distributions*. During the game, Alice will update the orientation σ, such that (1) always holds. Let σ_t be the orientation after t rounds and let p_t be the corresponding autopilot distribution.

Let ℓ be the line picked by Bob in round t. Then Alice removes one pebble from each box in the side of ℓ labeled "−" and adds one pebble to each box on the other side. She then gets σ_{t+1} by flipping the labels of the two sides of ℓ. We call this the *autopilot strategy*.

▶ **Lemma 2.1** (⋆). *If Alice starts with any autopilot distribution and plays the autopilot strategy, all distributions $p_0, \ldots$ of the pebbles to the boxes will be autopilot distributions.*

Since every autopilot distribution has at least one pebble in each box, we conclude:

▶ **Corollary 2.2.** *When Alice starts with an autopilot distribution and plays the autopilot strategy, Bob has no winning strategy.*

As a direct consequence, we obtain the "≤" inequality in Theorem 1.1.

▶ **Lemma 2.3.** *Let $\mathcal{L}$ be any line arrangement. Then $f(\mathcal{L}) \leq R + \sum_{\ell \in \mathcal{L}} c_\ell$.*

Proof. For the initial distribution of pebbles, Alice chooses the autopilot distribution p_a that, for each line $\ell \in \mathcal{L}$, assigns the orientation "−" to the half-plane that has at most half of the boxes. In this way, Alice has distributed precisely $\sum_{b \in B} p_a(b) = R + \sum_{\ell \in \mathcal{L}} c_\ell$ pebbles. If she then follows the autopilot strategy, by Corollary 2.2 no box will ever become empty. ◀

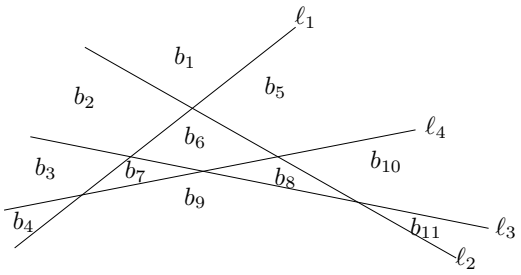


Figure 2 The indices of the lines show how all the lines would be ordered, if $\mathbf{b}$ was on the smaller side of all of them (each line has 4 or 5 boxes on their smaller side). ℓ_1 is before ℓ_2 as the first position of $v_1^{\ell_1} = v_1^{\ell_2}$ agree, but $v_2^{\ell_1} = 1$ and $v_2^{\ell_2} = 0$. If $\mathbf{b} = b_9$ Bob only plays ℓ_3 and ℓ_4

3 Monovariant strategy for Bob

In this section we give a strategy for Bob that allows him to win, if Alices places fewer than $R + \sum_{\ell \in \mathcal{L}} c_\ell$ pebbles. The game is subdivided into stages and we can show that after every stage either Bob wins, or the overall number of pebbles decreased, or the pebble distribution is smaller in some fixed linear order of the pebble distributions. As the latter two cases can only happen finitely many times, Bob will eventually win.

▶ **Lemma 3.1** (⋆). *Let $\mathcal{L}$ be any line arrangement. Then $f(\mathcal{L}) \geq R + \sum_{\ell \in \mathcal{L}} c_\ell$.*

Proof (Sketch). Label the boxes in any order $b_1, \ldots, b_R$, let p_a be an autopilot distribution for $\mathcal{L}$ as in Lemma 2.3, and let p be the current pebble distribution that uses strictly fewer pebbles than p_a. We define a total order $\prec_L$ on the pebble distribution p by the lexicographic order of the list $[p(b_1), \ldots, p(b_R)]$.

Bob will repeat the following stage that either results in an empty box, or reduces the total number of pebbles or the position of p in the total order $\prec_L$. As p uses strictly less pebbles than p_a, by the pigeonhole principle, there is at least one box $\mathbf{b}$ with $p(\mathbf{b}) < p_a(\mathbf{b})$. Bob considers the set of A lines that have $\mathbf{b}$ on their *smaller side*, that is the side with at most half of the boxes, where ties are broken such that b_1 is not on the smaller side. He orders the lines $\ell_1, \ldots, \ell_A$ first decreasing by rank and then within each rank lexicographically by $v^\ell \in \{0,1\}^R$ with $v_i^\ell = 1$ iff b_i is on the smaller side of ℓ. We say Alice *decreases* a line ℓ, if she takes the pebbles from the smaller side, otherwise she *increases* ℓ.

Bob starts by playing ℓ_1. Whenever Bob plays ℓ_i and Alice increases ℓ_i, he next plays ℓ_{i-1}, otherwise he plays ℓ_{i+1}. If Alice increases ℓ_1 or decreases ℓ_A at some point, the stage ends. The sequence of lines played by Bob is subdivided into a subsequence $\ell_1, \ldots, \ell_k$ of lines that Alice increased and a set of pairs (ℓ_i, ℓ_{i+1}) where Alice decreased ℓ_i and increased ℓ_{i+1}. We call such a pair *red*, if ℓ_i and ℓ_{i+1} have different ranks, and *green* otherwise.

If Alice increases ℓ_1 there are no lines that are not in a pair. Thus, the overall number of pebbles did not increase since the start of the stage. If there is at least one red pair or ℓ_1 has rank less than $R/2$ then the stage decreased the number of pebbles. In the other case, there are only green pairs and ℓ_1 has rank $R/2$. Because of the sorting by v^ℓ within each rank, increasing ℓ_1 decreases the most significant position of p and thus the position in $\prec_L$. This can happen only finitely many times, thus Bob eventually wins.

If Alice decreases ℓ_A she also has decreased $\ell_1, \ldots, \ell_{A-1}$ without them being in a pair. Each of the pairs does not change the number of pebbles in $\mathbf{b}$ and thus A pebbles were removed from $\mathbf{b}$, making it empty. Finally, if Alice never decreases ℓ_A or increases ℓ_1 then to

non-paired lines $\ell_1, \ldots, \ell_k$ only add finitely many pebbles and thus the number of red pairs is bounded. This implies that there is a point after which all lines played have the same rank. The ordering by v_ℓ then implies that the position in $\prec_L$ decreases with every green pair. This can happen only finitely many times, and thus Bob wins. ◀

Proof of Theorem 1.1. Immediate from Lemmas 3.1 and 2.3. ◀

4 General position

In this section we prove Theorem 1.2. The upper bound turns out to be simple (any autopilot will do). For the lower bound, we introduce several auxiliary notions, we prove a helpful lemma, and we use a known result about $\leq k$-levels.

▶ **Definition 4.1.** Let $\mathcal{L}$ be a line arrangement. Its *dual graph* is the unweighted graph $D(\mathcal{L}) = (B, E)$ that has one vertex for each box, and edges between boxes that share a side. The distance of two nodes u, v in $D(\mathcal{L})$ is denoted $d(u, v)$. Moreover, given a pebble distribution $p\colon B \to \mathbb{N}$, we say that a pair $u, v \in B$ is *small* if $p(u) + p(v) < d(u, v) + 2$.

▶ **Lemma 4.2** ($\star$)**.** *If a pebble distribution p contains a small pair, then Bob has a winning strategy.*

Proof (Sketch). Take a small pair u, v and consider any shortest path between u and v in the dual graph $D(\mathcal{L})$. Bob focuses on the lines that correspond to edges on that shortest path. He starts at the $p(u)$th line and moves towards u if Alice decreased on the side containing u and towards v otherwise. The claim then follows from similar arguments as in the $1d$ setting [4]. ◀

The second ingredient is a version of a $\leq k$-level Theorem that can be stated for sets of points, or dually for the line arrangements, see also a review [7, Section 1.3].

▶ **Theorem 4.3** (Theorem 1 in [1])**.** *Let $\mathcal{L}$ be an arrangement of n lines, $D(\mathcal{L}) = (V, E)$ its dual graph, and $1 \leq r < n/2$. Then for any $u \in V$ we have $|\{v \in V \mid d(u, v) \leq r\}| \leq r \cdot n$.*

Finally we are ready to prove Theorem 1.2.

Proof of Theorem 1.2. First, note that any autopilot strategy uses $\mathcal{O}(n^3)$ pebbles: Indeed, in every autopilot distribution, all boxes receive at most $n + 1$ pebbles. Since for a line arrangement in general position the number of boxes is $\binom{n+1}{2} + 1$, Alice can distribute $\mathcal{O}(n^3)$ pebbles and prevent Bob from winning.

Next, we prove the lower bound. Let p be any pebble configuration such that Alice wins. If each box has at least $\frac{1}{10}n$ pebbles then the total number of pebbles is at least $\frac{1}{10}n \cdot \frac{1}{2}n^2 = \Omega(n^3)$ and we are done. Suppose otherwise and let $b \in B$ be a box with fewer than $\frac{1}{10}n$ pebbles. Fix a distance $r = \frac{1}{3}n$. By Theorem 4.3 there are at most $r \cdot n = \frac{1}{3}n^2$ boxes at distance at most r from b, so there are at least $\frac{1}{2}n^2 - \frac{1}{3}n^2 = \frac{1}{6}n^2$ boxes at distance at least $\frac{1}{3}n$ from b. By Lemma 4.2, each of those boxes must have at least $r - n/10 \geq \frac{7}{30}n$ pebbles, else Bob wins. Thus, the total number of pebbles is at least $\frac{1}{6}n^2 \cdot \frac{7}{30}n = \Omega(n^3)$. ◀

5 Discussion

There are many directions for future research, we briefly mention two of them: First, given an integer n, determine the smallest and the largest possible value of $f(\mathcal{L}_n)$, where $\mathcal{L}_n$ is an arrangement of n lines in general position. For example, when $n = 5$ a check of all

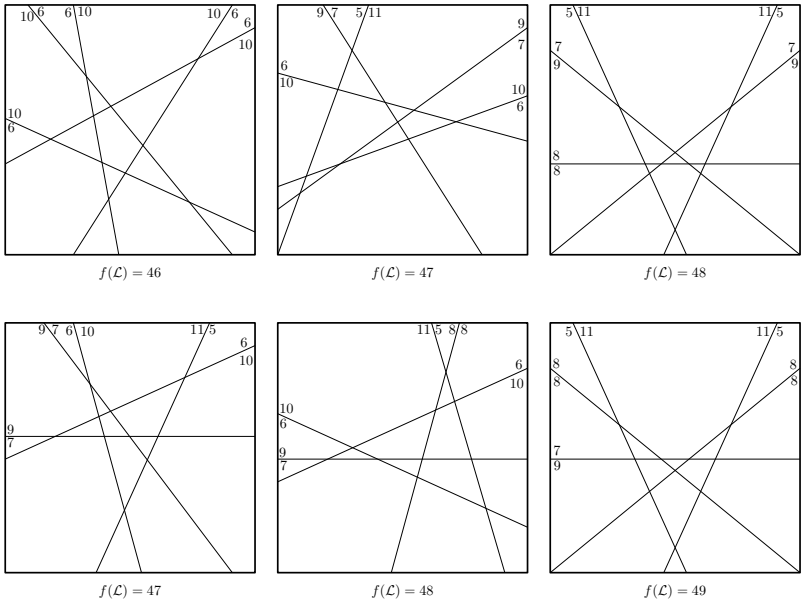


Figure 3 The six combinatorially different simple line arrangements for $n = 5$ lines [3]. Shown is the number of cells on both sides of each line. The optimal autopilot for each arrangement requires $46 \leq f(\mathcal{L}) = \sum_{\ell \in \mathcal{L}} c_\ell + 16 \leq 49$ pebbles.

combinatorially different simple line arrangements [3] shows that $46 \leq f(\mathcal{L}) \leq 49$, see Fig. 3. Second, is there a polynomial-time algorithm that, given a line arrangement and the pebbles in the boxes, decides whether Bob has a winning strategy? We believe we can show that the answer is yes if some box contains a single pebble, but we don't know about the general case.

Acknowledgment We would like to thank the organizers of GGWeek 2025, where this research started.

References

1 Noga Alon and Ervin Győri. The number of small semispaces of a finite set of points in the plane. *Journal of Combinatorial Theory, Series A*, 41(1):154–157, 1986.
2 Imre Bárány. On a class of balancing games. *Journal of Combinatorial Theory, Series A*, 26(2):115–126, 1979.
3 Tobias Christ. Database of combinatorially different simple line arrangements. URL: `https://geometry.inf.ethz.ch/christt/linearr/`.
4 IMO 2019 Comittee. 60th International Mathematical Olympiad. URL: `https://www.imo-official.org/problems/IMO2019SL.pdf#page=49`.
5 Aviezri Fraenkel. Combinatorial games: selected bibliography with a succinct gourmet introduction. *The Electronic Journal of Combinatorics*, pages DS2–Aug, 2012.
6 Joel Spencer. Balancing games. *Journal of Combinatorial Theory, Series B*, 23(1):68–74, 1977.
7 Uli Wagner. k-sets and k-facets. *Contemporary Mathematics*, 453:443, 2008.

Compatible triangulations of simple polygons

Peyman Afshani[1], Boris Aronov[2], Kevin Buchin[3], Maike Buchin[4], Otfried Cheong[5], Katharina Klost[6,8], Carolin Rehs[3,7], and Günter Rote[6]

1 **Department of Computer Science, Aarhus University, Aarhus, Denmark. peyman@cs.au.dk**
2 **Department of Computer Science and Engineering, Tandon School of Engineering, New York University, Brooklyn, NY, USA. boris.aronov@nyu.edu**
3 **Faculty of Computer Science, TU Dortmund, Germany. {kevin.buchin,carolin.rehs}@tu-dortmund.de**
4 **Faculty of Computer Science, Ruhr University Bochum, Germany**
5 **SCALGO, Aarhus, Denmark. otfried@scalgo.com**
6 **Institut für Informatik, Freie Universität Berlin.** {kathklost, rote}@inf.fu-berlin.de
7 **Department of Mathematics and Computer Science, TU Eindhoven, The Netherlands.**
8 **Universität Tübingen, Germany.**

Abstract

Let P and Q be simple polygons with n vertices each. We wish to compute triangulations of P and Q that are combinatorially equivalent, if they exist. We consider two versions of the problem: if a triangulation of P is given, we can decide in $O(n \log n + nr)$ time if Q has a compatible triangulation, where r is the number of reflex vertices of Q. If we are already given the correspondence between vertices of P and Q (but no triangulation), we can find compatible triangulations of P and Q in time $O(M(n))$, where $M(n)$ is the running time for multiplying two $n \times n$ matrices.

Related Version arXiv:2603.01282

1 Introduction

We consider two simple polygons P and Q that both have n vertices, and want to compute triangulations of P and Q that are *compatible*: we can number the vertices of both P and Q counter-clockwise from 0 to $n-1$, and a diagonal (i, j) is in the triangulation of P if and only if it is in the triangulation of Q, see Figure 1.

We study two variants of this problem: In the first variant, we are given a triangulation of P, and ask if there exists a numbering of the vertices of Q such that it has a triangulation compatible to the triangulation of P. Clearly this problem can be solved by trying all the

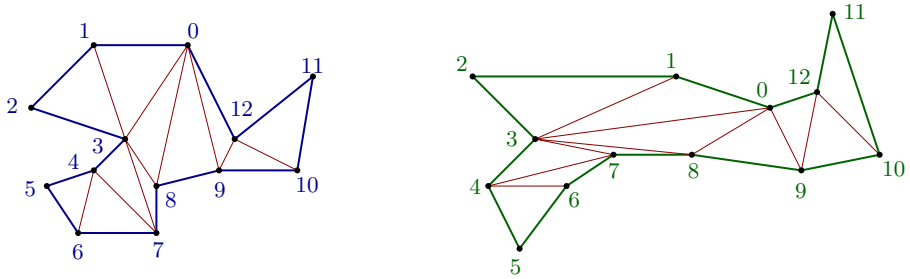


Figure 1 Compatible triangulations of two polygons

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

n distinct numberings of Q. For a fixed numbering, we then only need to test if the diagonals appearing in P exist in Q. This takes $O(n^2)$ time in total.

We improve this by giving an $O(n \log n + nr)$ time algorithm, where r is the number of reflex vertices of Q. As part of our algorithm, we give a data structure that stores a simple polygon P with n vertices in $O(n \log n)$ space, takes $O(n \log n)$ preprocessing time, and can answer in constant time questions of the form: are vertices i and j visible in P?

In the second variant, the vertex numbering of P and Q is given and fixed, and we ask whether there are compatible triangulations of P and Q. It is easy to transform this question to the following: Given a graph G drawn in the plane with straight edges (possibly with crossings) whose vertices form a convex n-gon C, does it contain a triangulation of C?

It is not difficult to solve this question in time $O(n^3)$ using dynamic programming [2]. We improve this to $O(n^\omega)$, where ω is the exponent of matrix multiplication.

Related Work. It is well-known that it is not always possible to find compatible triangulations for two simple polygons P and Q of n vertices. However, Aronov, Seidel, and Souvaine show that by adding $O(n^2)$ Steiner points, it is always possible to find compatible triangulations [2]. They further show that sometimes $\Omega(n^2)$ Steiner points are necessary. They also pose the question how to find the smallest number of Steiner points required for a given pair of simple polygons. This question is still open for simple polygons, whereas for polygons with holes it has been proven to be NP-hard by Lubiw and Mondal [4].

2 A data structure for vertex visibility queries in a simple polygon

▶ **Theorem 2.1.** *A simple n-gon P can be preprocessed in time $O(n \log n)$ into a data structure with $O(n \log n)$ space such that we can answer in constant time queries of the form: are vertex i and j visible in P?*

Proof. We start by computing a triangulation of P. We then recursively split the triangulation using diagonals, obtaining a binary tree $\mathcal{T}$ of height $O(\log n)$. Each node of $\mathcal{T}$ corresponds to a region R of P, which is then split by a diagonal $d = ab$ into the two regions corresponding to the children of the node. We compute, for each vertex v in the region R different from a and b, the interval d_v on d that is visible from v, and store this information with R, see Figure 2.

To answer a visibility query for two vertices v and w, we find the smallest region R that contains both v and w. Let d be the diagonal that splits R. If v and w are disjoint from d, the segment vw lies in P if and only if it intersects the diagonal d, and the intersection point x lies in both intervals d_v and d_w, see Figure 2. This can be checked in constant time.

If one of the points, say w, is an endpoint of d, the points v and w are visible if w lies in d_v.

The full paper explains how to find the region R for answering the query [1]. ◀

3 One triangulation is given

▶ **Theorem 3.1.** *Given two simple n-gons P and Q, and a triangulation T of P, we can determine in $O(n \log n + nr)$ time if there is a compatible triangulation in Q, where r is the number of reflex vertices of Q.*

Proof. Let $P = p_0, p_1, \ldots p_{n-1}$ and $Q = q_0, q_1, \ldots, q_{n-1}$ where vertices are numbered counterclockwise and vertex indices are considered modulo n. Our goal is to find a *rotation* of

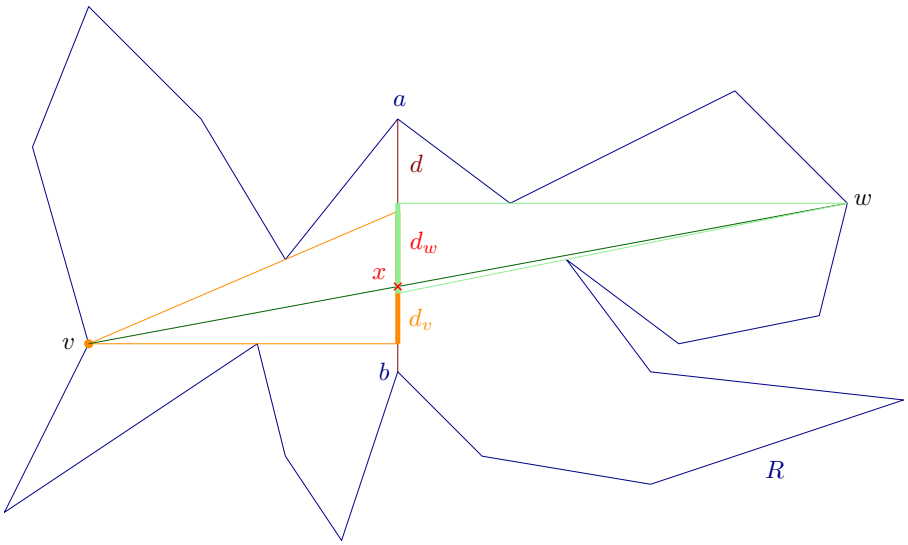


Figure 2 Splitting the region R by a diagonal d

the numbering of the vertices of Q by s steps, such that for all diagonals p_ip_j in T, the corresponding segment $q_{i+s}q_{j+s}$ is a diagonal lying inside Q. In fact, we will identify *all* such rotations s.

The following observation will be handy.

▶ Observation 1. f For every vertex p_i of P, there are indices $1 < s_1 < s_2 < \cdots < s_\ell < n-1$ such that $p_ip_{i+s_1}, p_ip_{i+s_2}, \ldots p_ip_{i+s_\ell}$ are the diagonals incident to p_i in the triangulation of P, in counterclockwise order around p_i. ◀

Consider a reflex vertex q_i of Q. If we extend the two edges incident to q_i into the interior of Q, they form a wedge w_i. Any triangulation of Q contains either

- a diagonal q_iq_j incident to q_i that lies inside the wedge w_i, or
- a triangle $q_iq_jq_k$ whose opposite side q_jq_k crosses the wedge w_i.

Let us say that a rotation s *satisfies* the reflex vertex q_i if the triangulation T of P, after applying the rotation s, fulfills one of the two conditions.

Conceptually, our algorithm fills a table G with n rows corresponding to the different rotations $s = 0, \ldots, n-1$, and r columns corresponding to the reflex vertices of Q. We will *mark* the cell $G[s, i]$ if rotation s satisfies q_i. In the end, we only need to identify the rows s of G where all cells are marked. Indeed, then there are diagonals of P that, after the rotation s, partition Q into pieces that are all convex, since all interior angles are smaller than 180°. It follows that these pieces can be triangulated in any way we want, in particular as prescribed by the given triangulation T of P.

We start by computing the visibility query data structure of Theorem 2.1 for Q. We also precompute, for each reflex vertex q_i of Q, the edges of Q hit by the extension of the edges $q_{i-1}q_i$ and $q_{i+1}q_i$, in $O(n)$ time per vertex q_i.

By Observation 1, applied to Q instead of P, this gives us an interval of indices $\{\sigma_i, \ldots, \tau_i - 1\}$ such that a diagonal q_iq_j lies in the wedge w_i if and only if $\sigma_i \leqslant j < \tau_i$. (If $\sigma_i = \tau_i$, then no such diagonals exist, and note that the segment q_iq_j is not necessarily a diagonal even if $\sigma_i \leqslant j < \tau_i$.)

Consider now a fixed rotation s and a reflex vertex q_i of Q. The rotation s maps the vertex p_{i-s} onto q_i, so we consider the set of diagonals of T incident to p_{i-s}.

- If there is an index j such that T contains a diagonal $p_{i-s}p_{j-s}$ with $\sigma_i \leqslant j < \tau_i$, then we test if the segment q_iq_j lies in Q. If this is not the case, we can immediately exclude the rotation s from further consideration. If it exists, we mark the cell $G[s, i]$. Note that we do not need to test *all* indices j in the range $\sigma_i \leqslant j < \tau_i$.
- If no such index j exists, then we find the triangle $p_{i-s}p_{j-s}p_{k-s}$ in T with $k < \sigma_i < \tau_i \leqslant j$. We test if the segments q_iq_k, q_iq_j, and q_kq_j lie in Q. (They might be diagonals or they might simply be edges of Q.) We mark the cell $G[s, i]$ accordingly.

Clearly the procedure above can be implemented by binary search in the list of edges of T incident to p_{i-s}, taking $O(\log n)$ time for testing each pair (s, i), which leads to an $O(nr \log n)$ time algorithm.

To improve this to $O(nr)$, we *batch* these tests. We take the indices σ_i and τ_i over *all* reflex vertices q_i, and sort them into a single common list R of length $2r$. We now consider the vertices p_i of P in turn. For each vertex p_i, we merge the list of the ℓ_i diagonals incident to p_i with the list R in time $O(r + \ell_i)$.

We only need to walk through this list once to find the diagonals to be tested. Since $\sum_i \ell_i = O(n)$, this approach leads to the promised $O(n \log n + nr)$ algorithm.

As a final optimization, we observe that we do not actually need to construct the table G. It suffices to keep a counter for each rotation, increasing its value whenever we mark a cell in the corresponding row. In the end we report the rotations where the counter has reached the value r. ◀

4 Vertex correspondence is given

We are given two simple n-gons, whose vertices are already numbered 0 to $n-1$ in counterclockwise order.

We first compute, in $O(n^2)$ time, the visibility graphs of both P and Q [3], obtaining the set of diagonals (i, j) that exist in both P and Q. We have now abstracted away from the polygons P and Q and solve our problem using the following theorem.

▶ **Theorem 4.1.** *Given a graph G drawn in the plane with straight edges (possibly with crossings) whose vertices are the vertices of a convex n-gon C. We can determine in $O(n^\omega)$ time whether G contains a triangulation of C, and return such a triangulation if it exists.*

Proof. Let A be the adjacency matrix of G, and let B be the Boolean matrix where $b_{ij} = 1$ if the edge (i, j) exists and there is a triangulation of the subpolygon bounded by this edge and the chain from i to j. Our goal is to determine if $b_{0,n-1} = 1$.

We can express B recursively:

$$b_{ij} = \left(\bigvee_{i<k<j} (b_{ik} \wedge b_{kj}) \right) \wedge a_{ij}, \text{ for } 0 \leqslant i, j < n,\ j \geqslant i+1 \tag{1}$$

$$b_{i,i+1} = a_{i,i+1} (= 1), \text{ for } 0 \leqslant i < n-1 \tag{2}$$

Note that (2) is actually just a special case of (1) because the disjunction over the empty range $i < k < i+1$ evaluates to *true*. We set the undefined values b_{ij} for $j \leqslant i$ to 0. This results in a matrix $B = (b_{ij})$ that is strictly upper-diagonal.

Equations (1) and (2) can be written in terms of Boolean matrix multiplication:

$$B = (B \otimes B) \wedge A \tag{3}$$

Since (3) defines B in terms of itself, B cannot be *computed* straightforwardly by matrix multiplication. Nevertheless B is well-defined because it is strictly upper-diagonal. The usual (sequential) way to compute the recursions (1) is according to increasing length $j-i$, or rowwise from left to right, or columnwise from bottom to top. One has to ensure that the quantities on the right-hand side of (1) are computed before they are used.

We will in fact work over the integers, and define the following *triplet operation*:

$$\textit{update}(i,j,k)\colon \quad \hat{b}_{ij} := \hat{b}_{ij} + b_{ik} \cdot b_{kj} \tag{4}$$

For a fixed pair (i,j), we initialize $\hat{b}_{ij}$ to zero and perform the triplet operation for all k with $i<k<j$. Once all products have been accumulated, we convert $\hat{b}_{ij}$ to its final Boolean value $b_{ij} = a_{ij} \wedge \min\{\hat{b}_{ij}, 1\}$.

The key to subcubic running time is to perform the triplet operation in *blocks*. We assume that n is a power of two, and recursively partition the range of indices $0,\dots,n-1$ into blocks of decreasing size. For fixed indices $0 \leqslant i < k < j < n$, the triplet operation *update*(i,j,k) is performed at the largest block size S such that the three indices i, k, and j lie in different blocks. At this block size, we can perform the triplet operation for all indices within the same blocks using a single matrix multiplication:

$$\textit{block-update}(S,u,v,w)\colon \quad \hat{B}^S_{uv} := \hat{B}^S_{uv} + B^S_{uw} \cdot B^S_{wv} \tag{5}$$

Here, S is the block size, which is always a power of two, and B^S_{uv} is the $S \times S$ submatrix of entries b_{ij} formed from the rows in the range $uS \leqslant i < (u+1)S$ and the columns in the range $vS \leqslant j < (v+1)S$.

The operation *block-update*(S,u,v,w) is possible for $0 \leqslant u < w < v < n/S$ (for $w=u$ or $w=v$ it would define $\hat{B}^S_{uv}$ in terms of itself, or, more precisely, in terms of B^S_{uv}).

Our recursive algorithm is shown in Figure 3. It consists of a single procedure *ComputeBlock*, which we start as *ComputeBlock*$(n,0,0)$ after initializing the $n \times n$ matrix $\hat{B}$ to zeroes. The notations $\hat{b}_{ij}$ and b_{ij} in the algorithm refer to the same storage location. The notation b_{ij} is used to remind us that the variable takes only the values 0 and 1, while $\hat{b}_{ij}$ contains an arbitrary non-negative integer.

ComputeBlock performs recursive calls on up to four submatrices labeled α, β, γ, and δ, see Figure 4, using the recursive partitioning

$$B^S_{uv} = \begin{pmatrix} B^{S/2}_{2u,2v} & B^{S/2}_{2u,2v+1} \\ B^{S/2}_{2u+1,2v} & B^{S/2}_{2u+1,2v+1} \end{pmatrix} = \begin{pmatrix} \gamma & \delta \\ \alpha & \beta \end{pmatrix} \tag{6}$$

To prove correctness of the algorithm, we show that for each entry (i,j) of the matrix, we first compute the matrix product

$$\hat{b}_{ij} = \sum_{i<k<j} b_{ik} \cdot b_{kj},$$

over the integers. In the base case of the recursion, where the block size S is one, we replace $\hat{b}_{ij}$ by $b_{ij} \in \{0,1\}$—this is where we perform the logical 'and' operation with the adjacency matrix entry a_{ij}. At this point, we say that the entry (i,j) has then been *finalized*. It remains unchanged from now on.

Precondition 1. When *ComputeBlock*(S,u,v) is called, the entries $\hat{b}_{ij}$ in the block B^S_{uv} fulfill the equation

$$\hat{b}_{ij} = \sum_{\lfloor i/S \rfloor < \lfloor k/S \rfloor < \lfloor j/S \rfloor} b_{ik} \cdot b_{kj}.$$

ComputeBlock(S, u, v):
  if $S = 1$:// Finalize b_{uv}:
    if $v = u + 1$ **or** $(\hat{b}_{uv} > 0 \wedge a_{uv} = 1)$:
      $b_{uv} := 1$
    else
      $b_{uv} := 0$
  else if $u < v$: // compute the four submatrices $\alpha, \ldots, \delta$
    α: *ComputeBlock*$(S/2, 2u + 1, 2v)$
    β: *block-update*$(S/2, 2u + 1, 2v + 1, 2v)$
      ComputeBlock$(S/2, 2u + 1, 2v + 1)$
    γ: *block-update*$(S/2, 2u, 2v, 2u + 1)$
      ComputeBlock$(S/2, 2u, 2v)$
    δ: *block-update*$(S/2, 2u, 2v + 1, 2u + 1)$
      block-update$(S/2, 2u, 2v + 1, 2v)$
      ComputeBlock$(S/2, 2u, 2v + 1)$
  else // the diagonal case $u = v$ simplifies, because the submatrix α does not exist
    β: *ComputeBlock*$(S/2, 2u + 1, 2u + 1)$
    γ: *ComputeBlock*$(S/2, 2u, 2u)$
    δ: *ComputeBlock*$(S/2, 2u, 2u + 1)$

Figure 3 The recursive procedure *ComputeBlock*

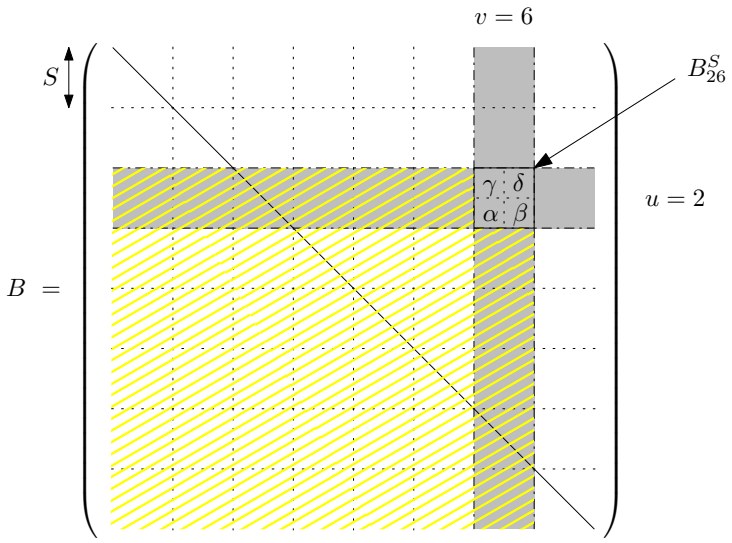


Figure 4 The matrix B is split into 8×8 square blocks of size $S = n/8$. The block $B^S_{uv} = B^S_{26}$ is shown to be composed of 4 subblocks of size $S/2$, labeled $\alpha, \beta, \gamma, \delta$ in the order in which they are recursively computed in the procedure *ComputeBlock*.

In other words, all triplet operations $update(i, j, k)$ for which k lies in a block of size S different from i and j have already been performed.

Precondition 2. When $ComputeBlock(S, u, v)$ is called, all entries b_{ij} with $i \geqslant uS$ and $j < (v+1)S$ have already been finalized, except for the block B^S_{uv} itself. (This is the hatched region in Figure 4.)

Postcondition. $ComputeBlock(S, u, v)$ finalizes the block B^S_{uv}. Together with Precondition 2, this means that all entries b_{ij} with $i \geqslant uS$ and $j < (v+1)S$ have been finalized.

Correctness of the algorithm can be established with the help of these preconditions and the postcondition. The proof is given in the full paper [1].

Runtime. We let $T(S)$ denote the running time of *ComputeBlock* for a block of size S. Clearly $T(1) = O(1)$ and $T(S) = O(S^\omega) + 4T(S/2)$ for $s > 1$, leading to $T(n) = O(n^\omega)$, assuming $\omega > 2$. ◀

▶ **Theorem 4.2.** *Given simple n-gons P and Q with a fixed vertex numbering on both polygons, we can decide in $O(n^\omega)$ time if there are compatible triangulations for P and Q.*

Counting triangulations

If we work with the integer entries $\hat{b}_{ij}$ as they are, only setting them to 0 when $a_{ij} = 0$, but otherwise keeping their values intact, the result will be the number of triangulations. The numbers can be as large as nearly 4^n; hence the bit-size of the numbers must then be taken into account in the analysis.

In the full paper [1] we discuss how our method relates to the word problem for context-free grammars.

Can we do better?

▶ **Lemma 4.3.** *The matrix B as defined above cannot be computed faster than Boolean matrix multiplication.*

See the full paper [1] for the proof. This statement implies that a faster algorithm can only be obtained by exploiting the fact that the graph G is not an arbitrary graph, but the intersection of the visibility graphs of two simple polygons.

References

1 Peyman Afshani, Boris Aronov, Kevin Buchin, Maike Buchin, Otfried Cheong, Katharina Klost, Carolin Rehs, and Günter Rote. Compatible triangulations of simple polygons, 2026. arXiv:2603.01282.

2 Boris Aronov, Raimund Seidel, and Diane Souvaine. On compatible triangulations of simple polygons. *Computational Geometry*, 3:27–35, 1993. doi:10.1016/0925-7721(93)90028-5.

3 Takao Asano, Tetsuo Asano, Leonidas J. Guibas, John Hershberger, and Hiroshi Imai. Visibility of disjoint polygons. *Algorithmica*, 1(1):49–63, 1986. doi:10.1007/BF01840436.

4 Anna Lubiw and Debajyoti Mondal. On compatible triangulations with a minimum number of Steiner points. *Theoretical Computer Science*, 835:97–107, 2020. doi:10.1016/j.tcs.2020.06.014.

Simple Topological Thickness*

Michael Hoffmann[1], Julia Oppermann[2], Rosna Paul[2], Jonathan Rollin[2], and Alexandra Weinberger[2]

1 **Department of Computer Science, ETH Zürich, Switzerland**
`hoffmann@inf.ethz.ch`
2 **FernUniversität in Hagen, Germany**
`firstname.lastname@fernuni-hagen.de`

Abstract

We introduce the notion of the simple topological thickness of a graph G as the smallest number of colors required to produce an edge-colored simple drawing of G without monochromatic crossings. A drawing is simple if adjacent edges do not cross and any two independent edges cross at most once. We show that simple topological thickness is different from geometric thickness and (graph) thickness. In particular, we show that the simple topological thickness of 2-degenerate graphs is at most two and there exists a biplanar graph with simple topological thickness at least three.

1 Introduction

We study edge partitions of graphs into planar subgraphs. This topic has a long history that originated from the study of *biplanar* graphs [13], that is, graphs whose edges can be partitioned into two sets that each induce a planar graph. This later developed into various notions of graph thickness and related concepts. Naturally, one may consider the subgraphs not drawn individually, but rather as a partition of a common drawing of the whole graph. This common drawing might then even need to comply with additional global drawing constraints. This can be seen as a layered drawing, a frequent interpretation of thickness in the literature, and also a useful framework in various applications.

Throughout this work, we treat edge partitions as edge-colorings. A *monochromatic crossing* is a crossing between edges of the same color. The standard definition of *thickness* [25] of a graph G, denoted by $\theta(G)$, is the smallest number of colors needed to color the edges of G such that each color class forms a planar subgraph of G. By a result of Pach and Wenger [23, 24] we can fix arbitrary vertex locations for the whole graph and draw each color class individually without crossings on these vertex locations. In this way we obtain a drawing of the whole graph G without monochromatic crossings using only thickness-many colors. So, an equivalent definition of thickness of a graph G is the smallest number of colors required to produce an edge-colored drawing of G without monochromatic crossings. Adding the global constraint that only straight edges are allowed, yields the notion of *geometric thickness* $\overline{\theta}(G)$ [16], which is the smallest number of colors required to produce an edge-colored straight-line drawing of G without monochromatic crossings. Many other constraints have been considered; for instance, the book thickness $\mathrm{bt}(G)$ [17, 22] imposes the additional global constraint that the vertices must be placed in convex position. For historic notes and for specific bounds and values for several graph classes we refer to the survey by Mutzel, Odenthal and Scharbordt [21] and for related thickness parameters to a handbook chapter by Bläsius, Kobourov and Rutter [4]. Computing thickness [19], geometric thickness [6, 7],

* We thank André Schulz for helpful discussions.

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

and book thickness [5] are also known to be NP-hard problems. Moreover, it is $\exists\,\mathbb{R}$-complete to decide if a given multigraph has geometric thickness k for $k \geq 30$ [12].

We introduce a parameter called *simple topological thickness* of a graph G, denoted by $\widetilde{\theta}(G)$, which is the smallest number of colors required to produce an edge-colored simple drawing (also known as a simple topological graph) of G without monochromatic crossings. Simple drawings appear frequently in the literature, where a drawing is *simple* if adjacent edges do not cross and any two independent edges cross at most once. A *simple biplane* drawing is an edge-colored simple drawing with two color classes that each induce a plane drawing. Note that dropping the simplicity constraint yields the usual notion of thickness. From the definitions, it follows that for any graph G, the inequalities $\theta(G) \leq \widetilde{\theta}(G) \leq \overline{\theta}(G) \leq \mathrm{bt}(G)$ hold. Eppstein demonstrated that $\theta(G)$, $\overline{\theta}(G)$, and $\mathrm{bt}(G)$ can differ arbitrarily using d-degenerate graphs, that is, graphs where every subgraph contains a vertex of degree at most d. In particular, for any integer k, there exists a 2-degenerate graph with geometric thickness two and book thickness at least k [8], as well as a 3-degenerate graph with thickness three and geometric thickness at least k [9, 10]. The latter result implies that bends or curved edges are sometimes necessary to avoid monochromatic crossings when using only $\theta(G)$ many colors.

Results. We show that the inequalities $\theta(G) \leq \widetilde{\theta}(G)$ and $\widetilde{\theta}(G) \leq \overline{\theta}(G)$ can be strict. For any 2-degenerate graph G, we show that $\widetilde{\theta}(G) \leq 2$ (Theorem 1) in Section 2; while 2-degenerate graphs G with $\overline{\theta}(G) \geq 3$ are known to exist [14, 15]. We describe a specific graph G with $\theta(G) = 2$ and $\widetilde{\theta}(G) = 3$ (Theorem 2) in Section 3.

2 Simple Topological Thickness of Two-Degenerate Graphs

In this section, we present an upper bound of two on the simple topological thickness of 2-degenerate graphs. This bound is best possible in general, as there exist 2-degenerate graphs with (graph) thickness two, such as subdivisions of K_n, for $n \geq 5$.

▶ **Theorem 1.** *For every 2-degenerate graph G we have $\widetilde{\theta}(G) \leq 2$.*

Proof Sketch. We insert the vertices one by one in an order $v_1, \ldots, v_n$ where every v_i has at most two neighbors in $V_i^- = \{v_1, \ldots, v_{i-1}\}$. We may assume that v_i has exactly two such neighbors for $i \geq 3$. We construct a simple drawing Γ and a partition of the edge set into a *red* forest R and a *blue* forest B such that both $\Gamma[B]$ and $\Gamma[R]$ are plane. When inserting v_i, for $i \geq 3$, we select one of its neighbors in V_i^- to be its blue parent bp_i and the other to be its red parent rp_i. Correspondingly, we draw a blue edge $b_i = \mathrm{bp}_i v_i$ and a red edge $r_i = \mathrm{rp}_i v_i$. We describe Γ as a drawing on a cylinder C, represented as a vertical strip S whose bounding lines are identified. To route the red edges we use a set $\mathcal{L}$ of n parallel lines that wind around C. Every vertex v is associated to a unique line $\ell(v) \in \mathcal{L}$ and placed on $\ell(v)$. This $\ell(v)$ is inserted into $\mathcal{L}$ at a specific position when v is inserted into Γ. The ray along $\ell(v)$ that leaves v to the right is the *red channel* of v. In S, each line of $\mathcal{L}$ appears as a set of parallel segments. See Figure 1 for an example.

We start with a single line $\mathcal{L} = \{\ell_1\}$ and place v_1 arbitrarily on the bottommost segment of ℓ_1 in S. Then we add another line ℓ_2 above ℓ_1 and place v_2 on the bottommost segment of ℓ_2 above and slightly to the left of v_1, and draw v_1v_2 as a blue straight-line segment.

Each vertex v_i, for $i \geq 3$, is inserted as follows. We select the lower of the two parents of v_i to be rp_i and the higher one to be bp_i, where height is measured by the sequence of segments of lines from $\mathcal{L}$ on S. We insert ℓ_i immediately above $\ell(\mathrm{rp}_i)$ and place v_i on the lowest segment of ℓ_i that passes above bp_i. Then v_i is placed on this segment almost

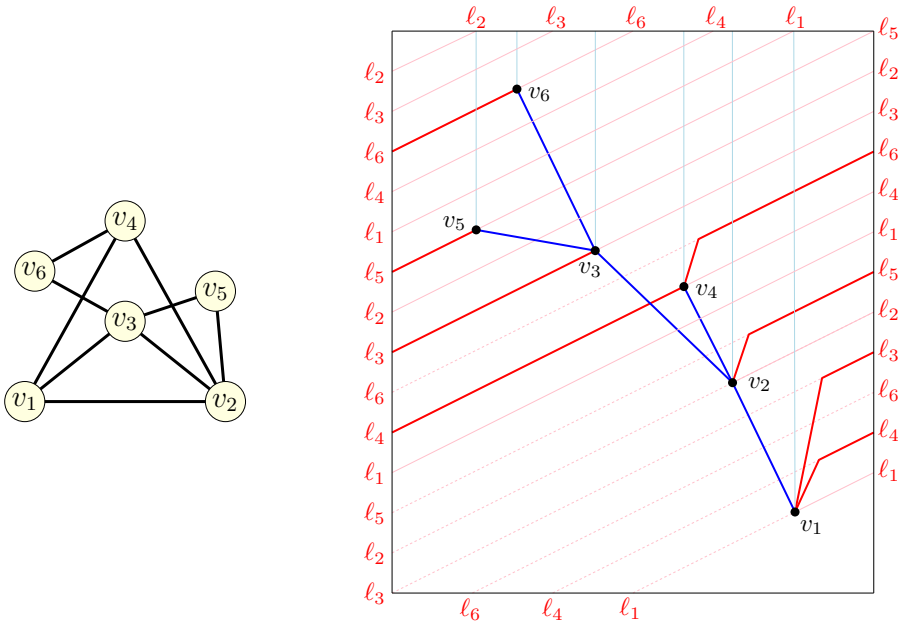


Figure 1 Drawing of a 2-degenerate graph on $n = 6$ vertices on the lateral surface area of a cylinder C. We represent C as a vertical strip S whose left and right bounding segments are identified. The n parallel lines in $\mathcal{L}$ wrap around S, which partitions each line into many parallel segments.

vertically above bp_i, slightly to the left but sufficiently close to the blue channel of bp_i, that is, the vertical upward ray emanating from bp_i. In this way, we can ensure that the new blue edge b_i does not cross any existing blue edge or channel. The new red edge r_i is drawn along ℓ_i, except for a small neighborhood of rp_i, where it connects $\ell(\mathrm{rp}_i)$ and ℓ_i by an almost-vertical segment of positive slope. In this way, we can ensure that r_i does not cross any red edge or channel and all of its crossings are on the segment of r_i drawn along ℓ_i.

Then it is not hard to see that both the red and the blue subdrawing are plane. To show that the drawing is simple, we rely on the following key property of the drawing constructed: all blue edges are *short*, that is, every blue edge spans at most $n-1$ consecutive segments of lines from $\mathcal{L}$ in S. In particular, as any two segments of the same line $\ell \in \mathcal{L}$ are at least n segments apart in S, a blue edge $v_j v_i$ crosses each line from $\mathcal{L}$ at most once. ◀

In the proof of Theorem 1 we construct a simple biplane drawing with additional properties: Edges of one color class are drawn as straight-line segments, and the edges of the other color class satisfy some monotonicity condition (they are spirals with a common center). Requiring both color classes to be drawn as straight-line segments is impossible in general because there exist 2-degenerate graphs with geometric thickness three [14, 15]. In the next section, we will see that not every biplanar graph can be realized as a simple biplane drawing.

3 Separating Thickness from Simple Topological Thickness

Our first aim is to show that there exist biplanar graphs for which the decomposition into two planar subgraphs is unique. For a graph G we denote by $\mathrm{V}(G)$ the set of vertices and by $\mathrm{E}(G)$ the set of edges. A graph $H = (A, B)$ is a subgraph of a graph $G = (V, E)$,

denoted by $H \leq G$, if $A \subseteq V$ and $E \subseteq B$. Let G be a *maximum biplanar* graph on $n \geq 3$ vertices, that is, G is the edge-disjoint union of two maximal planar graphs $T_1 = (V, E_1)$ and $T_2 = (V, E_2)$ on the same vertex set V. As T_1 and T_2 are triconnected, their facial triangles are uniquely determined, independently from any drawing. Let F_1 and F_2 denote the sets of facial triangles of T_1 and T_2, respectively. The *disjointness graph* $D(T_1, T_2)$ is a bipartite graph that connects the faces of T_1 and T_2. Its vertex set is the disjoint union of F_1 and F_2 (if a triangle is a face in both graphs, then it appears twice in $D(T_1, T_2)$), and two triangles $s \in F_1$ and $t \in F_2$ are adjacent in $D(T_1, T_2)$ if $\mathrm{V}(s) \cap \mathrm{V}(t) = \emptyset$. An edge st of $D(T_1, T_2)$ is *long* if there exist vertices $u \in \mathrm{V}(s)$ and $v \in \mathrm{V}(t)$ such that $uv \notin \mathrm{E}(G)$. The following lemma shows that uniquely decomposable biplanar graphs exist.

▶ **Lemma 1.** *Let G be a maximum biplanar graph that is the edge-disjoint union of two maximal planar graphs T_1 and T_2 whose disjointness graph has a Hamiltonian cycle with at least one long edge. Then there exists a maximum biplanar graph $G^+ \geq G$ that uniquely decomposes into two maximal planar graphs $T_1^+ \geq T_1$ and $T_2^+ \geq T_2$.*

Proof Sketch. To obtain G^+ we add the Hamiltonian cycle $s_1, \ldots, s_{4n-8}$ from $D(T_1, T_2)$ to G using new vertices $v_1, \ldots, v_{4n-8}$, where $n = |V(G)|$. We then add, for each i, certain edges between the (vertices of the) triangle s_i and $\{v_{i-1}, v_i\}$, so that s_i becomes triangulated when v_{i-1} and v_i are considered interior to s_i. This extends T_1 and T_2 to two edge-disjoint triangulations whose union is G^+. The key idea to make this decomposition unique, is a careful choice of the new edges so that most edges lie in exactly two triangles in G^+. Then both triangles must be in the same triangulation. Some arguments over the vertex degrees show that there is a unique distribution of the triangle pairs to the two triangulations. ◀

3.1 The Double Icosahedron

The double icosahedron DI is a 10-regular graph on twelve vertices that is isomorphic to K_{12} minus a perfect matching. The name reflects the fact that the graph can be formed as an edge-disjoint union of the graphs of two icosahedra. Its crossing behavior was previously studied by several authors [11, 20]. Denoting its vertex set by $V = \{0, \ldots, 11\}$, this decomposition (I_1, I_2) is depicted in Fig. 2.

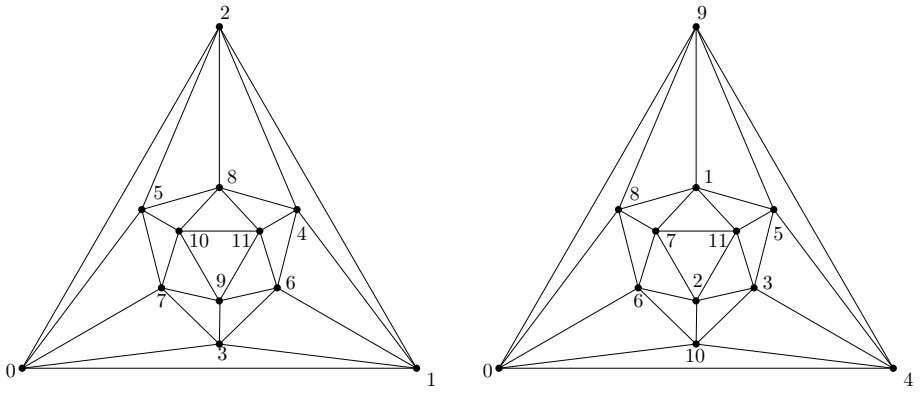


Figure 2 The double icosahedron is the edge-disjoint union of these two icosahedra I_1 and I_2.

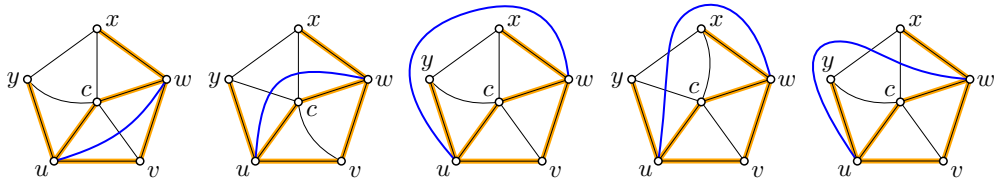
Clearly, the graph DI is biplanar. The perfect matching of non-edges in DI is formed by the antipodal pairs of vertices whose indices sum up to 11. The *antipodal map* $\tau \colon V \to V$ maps the vertices of each antipodal pair to each other, i.e., $\tau(i) = 11 - i$. One icosahedron

can be obtained from the other by flipping all edges. That is, each edge uv is replaced by the edge $u'v'$, where $uu'vv'$ form the quadrilateral given by the two triangles sharing uv. The decomposition of DI into two icosahedra is unique. However, there are many different ways to decompose DI into two planar graphs. For instance, we could exchange the two edge pairs $0, 10$ and $1, 11$ with $4, 6$ and $5, 7$ to obtain a different decomposition. Therefore, we will not use DI directly but first apply Lemma 1.

▶ **Theorem 2.** *There exists a biplanar graph that does not admit a simple biplane drawing.*

Proof Sketch. The disjointness graph $D(I_1, I_2)$ has a Hamiltonian cycle with several long edges. By Lemma 1, we obtain a graph $G \geq \mathrm{DI}$ that uniquely decomposes into two planar graphs T_1 and T_2 such that $T_1 \geq I_1$ and $T_2 \geq I_2$. So, any simple biplane drawing of G contains a simple biplane drawing Γ of (I_1, I_2). We will argue that no such drawing Γ exists.

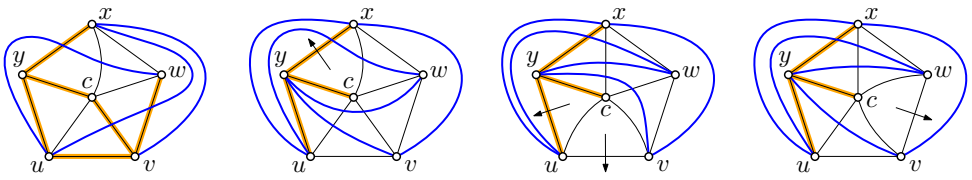
Consider a vertex c. Its neighborhood in I_1 forms a cycle $C_1(c) = (u, v, w, x, y)$, so c and this cycle form a so-called wheel. In I_2 the vertices of $C_1(c)$ also form a cycle $C_2(\tau(c)) = (u, w, y, v, x)$. By simplicity of Γ, and due to no edges of the same color crossing, we conclude that there are only five options to route the edge uw in Γ, which are shown in Fig. 3.



(a) direct **(b)** indirect **(c)** outer **(d)** switch **(e)** switch

Figure 3 The five different options to route the edge uw of I_2 in Γ, relative to the neighborhood of c in Γ. Adjacent edges, which must not be crossed, are shown bold and orange.

By analyzing these five options, we show that the cycle $C_2(\tau(c))$ uses at least one switch edge and without loss of generality we assume that this switch edge uw is drawn as in Fig. 3e. This forces vx to be an outer edge, and ux to be either an outer or a switch edge in $C_2(\tau(c))$ (see Fig. 4). Further investigations on how the cycle $C_2(\tau(c))$ can be drawn, lead to the observation that there are always adjacent vertices c and v in I_1 such that each of the cycles $C_2(c)$ and $C_2(v)$ are drawn in Γ as in Fig. 4c. From this configuration we can conclude that there is no valid position for $\tau(v)$. Hence, no simple biplane drawing of G exists.



(a) **(b)** **(c)** **(d)**

Figure 4 Edges of a cycle $C_2(\tau(c))$ in Γ, up to cyclic relabeling and reflection.

◀

4 Conclusion and open problems

We have separated simple topological thickness from geometric thickness and graph thickness by showing that there exists graphs G for which $\widetilde{\theta}(G) < \overline{\theta}(G)$ and a graph G' for which $\widetilde{\theta}(G') > \theta(G')$. The natural next step is to ask how large the difference can be.

▶ Open Problem 1. Are there constants c and c' such that for every k there exists a graph G with $\overline{\theta}(G) = k$ and $\widetilde{\theta}(G) = c$ and a graph G' with $\widetilde{\theta}(G') = k$ and $\theta(G) = c'$?

Another natural research direction is to study thickness parameters for other global drawing constraints (drawing families) and their relationship with established thickness parameters. Promising starting points might be given by the so-called convexity hierarchy [3]. Another drawing family are simple drawings that can be extended to simple drawings of complete graphs. As opposed to straight-line drawings or general drawings, there exist simple drawings that are not subdrawings of any simple drawing of the complete graph [18]. In fact, even asking whether a single edge can be added to a given simple drawing is NP-hard [1, 2].

We define *completable simple topological thickness* of a graph G, denoted by $\overline{\widetilde{\theta}}(G)$, as the smallest integer k for which there exists a simple drawing of a complete graph that contains a subdrawing D of G such that the edges of D can be colored with k many colors avoiding monochromatic crossings (among the edges of G). Clearly, $\widetilde{\theta}(G) \leq \overline{\widetilde{\theta}}(G) \leq \overline{\theta}(G)$, as geometric drawings always can be extended to simple (even geometric) drawings of a complete graph. We think that our construction in Section 2 in fact can be extended to a simple drawing of a complete graph, which would imply $\widetilde{\theta}(G) = \overline{\widetilde{\theta}}(G)$ for every 2-degenerate graph G.

▶ Open Problem 2. Does $\widetilde{\theta}(G) = \overline{\widetilde{\theta}}(G)$ hold for all graphs G?

References

1 Alan Arroyo, Fabian Klute, Irene Parada, Raimund Seidel, Birgit Vogtenhuber, and Tilo Wiedera. Inserting one edge into a simple drawing is hard. In Isolde Adler and Haiko Müller, editors, *WG 2020*, volume 12301 of *LNCS*, pages 325–338. Springer, 2020. doi:10.1007/978-3-030-60440-0_26.

2 Alan Arroyo, Fabian Klute, Irene Parada, Birgit Vogtenhuber, Raimund Seidel, and Tilo Wiedera. Inserting one edge into a simple drawing is hard. *Discret. Comput. Geom.*, 69(3):745–770, 2023. doi:10.1007/S00454-022-00394-9.

3 Alan Arroyo, Dan McQuillan, R. Bruce Richter, and Gelasio Salazar. Convex drawings of the complete graph: topology meets geometry. *Ars Math. Contemp.*, 22(3):3, 2022. doi:10.26493/1855-3974.2134.AC9.

4 Thomas Bläsius, Stephen G. Kobourov, and Ignaz Rutter. Simultaneous embedding of planar graphs. In Roberto Tamassia, editor, *Handbook on Graph Drawing and Visualization*, pages 349–381. Chapman and Hall/CRC, 2013. doi:10.1201/b15385.

5 Fan R. K. Chung, Frank T. Leighton, and Arnold L. Rosenberg. Embedding graphs in books: A layout problem with applications to VLSI design. *SIAM J. Algebr. Discrete Methods*, 8(1):33–58, 1987. doi:10.1137/0608002.

6 Stephane Durocher, Ellen Gethner, and Debajyoti Mondal. Thickness and colorability of geometric graphs. In Andreas Brandstädt, Klaus Jansen, and Rüdiger Reischuk, editors, *WG 2013*, volume 8165 of *LNCS*, pages 237–248. Springer, 2013. doi:10.1007/978-3-642-45043-3_21.

7 Stephane Durocher, Ellen Gethner, and Debajyoti Mondal. Thickness and colorability of geometric graphs. *Comput. Geom.*, 56:1–18, 2016. doi:10.1016/j.comgeo.2016.03.003.

8 David Eppstein. Separating geometric thickness from book thickness. *arXiv preprint math/0109195*, 2001. doi:10.48550/arXiv.math/0109195.

9 David Eppstein. Separating thickness from geometric thickness. In Stephen G. Kobourov and Michael T. Goodrich, editors, *GD 2002*, volume 2528 of *LNCS*, pages 150–161. Springer, 2002. doi:10.1007/3-540-36151-0_15.

10 David Eppstein. Separating thickness from geometric thickness. *Contemp. Math.*, 342:75–86, 2004. doi:10.1090/conm/342.

11 Stefan Felsner, Michael Hoffmann, Tomas Hruz, Tillmann Miltzow, Hannah Rotzoll, Shengzhe Wang, and Emo Welzl. On local and global crossing numbers. In Jan Kratochvíl and Giuseppe Liotta, editors, *EuroCG 2025*, pages 42:1–8, 2025. URL: https://kam.mff.cuni.cz/conferences/eurocg2025/booklet2025.pdf.

12 Henry Förster, Philipp Kindermann, Tillmann Miltzow, Irene Parada, Soeren Terziadis, and Birgit Vogtenhuber. Geometric thickness of multigraphs is-complete. *Algorithmica*, 88(1):3, 2026. doi:10.1007/S00453-025-01351-7.

13 Frank Harary. Research problem. *Bull. Amer. Math. Soc.*, 67(6):542, 1961. doi:10.1090/S0002-9904-1961-10677-0.

14 Rahul Jain, Marco Ricci, Jonathan Rollin, and André Schulz. On the geometric thickness of 2-degenerate graphs. In *SoCG 2023*, volume 258 of *LIPIcs*, pages 44:1–44:15, 2023. doi:10.4230/LIPIcs.SoCG.2023.44.

15 Rahul Jain, Marco Ricci, Jonathan Rollin, and André Schulz. On the geometric thickness of 2-degenerate graphs. *J. Comput. Geom.*, 15(2):94–123, 2024. doi:10.20382/jocg.v15i2a4.

16 Paul C. Kainen. Thickness and coarseness of graphs. In *Abh. Math. Semin. Univ. Hambg.*, volume 39, pages 88–95. Springer, 1973. doi:10.1007/BF02992822.

17 Paul C. Kainen. Some recent results in topological graph theory. In Ruth A. Bari and Frank Harary, editors, *Graphs and Combinatorics*, volume 406 of *LNM*, pages 76–108, 1974. doi:10.1007/BFb0066436.

18 Jan Kyncl. Improved enumeration of simple topological graphs. *Discret. Comput. Geom.*, 50(3):727–770, 2013. doi:10.1007/S00454-013-9535-8.

19 Anthony Mansfield. Determining the thickness of graphs is NP-hard. *Math. Proc. Cambridge Philos. Soc.*, 93(1):9–23, 1983. doi:10.1017/S030500410006028X.

20 Bojan Mohar. On a conjecture by Anthony Hill. *arXiv preprint 2009.03418*, 2020. doi:10.48550/arXiv.2009.03418.

21 Petra Mutzel, Thomas Odenthal, and Mark Scharbrodt. The thickness of graphs: A survey. *Graphs and Combinatorics*, 14:59–73, 1998. doi:10.1007/PL00007219.

22 L. Taylor Ollmann. On the book thicknesses of various graphs. In Frederick Hoffman, Roy B. Levow, and Robert S. D. Thomas, editors, *Fourth Southeastern Conference on Combinatorics, Graph Theory and Computing*, volume VIII of *Congressus Numerantium*, page 459, 1973.

23 János Pach and Rephael Wenger. Embedding planar graphs at fixed vertex locations. In Sue Whitesides, editor, *GD 1998*, volume 1547 of *LNCS*, pages 263–274. Springer, 1998. doi:10.1007/3-540-37623-2_20.

24 János Pach and Rephael Wenger. Embedding planar graphs at fixed vertex locations. *Graphs and Combinatorics*, 17(4):717–728, 2001. doi:10.1007/PL00007258.

25 W. T. Tutte. The thickness of a graph. *Indag. Math.*, 25:567–577, 1963. doi:10.1016/S1385-7258(63)50055-9.

On the Pathwidth of 2-Layer k-Matching-Planar Graphs

Saeed Odak[1], Jonathan Rollin[2], and Torben Scheele[3]

1 **Aalto University, Finland**
`saeed.odak@aalto.fi`
2 **FernUniversität in Hagen, Germany**
`jonathan.rollin@fernuni-hagen.de`
3 **TU Dortmund, Germany**
`torben.scheele@tu-dortmund.de`

Abstract

A graph drawing is *k-matching-planar*, if for each edge e the largest matching among the edges crossing e has size at most k. In a 2-layer drawing of a bipartite graph the vertices of one set of the bipartition are placed along a common horizontal line while the vertices of the other set are placed on another horizontal line. The edges are drawn as straightline segments in between. We prove that graphs with a 2-layer k-matching-planar drawing have pathwidth at most $2k + 1$ and describe, for each $k \geq 1$, a graph with a 2-layer k-matching-planar drawing and pathwidth $3\lfloor k/2 \rfloor + 1$.

1 Introduction

Hierarchical data relationships are often presented as layered graph drawings, where the vertices are placed on several consecutive horizontal lines. An important case are 2-*layer drawings* of bipartite graphs. In such drawings, the vertices of one set of the bipartition are placed along a common horizontal line while the vertices of the other set are placed on another horizontal line. The edges are drawn as straightline segments in between. A typical drawing objective, to improve readability, is the minimization of the number of crossings between the edges, which has been studied intensively [6, 7, 8, 10]. More recently, the local version of minimizing the maximum number of crossings per edge was studied [1, 9, 14]. Ideally, there are no crossings which is possible if and only if the graph is a caterpillar forest (an acyclic graph where the vertices of degree greater than one form a collection of paths, cf. Wood [19]). These graphs are exactly the graphs of pathwidth at most 1.

A *path decomposition* is defined as a sequence $B_1, \ldots, B_t \subseteq V(G)$, so that the following conditions hold:

1. Each vertex is in some bag: $\bigcup_{1 \leq i \leq t} B_i = V$.
2. Each edge is in some bag: $\bigcup_{1 \leq i \leq t} E(i) = E$ where $E(i) = \{\{u, v\} \in E \mid u, v \in B_i\}$.
3. Each vertex's bags induce a subpath: $B_i \cap B_k \subseteq B_j$ for $1 \leq i \leq j \leq k \leq t$.

The *width* of a path decomposition is defined as $\max_{1 \leq i \leq t} |B_i| - 1$. The *pathwidth* $\mathrm{pw}(G)$ of G is the minimum width over all path decompositions of G.

Pathwidth is an important parameter in graph structure theory and parameterized complexity [3, 4, 17], with many relations to graph drawing [5, 12, 18, 20]. Wood [19] established a tight relation between 2-layer drawings of bipartite graphs and their pathwidth: A class $\mathcal{G}$ of bipartite graphs has bounded pathwidth if and only if there exists p, q, $r \in \mathbb{N}$ such that every graph in $\mathcal{G}$ has a 2-layer drawing with no p-crossing and no (q, r)-crossing. Here, a p-crossing is formed by p pairwise crossing edges and a (q, r)-crossing is formed by two non-crossing matchings M_1 and M_2 of size q and r, respectively, such that each edge from

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

M_1 crosses each edge from M_2. Better bounds on the pathwidth for specific (beyond planar) classes of of 2-layer drawings are known. For instance, graphs with a 2-layer drawing where for each edge e all edges crossing e have a common endpoint, called *fan-planar* drawings, have pathwidth at most 2 [2, Corollary 3].[1] Graphs with a 2-layer drawing where each edge is crossed at most k times, called k-*planar* drawings, have pathwidth at most $k+1$ [1, 16]; both bounds are tight. Hendrey, Karol and Wood [11] recently introduced a common generalization of fan-planarity and k-planarity: A drawing is k-*matching-planar*, if for each edge e the largest matching among the edges crossing e has size at most k. The same concept was independently introduced by Merker et al. [15] under the name k-independent crossing drawings. Note that, for 2-layer or straightline drawings, the 1-matching-planar drawings are exactly the fan-planar drawings and that any k-planar drawing is k-matching-planar. The results by Wood [19] imply that bipartite graphs with a 2-layer k-matching-planar drawing have pathwidth in $O(k^3)$. We improve this bound by showing that every graph with a 2-layer k-matching-planar drawing has a path decomposition with width at most $2k+1$. We also provide a close lower bound of $3\lfloor k/2 \rfloor + 1$ for the pathwidth of certain graphs with a 2-layer k-matching-planar drawing.

Let us remark that no converse statements to our results hold: for each k any sufficiently large 1-subdivided star has pathwidth 2 and is not 2-layer k-matching-planar. Wood [19] observed a similar statement with respect to 2-layer k-planarity.

2 The Upper Bound

In this section we prove the upper bound on the pathwidth.

▶ **Theorem 2.1.** *The pathwidth of any graph with a 2-layer k-matching-planar drawing is at most $2k+1$.*

Proof. Let Γ be a 2-layer k-matching-planar drawing of some graph G. Without loss of generality, we assume that Γ is maximal (so any new straight edge between non-adjacent vertices would violate k-matching-planarity). Let H denote an edge-maximal plane subgraph in the drawing Γ which contains the edge between the two leftmost vertices, contains the edge between the two rightmost vertices, and contains no isolated vertices. These edges on the left and right boundary exist due to the maximality of Γ. Observe that H consists of entirely disjoint caterpillars. In the 2-layer drawing Γ, if a vertex u appears to the left (resp. right) of a vertex v on the same horizontal line Γ, we say u is left (resp. right) of v. Let $e_1, \ldots, e_m$ denote the edges in H from left to right, that is, uv is before $u'v'$ if either v is left of v' or $v = v'$ and u is left of u'. Let S_i denote the set of edges crossing e_i in Γ. We will choose a special vertex cover for each S_i as follows. Let L_i denote the set of vertices to the left of e_i and let R_i denote the set of vertices to the right of e_i (both excluding the endpoints of e_i). Then $(L_i \cup R_i, S_i)$ is a bipartite graph. Let $\mathcal{F}_i$ denote the family of all minimum vertex covers of this graph. We define a *left-Kőnig-cover* as the set C_i that consists of all vertices from L_i that are contained in the union of vertex covers in $\mathcal{F}_i$ and of all vertices from R_i that are contained in the intersection of vertex covers in $\mathcal{F}_i$. That is

$$C_i = \left(\bigcup_{C \in \mathcal{F}_i} L_i \cap C \right) \cup \left(\bigcap_{C \in \mathcal{F}_i} R_i \cap C \right).$$

[1] Layered drawings where each edge has its endpoints on adjacent layers are called proper in [2].

Jeong, Sæther, and Telle [13, Corollary 3.3. and Lemma 3.4] proved the following properties.[2]

▶ **Lemma 2.2** ([13]). ***(I)*** *For each i, C_i is a minimum vertex cover of S_i.*
(II) *For each i, $1 \leq i < m$, we have $C_i \cap L_i \supseteq C_{i+1} \cap L_i$ and $C_i \cap R_{i+1} \subseteq C_{i+1} \cap R_{i+1}$.*

As Γ is k-matching-planar, Lemma 2.2 (I) implies that C_i has size at most k. Now we are ready to describe a path-decomposition of G.

- Let $B_1^R = e_1 \cup C_1 \cup C_2$ and let $B_m^L = e_m \cup C_m \cup C_{m-1}$.
- For each i, $1 < i < m$, let $B_i^L = e_i \cup C_i \cup C_{i-1}$ and let $B_i^R = e_i \cup C_i \cup C_{i+1}$.
- For each i, $1 \leq i < m$, let $u_{i,1}, \ldots, u_{i,t_i}$ denote the vertices between e_i and e_{i+1} in an arbitrary order. For each j, $1 \leq j \leq t_i$, let $B_{i,j} = \{u_{i,j}\} \cup (e_i \cap e_{i+1}) \cup C_i \cup C_{i+1}$.

Each bag B_i^L, B_i^R, and $B_{i,j}$ has size at most $2k + 2$. We claim that

$$B_1^R, B_{1,1}, \ldots, B_{1,t_1}, B_2^L, B_2^R B_{2,1} \ldots B_{2,t_2}, \ldots, B_m^L$$

is a path decomposition of G.

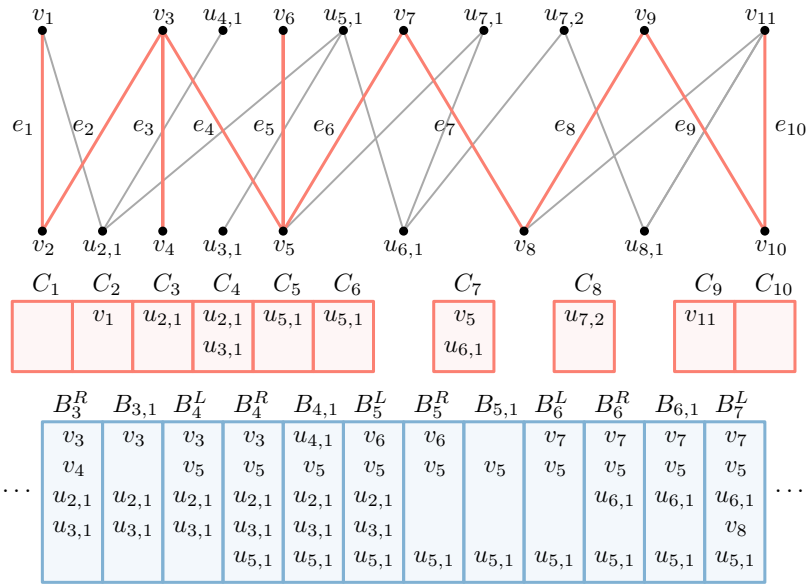


Figure 1 A 2-layer 2-matching-planar drawing with an edge-maximal plane subgraph (red) as well as the left-Kőnig-cover for its edges and the resulting path decomposition of width ≤ 5.

Each vertex in some bag: Due to the technical assumptions on H, each vertex from G is either incident to an edge from H or between two edges from H. Each vertex incident to an edge from H is in some bag B_i^L or B_i^R and any other vertex is in some bag $B_{i,j}$.

Each edge in some bag: We have $e_1 \subseteq B_1^R$, $e_m \subseteq B_m^L$ and, for $1 < i < m$, $e_i \subseteq B_i^L$. Any other edge uv crosses some edge from H, as H is edge-maximal plane. Let e_ℓ and e_r denote the leftmost and rightmost edges from H crossed by uv. Then uv crosses each edge e_i in H with $\ell \leq i \leq r$. For each i, $\ell \leq i \leq r$, uv is in S_i and so the left-Kőnig cover

[2] In their notation C_i is an L_i-Kőnig cover of the bipartite graph $(L_i \cup R_i, S_i)$.

C_i of S_i contains at least one of u or v. Assume that u is to the left of e_ℓ. If $v \in C_\ell$ and u is an endpoint of $e_{\ell-1}$, then $u, v \in B^R_{\ell-1}$. If $v \in C_\ell$ and u is between $e_{\ell-1}$ and e_ℓ, then $u, v \in B_{\ell-1,j}$ for some j. Similarly, If $u \in C_r$, then $u, v \in B^L_{r+1}$ or $u, v \in B_{r,j}$ for some j. If neither $v \in C_\ell$ nor $u \in C_r$, then there is some i, $\ell \le i < m$, such that $u, v \in C_i \cup C_{i+1}$. Hence, $u, v \in B^R_i$.

Each vertex's bags induce a subpath: We distinguish between vertices based on their inclusion in the maximal plane subgraph H and the left-Kőnig covers.

- **Vertices not in H and not in any cover.** Let u be such a vertex. Since $u \notin H \cup \bigcup C_i$, u appears exclusively in the bag $B_{i,j}$ corresponding to its specific position.
- **Vertices not in H, but in some cover.** Let u be a vertex not incident to any edge in H. Therefore, u is located strictly between two edges e_j and e_{j+1}. For any $i > j$, u lies to the left of e_i ($u \in L_i$). If $u \in C_i$, then by Lemma 2.2 (II) we have $u \in C_{i-1}$ down to C_{j+1}. Symmetrically, for any $i \le j$, u lies to the right of e_i ($u \in R_i$). If $u \in C_i$, then $u \in C_{i+1}$ up to C_j. Thus, the index set $\{p \mid u \in C_p\}$ is a contiguous interval, and the bags containing u induce a subpath.
- **Vertices in H.** Let $u \in H$. Since H is a plane subgraph consisting of caterpillars, the edges of H incident to the vertex u form a contiguous subsequence $e_\ell, e_{\ell+1}, \ldots, e_r$. Consequently, u is explicitly contained in every bag from B^L_ℓ to B^R_r (the *spine interval*). Note that u cannot be in C_i for any $\ell \le i \le r$, because the sets L_i and R_i exclude the endpoints of e_i. However, u may belong to covers outside this range. If $u \in C_p$ for some $p > r$, then u lies to the left of e_p. By Lemma 2.2 (II), u must also be in $C_{r+1}, \ldots, C_{p-1}$. Since $u \in C_{r+1}$, it is included in the bag B^R_r (which contains C_{r+1}). This connects the spine interval to the right. Symmetrically, if $u \in C_p$ for some $p < \ell$, monotonicity implies $u \in C_{p+1}, \ldots, C_{\ell-1}$. Since $u \in C_{\ell-1}$, it is included in B^L_ℓ, connecting the spine interval to the left. Therefore, the union of bags containing u forms a single subpath. ◀

3 The Lower Bound

In this section, we construct a graph with a 2-layer k-matching-planar drawing that contains a $K_{3\lfloor k/2 \rfloor+2}$ minor. Let $\tau = \lfloor \frac{k}{2} \rfloor$ be a positive integer. Define a bipartite graph $G_k = (A \cup B, E)$ with bipartition A and B, where each side is partitioned into three blocks:

$$A = X_1 \dot\cup X_2 \dot\cup X_3, \qquad B = Y_1 \dot\cup Y_2 \dot\cup Y_3,$$

and the block sizes are $|X_1| = |X_3| = \tau$, $|X_2| = \tau + 1$, $|Y_1| = |Y_3| = \tau$, and $|Y_2| = \tau + 1$.

The edge set E consists of all edges between the following pairs of blocks:

$$E = E(X_1, Y_1) \cup E(X_1, Y_2) \cup E(X_2, Y_1) \cup E(X_2, Y_2) \cup E(X_2, Y_3) \cup E(X_3, Y_2) \cup E(X_3, Y_3),$$

where for any two vertex sets U and V, $E(U, V) := \{\, uv : u \in U,\ v \in V \,\}$ (See fig. 2).

▶ **Theorem 3.1.** *The graph G_k is 2-layer k-matching-planar and* $\mathrm{pw}(G_k) = 3\lfloor k/2 \rfloor + 1$.

Proof. First, we describe a 2-layer drawing of G_k and show that it is k-matching-planar. Place the vertices of A on a horizontal line ℓ_A and the vertices of B on a parallel horizontal line ℓ_B below ℓ_A. Along ℓ_A, order the vertices from left to right so that all vertices of X_1 come first, followed by all vertices of X_2, and finally all vertices of X_3. Within each block the order is arbitrary. Similarly, along ℓ_B, order the vertices from left to right so that all vertices of Y_1 come first, followed by all vertices of Y_2, and finally all vertices of Y_3, again

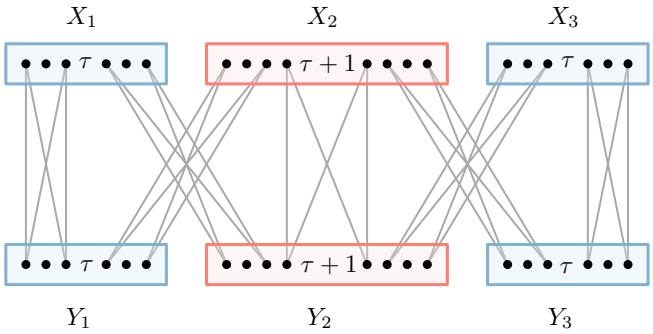


Figure 2 Illustration of the construction of the graph G_k.

with arbitrary order inside each block. Draw every edge of G_k as a straight-line segment between its endpoints. This is a valid 2-layer drawing of G_k.

Let $e = \{a, b\}$ with $a \in A$ and $b \in B$. Two edges cross if and only if their endpoints appear in opposite orders on the two layers. The drawing is symmetric under reflection and under exchanging the two layers. Hence, up to symmetry, it suffices to consider the following three cases.

- **Case 1:** $a \in X_1$, $b \in Y_1$. Every edge crossing e has at least one endpoint in the set $(X_1 \cup Y_1) \setminus \{a, b\}$, which has size $2\tau - 2 \leq k - 2$.
- **Case 2:** $a \in X_1$, $b \in Y_2$. Every edge crossing e has at least one endpoint in the set $(Y_1 \cup Y_2) \setminus \{b\}$, which has size $2\tau \leq k$.
- **Case 3:** $a \in X_2$, $b \in Y_2$. Every edge crossing e has at least one endpoint in the set $(X_2 \cup Y_2) \setminus \{a, b\}$, which has size $2\tau \leq k$.

Thus any matching of crossing edges of e has size at most k. Therefore, the described 2-layer drawing of G_k is k-matching-planar.

Note that $|A| = |B| = 3\tau + 1 = 3\lfloor k/2 \rfloor + 1$. Let $n = 3\tau + 1$. As G_k is a subgraph of the complete bipartite graph $K_{n,n}$ we have $\mathrm{pw}(G_k) \leq \mathrm{pw}(K_{n,n}) = n = 3\lfloor k/2 \rfloor + 1$.

We now show that this is the minimum possible width by finding a $K_{3\tau+2}$ minor in G_k. First we take a maximum matching on the edges $E(X_1, Y_1)$ and contract every vertex of Y_1 along its matching edge to the other endpoint in X_1. As for every pair u, v of vertices in X_1 there was a vertex y in Y_1 that was contracted to u and had edges to both u and v, there now is a direct edge between u and v. Therefore, the vertices of X_1 now form a complete graph. Note that every vertex in X_1 now has an edge to every vertex in X_2 and Y_2. Similarly, we take maximum matchings on $E(X_2, Y_3)$ and $E(X_3, Y_2)$ and contract the vertices of X_3 and Y_3 along their matching edges to the corresponding endpoints in X_2 and Y_3. This results in a complete graph on the vertices of X_2 as well as a complete graph on the vertices of Y_2. Since between X_2 and Y_2 there is a complete bipartite graph and X_1 has edges to all vertices of $X_2 \cup Y_2$ the resulting graph is a complete graph on $3\tau + 2$ vertices. That is $\mathrm{pw}(G_k) \geq 3\tau + 1 = 3\lfloor k/2 \rfloor + 1$.

◀

4 Future Work

Closing the gap between the lower and upper bound on the pathwidth of graphs with a 2-layer k-matching-planar drawing remains as an obvious open question.

Hendery, Karol, and Wood [11] studied circular k-matching-planar drawings, where the vertices are placed on a circle and the edges are drawn as straight-line segments (these drawings are also called convex). Any 2-layer k-matching-planar drawing can be interpreted as a circular drawing, by treating the horizontal vertex layers as two consecutive segments of a circle. Hendery et al. [11] proved that graphs with such a drawing have treewidth in $O(k^3 \log^2 k)$. We think it is interesting to study this further. Note that the pathwidth of the graphs with such drawings is unbounded, as outerplanar graphs have arbitrarily large pathwidth and, by definition, circular (0-matching-)planar drawings.

Acknowledgement. This work is part of an ongoing research initiated at the Rhein-Ruhr Computational Geometry Workshop held in Hagen, Germany from September 2 to 5, 2025. The authors would like to thank the organizers and participants of the workshop for providing a welcoming environment and a stimulating atmosphere for collaboration.

References

1 Patrizio Angelini, Giordano Da Lozzo, Henry Förster, and Thomas Schneck. 2-layer k-planar graphs density, crossing lemma, relationships and pathwidth. *Comput. J.*, 67(3):1005–1016, 2024. doi:10.1093/COMJNL/BXAD038.

2 Therese Biedl, Steven Chaplick, Michael Kaufmann, Fabrizio Montecchiani, Martin Nöllenburg, and Chrysanthi N. Raftopoulou. Layered fan-planar graph drawings. In Javier Esparza and Daniel Král', editors, *MFCS 2020*, volume 170 of *LIPIcs*, pages 14:1–14:13. LIPIcs, 2020. doi:10.4230/LIPICS.MFCS.2020.14.

3 Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 2015. doi:10.1007/978-3-319-21275-3.

4 Erik D. Demaine, Mohammad Taghi Hajiaghayi, and Ken-ichi Kawarabayashi. Algorithmic graph minor theory: Decomposition, approximation, and coloring. In *FOCS 2005*, pages 637–646. IEEE Computer Society, 2005. doi:10.1109/SFCS.2005.14.

5 Vida Dujmovic, Michael R. Fellows, Matthew Kitching, Giuseppe Liotta, Catherine McCartin, Naomi Nishimura, Prabhakar Ragde, Frances A. Rosamond, Sue Whitesides, and David R. Wood. On the parameterized complexity of layered graph drawing. *Algorithmica*, 52(2):267–292, 2008. doi:10.1007/S00453-007-9151-1.

6 Peter Eades and Sue Whitesides. Drawing graphs in two layers. *Theor. Comput. Sci.*, 131(2):361–374, 1994. doi:10.1016/0304-3975(94)90179-1.

7 Peter Eades and Nicholas C. Wormald. Edge crossings in drawings of bipartite graphs. *Algorithmica*, 11(4):379–403, 1994. doi:10.1007/BF01187020.

8 Michael R. Garey and David S. Johnson. Crossing number is NP-complete. *SIAM J. Algebraic Discrete Methods*, 4(3):312–316, 1983. doi:10.1137/0604033.

9 Grzegorz Gutowski, Panos Giannopoulos, Miriam Goetze, Maarten Löffler, Martin Nöllenburg, Yuto Okada, Jonathan Rollin, Birgit Vogtenhuber, and Alexander Wolff. One-sided local crossing minimization. private communication, 2025. preliminary version at CoRR, 2510.00331. doi:10.48550/ARXIV.2510.00331.

10 Patrick Healy and Nikola S. Nikolov. Hierarchical drawing algorithms. In Roberto Tamassia, editor, *Handbook on Graph Drawing and Visualization*, pages 409–453. Chapman and Hall/CRC, 2013. doi:10.1201/b15385.

11 Kevin Hendrey, Nikolai Karol, and David R. Wood. Structure of k-matching-planar graphs. *CoRR*, abs/2507.22395, 2025. doi:10.48550/ARXIV.2507.22395.

12 Petr Hlinený. Crossing-number critical graphs have bounded path-width. *J. Comb. Theory B*, 88(2):347–367, 2003. doi:10.1016/S0095-8956(03)00037-6.

13 Jisu Jeong, Sigve Hortemo Sæther, and Jan Arne Telle. Maximum matching width: New characterizations and a fast algorithm for dominating set. *Discret. Appl. Math.*, 248:114–124, 2018. doi:10.1016/J.DAM.2017.09.019.

14 Yasuaki Kobayashi, Yuto Okada, and Alexander Wolff. Recognizing 2-layer and outer k-planar graphs. In Oswin Aichholzer and Haitao Wang, editors, *SoCG 2025*, volume 332 of *LIPIcs*, pages 65:1–65:16. LIPIcs, 2025. doi:10.4230/LIPICS.SOCG.2025.65.

15 Laura Merker, Lena Scherzer, Samuel Schneider, and Torsten Ueckerdt. Intersection graphs with and without product structure. In Stefan Felsner and Karsten Klein, editors, *GD 2024*, volume 320 of *LIPIcs*, pages 23:1–23:19. LIPIcs, 2024. doi:10.4230/LIPICS.GD.2024.23.

16 Yuto Okada. Pathwidth of 2-layer k-planar graphs. *CoRR*, 2507.21864, 2025. doi:10.48550/ARXIV.2507.21864.

17 Neil Robertson and Paul D. Seymour. Graph minors. i. excluding a forest. *J. Comb. Theory B*, 35(1):39–61, 1983. doi:10.1016/0095-8956(83)90079-5.

18 Matthew Suderman. Pathwidth and layered drawings of trees. *Int. J. Comput. Geom. Appl.*, 14(3):203–225, 2004. doi:10.1142/S0218195904001433.

19 David R. Wood. 2-layer graph drawings with bounded pathwidth. *J. Graph Algorithms Appl.*, 27(9):843–851, 2023. doi:10.7155/JGAA.00647.

20 Meirav Zehavi. Parameterized analysis and crossing minimization problems. *Comput. Sci. Rev.*, 45:100490, 2022. doi:10.1016/J.COSREV.2022.100490.

High Beer Index Implies Big Hollow Triangles*

Arun Kumar Das[1], Vít Jelínek[2], Jan Kynčl[2], Martin Pergel[2], Felix Schröder[2], Peter Stumpf[2], and Pavel Valtr[2]

1 **Czech Technical University**
2 **Charles University, Prague**

Abstract

The Beer index of a set $S \subseteq \mathbb{R}^2$ is the probability that two points chosen uniformly independently in S will see each other, that is, the segment connecting the two points will be contained in S. Previously, it has been shown that a simply connected set of unit Lebesgue measure with Beer index $\beta > 0$ contains a convex subset of measure $\Omega(\beta)$. Easy examples show that the simple connectivity assumption cannot be omitted: a general set may have Beer index equal to 1 and no convex subset of positive measure.

However, in this paper we extend the above result to general sets as follows: we show that a set $S \subseteq \mathbb{R}^2$ of unit Lebesgue measure with Beer index $\beta > 0$ contains the boundary of a triangle of measure $\Omega(\beta^9)$. Furthermore, if S is an open domain of unit measure with K holes, then it contains the boundary of a triangle of measure $\Omega(\beta/K)$ and a convex subset of measure $\Omega(\beta/K^2)$.

1 Introduction

Convexity is without doubt one of the central notions of geometry. Closely related to convexity is the notion of *visibility*: given a set $S \subseteq \mathbb{R}^d$, we say that a pair of points $x, y \in S$ *see each other (in S)* if the segment connecting x to y is wholly contained in S. In particular, a set is convex if and only if any two of its points see each other.

In the past, various approaches have been proposed to quantify how close a given set is to being convex; see, e.g., the works of Stern [15], Boxer [7], Zunic and Rosin [16], Rahtu, Salo and Heikkila [11], or Rosin and Mumford [12].

In this paper, we consider one of the oldest, and arguably one of the most natural measures of convexity, called *the Beer index*. The Beer index of a set S, denoted $b(S)$, is the probability that any two random points chosen uniformly independently from a set $S \subseteq \mathbb{R}^d$ see each other. This notion has been introduced and thoroughly studied by Beer [3, 4, 5] in the 1970s under the name 'index of convexity', and later independently considered by Stern [15].

The Beer index has been previously studied from a computational perspective: Rote [13] has shown that the index can be computed in polynomial time for a polygonal set (possibly with holes) with boundary consisting of n segments. Abrahamsen and Fredslund-Hansen [1] present another algorithm for this problem. Meanwhile Buchin, Kostitsyna, Löffler and Silveira [8] have considered an analogue of the Beer index defined with respect to a non-uniform probability distribution, and found an efficient way to approximate it.

The Beer index has also been employed as a tool in the analysis of algorithms that tackle other geometric problems. For instance, Cabello, Cibulka, Kynčl, Saumell and Valtr [9] developed a probabilistic near-linear algorithm that approximates the largest convex subset of a given convex polygon – a problem known as 'the potato-peeling problem' [10]. A crucial

* This research is supported by the Czech Science Foundation under the grant agreement no. 23-04949X.

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

step of their analysis is a result that shows that the Beer index of a polygon can be used to bound from below the size of the largest convex subset in a polygon.

The relationship between the Beer index and the existence of large convex subsets has been generalized and tightened by Balko, Jelínek, Valtr and Walczak [2], who showed that a simply connected set of unit measure contains a convex subset of measure at least linear in the Beer index (see Theorem 2.1 below for the precise statement). They also made a series of conjectures pointing towards possible generalisations to sets that are not assumed to be simply connected. For such sets, it is not in general possible to claim that positive Beer index implies large convex subsets – indeed, there are sets with Beer index equal to 1 and no convex subsets of positive measure. However, Balko et al. conjectured [2, Conjecture 4.5] that in a planar set S of unit measure, a positive Beer index implies that S contains the boundary of a triangle whose area can be bounded from below by a function of the Beer index.

As our first main result, we confirm this conjecture. While this result applies to arbitrary planar sets satisfying only basic measurability assumptions, we also focus on sets that possess a more explicit combinatorial structure, namely polygons with a bounded number of polygonal holes. The number of such holes gives us a convenient measure of how far a given set is from being simply connected, and we obtain bounds on the size of both hollow and full triangles contained in such a polygonal set, as a function of the Beer index and the number of holes.

One of our motivations for the study of the Beer index stems from the goal of understanding graph-theoretic properties of visibility graphs. A *visibility graph* of a set S is the (typically infinite) graph whose vertices are the points of S, with two vertices being adjacent if the two corresponding points see each other. The Beer index is precisely the edge density of the visibility graph.

Visibility graphs may contain an arbitrary finite or countable graph as an induced subgraph. However, we conjecture that their subgraph densities are far from arbitrary. Specifically, we conjecture that a visibility graph with positive edge density must also have positive triangle density. This contrasts with general graphs, which may be triangle-free while having edge density up to $\frac{1}{2}$.

Our result on high Beer index implying a large hollow triangle is a weaker version of the above conjecture, since it is known [2] that a positive triangle density in the visibility graph implies the existence of large hollow triangles.

In this extended abstract, we limit ourselves to introducing the necessary definitions and stating our main results. A full version of the paper is in the Appendix.

2 Definitions and results

Recall that two points x, y of a set $S \subseteq \mathbb{R}^d$ *see each other* if the closed segment with endpoints x and y is a subset of S; this is denoted by $x \overset{S}{\sim} y$, or just $x \sim y$ if S is clear from the context.

The *visibility set* of a point x in a set $S \subseteq \mathbb{R}^2$ is the set $\mathrm{Vis}(x) = \{y \in S;\ x \overset{S}{\sim} y\}$, and the *segment set* of $S \subseteq \mathbb{R}^d$ is the set $\mathrm{Seg}(S) = \{(x, y) \in S \times S;\ x \overset{S}{\sim} y\}$.

For ease of notation, we use the following indicator function for $x, y \in S$:

$$I(x \sim y) = \begin{cases} 1 & \text{if } x \text{ and } y \text{ see each other in } S \\ 0 & \text{otherwise.} \end{cases}$$

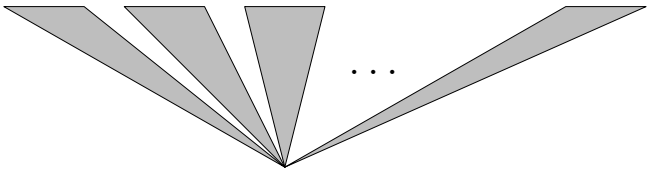


Figure 1 An example of a set S consisting of N triangles of measure $\frac{1}{N}$. We then see that $b(S) = \frac{1}{N}$ and the largest convex subset of S has measure $\frac{1}{N}$.

For a set $S \subseteq \mathbb{R}^d$, we let $\lambda_d(S)$ denote the d-dimensional Lebesgue measure of S. We refer to it simply as 'the measure of S'. We implicitly assume that all the random choices we use are performed with respect to this measure, and all the integrals are evaluated with respect to it. Let $\mathbb{E}[f]$ denote the expectation of a random variable f.

For a set $S \subseteq \mathbb{R}^d$ of positive finite measure, we define the Beer index $b(S)$ as the probability that two randomly independently chosen points of S see each other, or in other words

$$b(S) = \mathbb{E}_{x,y \in S}[I(x \sim y)] = \frac{1}{\lambda_d(S)^2} \int_S \int_S I(x \sim y)\, \mathrm{d}x\, \mathrm{d}y = \frac{\lambda_{2d}(\mathrm{Seg}(S))}{\lambda_d(S)^2}.$$

Note that the Beer index can equivalently be viewed as the average normalized measure of the visibility set, i.e.,

$$b(S) = \mathbb{E}_{x \in S}\left[\frac{\lambda_d(\mathrm{Vis}(x))}{\lambda_d(S)}\right] = \frac{1}{\lambda_d(S)^2} \int_S \lambda_d(\mathrm{Vis}(x))\, \mathrm{d}x.$$

Another way to view the Beer index of a set $S \subseteq \mathbb{R}^d$ is as the edge density of the *visibility graph* of S, which is the graph whose vertices are the points of S and edges connect the pairs of vertices that see each other. The *p-component of S* (which is short for 'polygonal path connectivity component of S') is the vertex set of a connected component of the visibility graph of S, or equivalently, any maximal subset of S in which any two vertices can be connected by a piecewise linear curve contained in S. A set S is *p-connected* if it consists of a single p-component.

Following standard topological terminology, we say that a p-connected set S is *simply connected* if any closed curve in S can be contracted to a point. We say that a set is *p-componentwise simply connected* if each of its p-components is simply connected. The following theorem is one of the main results of [2].

▶ **Theorem 2.1** ([2, Theorem 1.3]). *Every p-componentwise simply connected set S in the plane of unit measure contains a convex subset of measure $\Omega(b(S))$.*

Any convex set in the plane contains a triangle covering a constant proportion of its measure; specifically, Blaschke [6] showed that any convex set $C \subseteq \mathbb{R}^2$ contains a triangle of measure at least $\frac{3\sqrt{3}}{4\pi}\lambda_2(C)$. Thus, the following is an equivalent version of Theorem 2.1.

▶ **Corollary 2.2.** *Every p-componentwise simply connected set S in the plane of unit measure contains a triangle of measure $\Omega(b(S))$ as a subset.*

Note that the linear dependency on $b(S)$ in Theorem 2.1 and Corollary 2.2 is in general best possible, as shown by the example of a set S which is the union of N disjoint convex sets, each of measure $\frac{1}{N}$; see Figure 1.

Our motivation is to extend Theorem 2.1 and Corollary 2.2 to sets which are not simply connected. One cannot, however, simply drop the simple-connectivity assumption, as shown

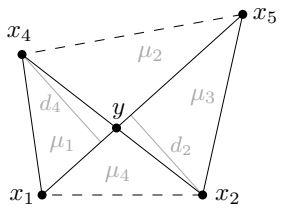


Figure 2 Illustration of the proof of Theorem 2.3.

by the set $[0,1] \times [0,1] \setminus (\mathbb{Q} \times \mathbb{Q})$ which has Beer index 1 and no convex subset of positive measure.

One possible way to generalize Corollary 2.2, which has been previously suggested as a conjecture [2], is to consider hollow triangles instead of triangles. A *hollow triangle* is the boundary of a triangle, that is, a union of three segments xy, yz and xz, for a triple of points x, y, z. The *size* of a hollow triangle is the measure of its convex hull. Our first main result confirms that such a generalization is possible, at least in a qualitative sense:

▶ **Theorem 2.3.** *A set S in the plane of unit measure with $b(S) > 0$ contains a hollow triangle of size at least $\Omega(b(S)^9)$.*

Let us briefly sketch the proof of Theorem 2.3. The argument is based upon two lemmas, the first of which is inspired by the more general Theorem 1.5 from [2].

▶ **Lemma 2.4.** *Let $S \subseteq \mathbb{R}^2$ be a set of unit measure and x, y, z a triple of points chosen independently uniformly at random in S. Then for any $\delta > 0$, the probability that the convex hull of $\{x, y, z\}$ has measure at most δ is at most 12δ.*

The second ingredient of the proof of Theorem 2.3 is a variant of a known extremal result on the density of bipartite subgraphs. It can be viewed as the special case of Sidorenko's inequality for the complete bipartite graph $K_{3,3}$ [14].

▶ **Lemma 2.5.** *Let $S \subseteq \mathbb{R}^2$ be a set of unit measure with $b(S) > 0$. Let $(x_1, x_2, x_3, x_4, x_5, x_6)$ be a sixtuple of points chosen uniformly independently from S. Then, with probability at least $b(S)^9$, each of the points x_1, x_2, x_3 sees all the points x_4, x_5, x_6 in S.*

To prove Theorem 2.3, choose uniformly and independently six points $x_1, x_2, \ldots, x_6$ in S. In view of Lemmas 2.4 and 2.5, we can argue that with positive probability each point from the set $A = \{x_1, x_2, x_3\}$ will see all the points from $B = \{x_4, x_5, x_6\}$, and moreover, the convex hull of any three of the six points will have measure at least $\Omega(b(S)^9)$.

Since the graph $K_{3,3}$ is nonplanar, among the nine segments connecting a point in A to a point in B, at least two cross each other. Suppose that the segment x_1x_5 crosses the segment x_2x_4 in a point $y \in S$, as in Figure 2.

Let μ_1, μ_2, μ_3 and μ_4 denote respectively the measures of the triangles x_1x_4y, x_4x_5y, x_2x_5y and x_1x_2y. Let d_2 and d_4 be the distance of x_2 and x_4 to the line through x_1 and x_5. Note that

$$\mu_1\mu_3 = |x_1y| \cdot |x_5y| \cdot d_2 \cdot d_4 = \mu_2\mu_4.$$

It follows that $\max\{\mu_1, \mu_3\} \geq \min\{\mu_2, \mu_4\}$. Our assumptions on A and B also guarantee that

$$\max\{\mu_1, \mu_3\} + \min\{\mu_2, \mu_4\} \geq \Omega(b(S)^9).$$

We conclude that at least one of the two triangles x_1x_4y and x_2x_5y has measure at least $\Omega(b(S)^9)$, and Theorem 2.3 follows.

The bound of $\Omega(b(S)^9)$ in Theorem 2.3 is likely far from best possible; in fact, we conjecture that it might be replaced by a linear bound to match Theorem 2.1 and Corollary 2.2 – see Conjecture 3.1 below. To bridge the gap between the quantitative bounds for general sets and for simply connected sets, we focus on sets that have only a bounded number of holes in them. Formally, for an integer K, a *K-hole set* is a set S that can be expressed as

$$S = P \setminus (H_1 \cup H_2 \cup \cdots \cup H_K),$$

where P is a bounded, open and simply connected subset of $\mathbb{R}^2$, while $H_1, \ldots, H_K$ are pairwise disjoint closed and simply connected subsets of P, which we call *holes*.

Thus, K-hole sets represent a convenient relaxation of simply connected sets, and we will show that they contain both hollow triangles and convex subsets whose size can be bounded in terms of $b(S)$ and K. Specifically, we obtain the following results, whose proofs we omit.

▶ **Theorem 2.6.** *Every K-hole set $S \subseteq \mathbb{R}^2$ of unit measure contains a hollow triangle of size $\Omega(b(S)/K)$.*

▶ **Theorem 2.7.** *Every K-hole set $S \subseteq \mathbb{R}^2$ of unit measure has a convex subset of measure $\Omega(b(S)/K^2)$.*

3 Open problems and future work

Theorem 2.3 shows that a planar set S has a hollow triangle of size $\Omega(b(S)^9)$. The degree-9 dependency is probably not optimal, and we conjecture that it can be improved to linear.

▶ Conjecture 3.1. Every set S in the plane of unit measure contains a hollow triangle of size $\Omega(b(S))$ as a subset.

Theorem 2.7 involves a quadratic dependency on K, which is perhaps also not tight.

Theorems 2.3 and 2.6 show that under various assumptions a set with high Beer index must contain a hollow triangle of large size. A natural way to further strengthen these results, apart from improvements in quantitative bounds, is to show that large Beer index guarantees a positive density of hollow triangles.

The *triangle density* $t(S)$ of a set S is the probability that for three points $x, y, z \in S$ chosen uniformly independently at random, every two of them see each other (in other words, they induce a hollow triangle in S).

Theorem 2.1 implies that for a simply connected set S, we have $t(S) = \Omega(b(S)^3)$. We conjecture that this is true even without the assumption of simple connectivity, possibly with a worse dependency between $t(S)$ and $b(S)$:

▶ Conjecture 3.2. There is a function $f\colon (0,1) \to (0,1)$ such that every set $S \subseteq \mathbb{R}^2$ with $b(S) > 0$ satisfies $t(S) \geq f(b(S))$.

Conjecture 3.2, if true, would imply that the subgraph densities in infinite visibility graphs are more constrained than in general graphs. Indeed, a general graph may have edge density up to $\frac{1}{2}$ and triangle density 0, as witnessed by the complete bipartite graphs.

In fact, we are confident enough to make a stronger version of Conjecture 3.2, and conjecture that the cubic bound we have for simply connected sets may hold without the simple connectivity assumption.

▶ Conjecture 3.3. For any set $S \subseteq \mathbb{R}^2$, $t(S) = \Omega(b(S)^3)$ (provided $b(S)$ and $t(S)$ exist).

It can be shown that every set S of unit measure contains a hollow triangle of size $\Omega(t(S))$, since more triples of points in S give hollow triangles than triangles of smaller size.

Thus, Conjectures 3.2 and 3.3 can be seen as attempts to further generalize Theorem 2.3, by showing that instead of a single large hollow triangle, we can in fact guarantee a positive density of hollow triangles, some of which are then necessarily large.

In fact, even the cubic bound in Conjecture 3.3 may still not be optimal. Consider, for an integer $N \geq 1$, a set S consisting of N disjoint convex sets of measure $\frac{1}{N}$. Clearly, $b(S) = \frac{1}{N}$ and $t(S) = \frac{1}{N^2}$, providing an example of sets with $t(S) = b^2(S)$ as $b(S)$ tends to 0. We are not aware of any family of sets S with $t(S) = o(b^2(S))$ as $b(S)$ tends to 0.

References

1 Mikkel Abrahamsen and Viktor Fredslund-Hansen. Degree of convexity and expected distances in polygons, 2022. doi:10.48550/arXiv.2208.07106.

2 Martin Balko, Vít Jelínek, Pavel Valtr, and Bartosz Walczak. On the Beer index of convexity and its variants. *Discrete Comput. Geom.*, 57(1):179–214, 2017. doi:10.1007/s00454-016-9821-3.

3 Gerald Beer. Continuity properties of the visibility function. *Michigan Math. J.*, 20:297–302, 1973.

4 Gerald Beer. The index of convexity and the visibility function. *Pacific J. Math.*, 44(1):59–67, 1973.

5 Gerald Beer. The index of convexity and parallel bodies. *Pacific J. Math.*, 53(2):337–345, 1974.

6 Wilhelm Blaschke. Über affine Geometrie III: Eine Minimumeigenschaft der Ellipse. *Ber. Verh. Kön. Sächs. Ges. Wiss. Leipzig Math.-Phys. Kl.*, 69:3–12, 1917.

7 Laurence Boxer. Computing deviations from convexity in polygons. *Pattern Recognition Letters*, 14(3):163–167, 1993. doi:10.1016/0167-8655(93)90067-N.

8 Kevin Buchin, Irina Kostitsyna, Maarten Löffler, and Rodrigo I. Silveira. Region-based approximation of probability distributions (for visibility between imprecise points among obstacles). *Algorithmica*, 81(7):2682–2715, 2019. doi:10.1007/s00453-019-00551-2.

9 Sergio Cabello, Josef Cibulka, Jan Kynčl, Maria Saumell, and Pavel Valtr. Peeling potatoes near-optimally in near-linear time. *SIAM Journal on Computing*, 46(5):1574–1602, 2017. doi:10.1137/16M1079695.

10 Jacob E. Goodman. On the largest convex polygon contained in a non-convex n-gon, or how to peel a potato. *Geom. Dedicata*, 11(1):99–106, 1981.

11 Esa Rahtu, Mikko Salo, and Janne Heikkilä. A new convexity measure based on a probabilistic interpretation of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(9):1501–1512, 2006. doi:10.1109/TPAMI.2006.175.

12 Paul L. Rosin and Christine L. Mumford. A symmetric convexity measure. *Computer Vision and Image Understanding*, 103(2):101–111, 2006. doi:10.1016/j.cviu.2006.04.002.

13 Günter Rote. The degree of convexity. In Sándor Fekete, editor, *Abstracts of the 29th European Workshop on Computational Geometry (EuroCG'13)*, pages 69–72, March 2013. URL: http://page.mi.fu-berlin.de/rote/Papers/pdf/The+degree+of+convexity.pdf.

14 Alexander F. Sidorenko. Inequalities for functionals generated by bipartite graphs. *Discrete Mathematics and Applications*, 2(5):489–504, 1992. doi:doi:10.1515/dma.1992.2.5.489.

15 Helman I. Stern. Polygonal entropy: a convexity measure for polygons. *Pattern Recogn. Lett.*, 10(4):229–235, 1989.

16 Jovisa Zunic and Paul L. Rosin. A new convexity measure for polygons. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(7):923–934, 2004. doi:10.1109/TPAMI.2004.19.

On the Surjectivity of a Map by Kapranov and Voevodsky in four Dimensions

Yan Alves Radtke[1]

1 **TU Berlin**
alves@math.tu-berlin.de

Abstract

We consider a map defined by Kapranov and Voevodsky, mapping from the higher Bruhat orders to the higher Stasheff-Tamari orders on triangulations of the cyclic polytope. This map generalizes the quotient map given by the Silvester congruence, which is a map from the weak Bruhat order on permutations to the Tamari lattice. It is an open question whether this map is surjective. The surjectivity for triangulations of two and three dimensional cyclic polytopes was proven by Thomas in 2003. We prove that the map is surjective in the four dimensional case.

1 Introduction

The weak Bruhat order on permutations and the Tamari Lattice on triangulations of the n-gon are well studied partial orders. One striking similarity between them is that both can be generalized to higher ranks by considering maximal chains of the respective orders. The resulting partial orders are the so called *higher Bruhat* and *higher Stasheff-Tamari* orders. There is a well-known map from permutations to triangulations, see Figure 1. This map can be extended to a map between the higher rank generalizations of the respective orders. In the following, we introduce the relevant notions to define this map.

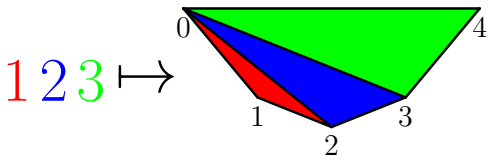


Figure 1 An example for the map from the weak Bruhat order to the Tamari Lattice. We iterate through the permutation from left to right. For the entry i in the permutation, we cut off the vertex i from the triangulation. We therefore first cut off the vertex 1, to obtain the red triangle $(0, 1, 2)$. Since the vertex 1 is removed, by cutting off 2 we obtain the blue triangle $(0, 2, 3)$. The remaining green triangle is obtained by cutting off 3.

1.1 Higher Stasheff-Tamari Orders

Cyclic polytopes are fundamental objects in convex geometry and combinatorial polytope theory. Its triangulations are ordered by the so called higher Stasheff-Tamari orders, which were introduced by Kapranov and Voevodsky [8] and Edelman and Reiner [3].

We introduce some facts about the higher Stasheff-Tamari orders. For a more complete overview, we refer to the survey of Rambau and Reiner [11]. The d-dimensional moment curve is the image of the function

$$M_d(t) \coloneqq (t, t^2, \ldots, t^d).$$

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

It is well known that for $t_1 < t_2 < \cdots < t_n$, the underlying oriented matroid of the points $\{M_d(t_i) \mid i = 1, \ldots, n\}$ does not depend on the particular choice of t_i. We therefore refer to the convex hull of any n points on the moment curve as *the* cyclic polytope $C(n, d)$. We identify the vertex $M_d(t_i)$ with i.

Let P be a d-dimensional polytope with vertex set V. A subset T of the $(d+1)$-subsets of V is a *triangulation* if the union over all convex hulls over all subsets in T is exactly P and the intersection between the convex hulls of two different elements in T is empty or the convex hull of the vertices that lie in both subsets. In abuse of notation, we also refer to a $(d+1)$-subset of vertices of a d-dimensional polytope as simplices.

The two triangulations of $C(d+2, d)$ have a particularly nice description: Consider the polytope $C(d+2, d+1)$. If we look at the lower and upper convex hulls of this polytope, with respect to the last coordinate, we obtain the two triangulations of $C(d+2, d)$. We call them T_{bottom} and T_{top}, respectively.

A triangulation T can admit $d+2$ vertices V', so that T induces a triangulation of $\operatorname{conv} V'$. We can then *flip*, where we replace in T the triangulation of V' induced by T with the other unique triangulation of V'. This again is a triangulation of P.

We define a partial order on the set of all triangulations of $C(n, d)$ by the following cover relation: $T_1 \lessdot T_2$, if we can obtain T_2 by flipping a T_{bottom} to T_{top} in T_1. In the case of $d = 2$, this is the so called *Tamari lattice*, see also Figure 2. For $d \geq 3$, this is the so called first higher Stasheff-Tamari order. There is also the second higher Stasheff-Tamari order on the set of triangulations of $C(n, d)$. Williams [13] showed that both orders coincide. We therefore only refer to them as the higher Stasheff-Tamari order or $\mathbf{HST}(n, d)$. Rambau [10] showed that $\mathbf{HST}(n, d)$ has a unique minimal element and a unique maximal element. For a maximal chain $T_1 \lessdot T_2 \lessdot \cdots \lessdot T_k$ in $\mathbf{HST}(n, d)$ we can associate a sequence of sets $S_1, \ldots, S_{k-1}$, such that we obtain T_{i+1} from T_i by flipping S_i. We call two maximal chains in $\mathbf{HST}(n, d)$ equivalent if their sequence of sets differs only in the order of the sets.

▶ **Theorem 1.1** ([10]). *For $d \geq 2$ there is a bijection between $\mathbf{HST}(n, d)$ and equivalence classes of maximal chains in $\mathbf{HST}(n, d-1)$. The bijection is given by taking the sets of a sequence of sets in a maximal chain and considering them as simplices of a triangulation of $C(n, d+1)$.*

If a maximal chain C is in the equivalence class corresponding to T, we say that C is a *decomposition* of T. For an example, see Figure 2.

1.2 Higher Bruhat Order and Signotopes

The higher Bruhat orders were introduced by Manin and Schechtmann [9]. The elements of the higher Bruhat order are the signotopes, which were introduced by Felsner and Weil [6]. An oriented matroid view was given by Ziegler [14].

In this section, we introduce the concepts on signotopes that we need. Complete proofs can be found in [6].

The *packet* of $\{x_1 < x_2 < \cdots < x_l\} \subseteq [n]$ is the sequence of sets:

$$\{x_2, x_3, \ldots, x_k\}, \{x_1, x_3, \ldots, x_k\}, \ldots, \{x_1, x_2, \ldots, x_{k-1}\}.$$

▶ **Definition 1.2.** For $n \geq r \geq 1$ a function $\sigma : \binom{[n]}{r} \to \{+, -\}$ is a rank r signotope on n elements, if for all $(r+1)$-subsets S of $[n]$, there is at most one sign change of σ on the packet of S.

Some examples of signotopes are depicted in Figure 3.

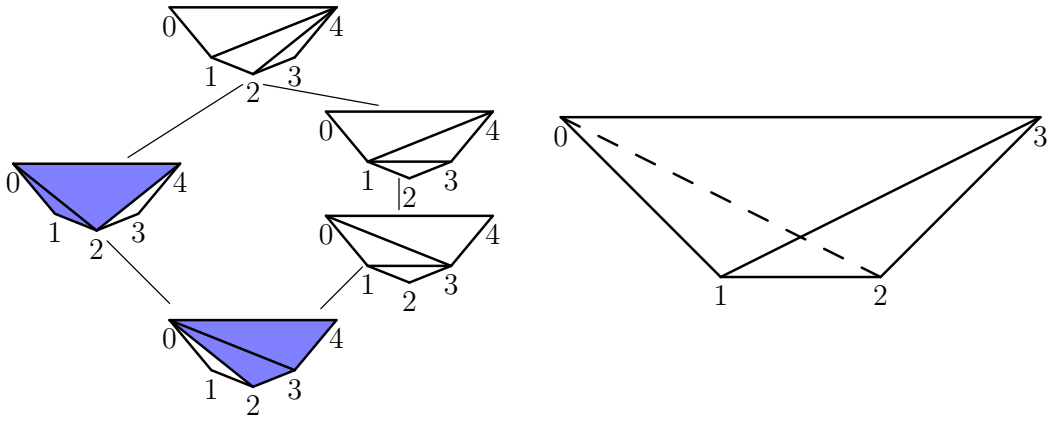


Figure 2 On the right side, the 3 dimensional cyclic polytope with 4 vertices is shown. We can see that $\{0,1,2\},\{0,2,3\}$ is the triangulation T_{bottom}. On the left side, $\mathbf{HST}(5,2)$ is depicted. The left chain corresponds to the triangulation $\{0,2,3,4\},\{0,1,2,4\}$ of $C(5,3)$. The flips are highlighted.

$\sigma(\{2,3,4\})$	$\sigma(\{1,3,4\})$	$\sigma(\{1,2,4\})$	$\sigma(\{1,2,3\})$
−	−	−	−
+	−	−	−
+	+	−	−
+	+	+	−
+	+	+	+
−	+	+	+
−	−	+	+
−	−	−	+

Figure 3 All rank 3 signotopes on 4 elements.

▶ **Definition 1.3** (Higher Bruhat Order)**.** Let σ_1,σ_2 be two rank r signotopes on n elements. The higher Bruhat order is given by the covering relation where $\sigma_1 \lessdot \sigma_2$, if σ_1 and σ_2 agree on all r-subsets of $[n]$, except for one, which σ_1 maps to $-$ and σ_2 to $+$.

From now on, we always refer to $\mathbf{S}(n,r)$ as the set of signotopes ordered by the higher Bruhat order.

It is known that $\mathbf{S}(n,r)$ has unique minimal and maximal elements: the all $-$ and all $+$ signotopes. Similarly to higher Stasheff-Tamari orders, we can decompose elements of $\mathbf{S}(n,r)$ into equivalence classes of maximal chains of $\mathbf{S}(n,r-1)$. Let $\sigma_1,\ldots,\sigma_l$ be a maximal chain in $\mathbf{S}(n,r-1)$. The signotopes σ_i and σ_{i+1} differ in exactly one $(r-1)$-subset of $[n]$. This observation gives a linear order of $\binom{[n]}{r-1}$. We call this the flip sequence of the chain. We say that two maximal chains are equivalent if for every $(r+1)$-subset S, the packet of S appears in the same order in the flip sequences. We set $\sigma(S)=+$, if and only if this induced order is reverse-lexicographic. Then σ is a signotope in $\mathbf{S}(n,r)$. We also refer to $\sigma_1,\ldots,\sigma_l$ as a *decomposition* of σ, motivated by the following theorem.

▶ **Theorem 1.4** ([6])**.** *There is a bijection between signotopes in $\mathbf{S}(n,r)$ and equivalence classes of maximal chains in $\mathbf{S}(n,r-1)$. A maximal chain is mapped to the signotope described above.*

It is straightforward to see that $\mathbf{S}(n,1)$ is isomorphic to the Boolean lattice, by considering the sets $\sigma^{-1}(\{+\})$. The weak Bruhat order on permutations is isomorphic to $\mathbf{S}(n,2)$ under the identification in which the positive entries of σ correspond to the inversions of the

permutation. For example, the chain $---, --+, -++, +++$ in $\mathbf{S}(3,1)$ corresponds to the chain $\{\}, \{2\}, \{2,1\}, \{1,2,3\}$ and is a decomposition of the signotope $--+$ in $\mathbf{S}(3,2)$, which corresponds to the permutation 213.

1.3 The Map by Kapranov and Voevodsky.

We introduce the map defined by Kapranov and Voevodsky in [8] between the higher Bruhat order and the higher Stasheff-Tamari order. We will omit technical details.

A triangulation of $[0, n+1] = \{0, 1, 2, \ldots, n+1\}$ is fully characterized by the set of endpoints of its intervals different from 0 and $n+1$. Therefore,

$$[0, x_1], [x_1, x_2], \ldots, [x_{k-1}, x_k], [x_k, n+1] \Leftrightarrow \sigma : [n] \to \{+, -\}, \sigma(x) = \begin{cases} -, x \in \{x_1, \ldots, x_k\} \\ +, \text{else} \end{cases}$$

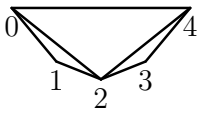
is easily seen to be an order preserving bijection f from $\mathbf{S}(n,1)$ to $\mathbf{HST}(n+2,1)$.

One can extend this function to higher dimensions and obtain a map f from $\mathbf{S}(n,d)$ to $\mathbf{HST}(n+2,d)$ inductively: Assume that f is already defined for $d-1$.

Let $\sigma \in S(n,d)$. By Theorem 1.4, there is an equivalence class of maximal chains of $\mathbf{S}(n,d-1)$ corresponding to σ. Let $\sigma_1 < \sigma_2 < \cdots < \sigma_l$ be one of these chains. If we consider $f(\sigma_1) < f(\sigma_2) < \cdots < f(\sigma_l)$, we can remove duplicates to obtain a maximal chain in $\mathbf{HST}(n+2,d-1)$. By Theorem 1.1, this chain corresponds to a triangulation T in $\mathbf{HST}(n+2,d)$. We define $f(\sigma) \coloneqq T$.

The following is an example for the two dimensional case.

$$312 \Leftrightarrow {+}{+}{-} \xleftrightarrow{\text{Decomp.}} \begin{pmatrix} +++ \\ ++- \\ +-- \\ --- \end{pmatrix} \xmapsto{f} \begin{pmatrix} [0,4] \\ [0,2],[2,4] \\ [0,1][1,2][2,4] \\ [0,1],[1,2][2,3],[3,4] \end{pmatrix} \xleftrightarrow{\text{Decomp.}}$$



▶ **Theorem 1.5** ([8, 10]). *The map $f : \mathbf{S}(n,d) \to \mathbf{HST}(n+2,d)$ is well defined and order preserving.*

The map f has a nice description of $d = 2$, see Figure 1.

Thomas proved that the following conjecture holds in the two- and three dimensional case, [12].

▶ **Conjecture 1.6.** *The map $f\colon \mathbf{S}(n,d) \to \mathbf{HST}(n+2,d)$ is surjective.*

Our focus will be the four dimensional case.

▶ **Theorem 1.7.** *The map $f\colon \mathbf{S}(n,4) \to \mathbf{HST}(n+2,4)$ is surjective.*

2 Background

2.1 Persistent Graphs and Pseudo Terrains

A pseudoline is a simple curve that separates the Euclidean plane into two unbounded components. We call a pseudoline x-monotone, if it is the graph of a function from the reals to the reals. A pseudoline arrangement is a collection of pseudolines such that any two pseudolines intersect in exactly one point where they intersect transversely.

▶ **Definition 2.1.** A *pseudoconfiguration of points* consists of n points with pairwise distinct x-coordinate P and a pseudoline arrangement $\mathcal{A}$, such that every pseudoline in $\mathcal{A}$ passes through exactly two points from P and for every two points x, y from P there is a unique pseudoline $\ell(x, y)$ from $\mathcal{A}$ passing through. Moreover, all the pseudolines in $\mathcal{A}$ are x-monotone.

We identify the points in P with elements of $[n]$ in the order of their x-coordinates. We refer to pseudoconfigurations of points simply as pseudoconfigurations. For a given pseudoconfiguration PP on n points, we can define the *triangle-orientation function*

$$\sigma_{PP} : \binom{[n]}{3} \to \{+, -\},\ \sigma(\{x < y < z\}) \coloneqq \begin{cases} +, \text{ if } y \text{ above } \ell(x, z) \\ -, \text{ if } y \text{ below } \ell(x, z). \end{cases}$$

A connection between Euclidean pseudoline arrangements and rank three signotopes was demonstrated in [6]. A duality between Euclidean pseudoline arrangements and pseudoconfiguration of points was shown by Agarwal and Sharir [2]. Thus, the following holds.

▶ **Theorem 2.2.** *The triangle orientation functions of pseudoconfigurations on n points are exactly the rank 3 signotopes with n elements.*

For a given PP, consider the graph on $[n]$ defined in the following way: (x, y) is an edge if and only if there is no point in PP above the segment connecting (x, y) on the pseudoline incident to x and y. We refer to this graph as the *pseudoterrain visibility graph* of PP or short *visibility graph*. Note that this graph only depends on σ_{PP}. Abello, Egecioğlu, and Kumar [1] proved that the visibility graphs are exactly the persistent graphs, defined in the following.

▶ **Definition 2.3.** A graph G on $[n]$ is *persistent* if all the following conditions are met.

1. It contains the Hamiltonian path $1, 2, \ldots, n$
2. For every edge $\{u, v\}$, where $|u - v| > 1$, there is a w with $u < w < v$, such that $\{u, w\}$ and $\{v, w\}$ are edges of G
3. If there are $a < b < c < d$ such that $\{a, c\}$ and $\{b, d\}$ are edges of G, then the edge $\{a, d\}$ is in G.

Let $\mathbf{PG}_n$ be the set of all persistent graphs on $[n]$ ordered by reverse inclusion of edges. For a given persistent graph G on $[n]$, consider the function

$$\sigma_G : \binom{[n]}{k} \to \{+, -\},\ \sigma(\{x < y < z\}) = \begin{cases} -, \exists u, v :\ x \le u < y < v \le z \text{ and } (u, v) \in E(G) \\ +, \text{else.} \end{cases}$$

The construction given by Evans and Saeedi [4] can easily be seen to be equivalent to the definition of σ_G.

▶ **Theorem 2.4** ([4])**.** *For a given persistent graph G, the function σ_G is a signotope of a pseudoconfiguration whose visibility graph is G.*

The higher Bruhat order is given by single step inclusion. It is also possible to define an inclusion order whose relation symbol is $\subseteq$. There $\sigma_1 \subseteq \sigma_2$, if $\sigma_1(x) = + \Rightarrow \sigma_2(x) = +$. It is known that these two orders are not equal for $r > 3$ [14]. Felsner and Weil proved equivalence in the $d = 3$ case.

▶ **Theorem 2.5** ([5])**.** *Let σ_1, σ_2 be two rank 3 signotopes on n elements. If $\sigma_1 \subseteq \sigma_2$ holds, then also $\sigma_1 < \sigma_2$ holds in the higher Bruhat order.*

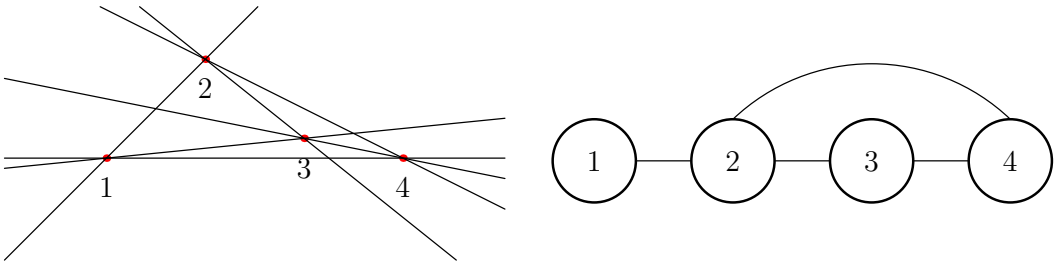


Figure 4 A pseudoconfiguration with signotope $-++-$. On the right side there is the associated visibility graph G. Note that point 2 blocks the visibility between 1 and 3. σ_G is $--++$.

Since adding edges to G implies that some $+$ change to $-$, Theorem 2.5 implies the following lemma.

▶ **Lemma 2.6.** *The map* $\mathbf{PG}_n \to \mathbf{S}(n,3), G \mapsto \sigma_G$ *is order preserving.*

There is a close connection between persistent graphs and triangulations of the cyclic polytope in three dimensions, proven by Froese and Renken.

▶ **Theorem 2.7** ([7])**.** *There is an order preserving bijection* g *from* $\mathbf{PG}_n$ *to* $\mathbf{HST}(n+2,3)$.

For $d = 3$ it was claimed in [7] that f factors in two maps. Let h be the map that takes a rank 3 signotope and maps it to the visibility graph of a corresponding pseudoconfiguration. Recall that g is an order preserving bijection from $\mathbf{PG}_n$ to $\mathbf{HST}(n+2,3)$.

▶ **Theorem 2.8** ([7])**.** *For* $d = 3$*, it holds that* $f = g \circ h$.

This can be verified by considering an alternative description of f suggested by Rambau [10] and proven by Thomas [12].

3 Proof of the Main Theorem

Proof of Theorem 1.7. Let T be a triangulation of $C(n+2,4)$. Let $T_1, T_2, \ldots, T_l$ be a decomposition of T into a chain in $\mathbf{HST}(n+2,3)$, which exist by Theorem 1.1. We examine the sequence of persistent graphs $g^{-1}(T_1), \ldots, g^{-1}(T_l)$, which exists and is a chain in $\mathbf{PG}_n$ by Theorem 2.7. For $i = 1, \ldots, l$, let $G_i = g^{-1}(T_i)$. By Lemma 2.6, the sequence of signotopes $\sigma_{G_1}, \sigma_{G_2}, \ldots, \sigma_{G_l}$ is a chain in $\mathbf{S}(n,3)$. We can complete this chain to a maximal chain $\tau_1, \tau_2, \ldots, \tau_m$ in $\mathbf{S}(n,3)$. This maximal chain corresponds to a signotope σ in $\mathbf{S}(n,4)$, according to Theorem 1.4. Note that by Theorem 2.8, $f(\sigma_{G_i}) = T_i$ for all $i = 1, \ldots, l$.

We now confirm $f(\sigma) = T$. Consider the sequence $f(\tau_1), \ldots, f(\tau_m)$. Since $\sigma_{G_1}, \ldots, \sigma_{G_l}$ is a subsequence of $\tau_1, \ldots, \tau_m$, the sequence $T_1, \ldots, T_l$ is a subsequence of $f(\tau_1), \ldots, f(\tau_m)$. Since f is order preserving and $\tau_1, \ldots, \tau_m$ is a chain, $f(\tau_1), \ldots f(\tau_m)$ is a chain after possibly removing duplicate entries. But $T_1, \ldots, T_l$ is a maximal chain and is contained in $f(\tau_1), \ldots, f(\tau_m)$, hence $f(\tau_1), \ldots f(\tau_m)$ reduces to $T_1, \ldots, T_l$ after removing duplicates. Thus, $f(\sigma)$ is the triangulation with decomposition $T_1, \ldots, T_l$, which is T.

◀

4 Discussion

We proved that the map f is surjective in the four dimensional case.

Our approach follows similar lines as the proof for $d = 3$ in [12]: for d, we make an order-preserving choice of a preimage of f. For an element of $\mathbf{HST}(n+2, d+1)$, we take a decomposition, take the aforementioned preimages, and then complete the chain to a maximal chain. Using a SAT-solver approach, we could show that for $d = 4$ there is no such choice of preimages. Thomas [12] showed that the preimage of f for $d = 3$ does not have a maximum. The construction of Evans and Saeedi [4] that we utilized gives the unique minimal element in the preimage of f.

Acknowledgments. I want to thank Stefan Felsner for bringing Conjecture 1.6 to my attention, pointing me to [7], and helpful discussions.

References

1 J. Abello, Ö. Egecioğlu, and K. Kumar. Visibility graphs of staircase polygons and the weak Bruhat order I: From visibility graphs to maximal chains. *Discrete & Computational Geometry*, 14(3):331–358, 1995.

2 P. Agarwal and M. Sharir. Pseudo-line arrangements: Duality, algorithms, and applications. *SIAM Journal on Computing*, 34(3):526–552, 2005.

3 P. Edelman and V. Reiner. The higher Stasheff-Tamari posets. *Mathematika*, 43(1):127–154, 1996.

4 W. Evans and N. Saeedi. On characterizing terrain visibility graphs. *Journal of Computational Geometry*, 6(1):108–141, Jun. 2015.

5 S. Felsner and H. Weil. A Theorem on Higher Bruhat Orders. *Discrete & Computational Geometry*, 23(1–2):121–127, 2000.

6 S. Felsner and H. Weil. Sweeps, arrangements and signotopes. *Discrete Applied Mathematics*, 109(1–2):67–94, 2001.

7 V. Froese and M. Renken. Persistent Graphs and Cyclic Polytope Triangulations. *Combinatorica*, 41(3):407–423, Jun 2021.

8 M. M. Kapranov and V. A. Voevodsky. Combinatorial-geometric aspects of polycategory theory: pasting schemes and higher Bruhat orders (list of results). *Cahiers de Topologie et Géométrie Différentielle Catégoriques*, 32:11–27, 1991.

9 Y.I. Manin and V.V Schechtmann. Higher Bruhat orders, related to the symmetric group. *Functional Analysis and Its Applications*, 20(2):148—-150, 1986.

10 J. Rambau. Triangulations of cyclic polytopes and higher Bruhat orders. *Mathematika*, 44(1):162–194, 1997.

11 J. Rambau and V. Reiner. A survey of the higher Stasheff-Tamari orders. In Folkert Müller-Hoissen, Jean Marcel Pallo, and Jim Stasheff, editors, *Associahedra, Tamari Lattices and Related Structures: Tamari Memorial Festschrift*, pages 351–390. Springer Basel, Basel, 2012.

12 Hugh Thomas. Maps between higher Bruhat orders and higher Stasheff–Tamari posets, 2002. Preprint. URL: `https://arxiv.org/abs/math/0212097`.

13 N. Williams. The two higher Stasheff–Tamari orders are equal. *Journal of the European Mathematical Society*, 2024.

14 G. Ziegler. Higher Bruhat orders and cyclic hyperplane arrangements. *Topology*, 32(2):259–279, 1993.

Partitioning the boundary of an art gallery with visibility polygons

Robert D. Barish[1] and Tetsuo Shibuya[1]

1 **Division of Medical Data Informatics, Human Genome Center, Institute of Medical Science, University of Tokyo, 4-6-1 Shirokanedai, Minato-ku, Tokyo 108-8639, Japan**
rbarish@ims.u-tokyo.ac.jp
tshibuya@hgc.jp

Abstract

We consider what we denote the Boundary Visibility Partition Problem (BVPP) of finding generating points strictly interior to an n vertex simple polygon $\mathcal{P}$ that induce a set of visibility polygons partitioning the boundary of $\mathcal{P}$ into pairwise interior-disjoint segments. In particular, we require that every point on the boundary of $\mathcal{P}$ either belongs to exactly one visibility polygon, or alternatively, is a common vertex point for all visibility polygons that contain it. Assuming the Exponential Time Hypothesis (ETH), and assuming the real RAM model of computation for operations involving geometric primitives (e.g., coordinates for vertices or generating points), we rule out a $2^{o(n)}$ time algorithm for the BVPP with rational inputs.

1 Introduction

The desire to understand the nature of visibility in a given space, or to design a space to meet bespoke visibility requirements, underlies many of the quintessential problems in computational geometry (and possibly museum curation). There is, for instance, Pólya's orchard problem [3, 10, 15] and its numerous variations (see e.g., [10, 13, 14]), which concern embeddings of geometric objects in the plane that prevent a fixed point from seeing other points past a certain distance. A still more pertinent example is the classic *Art Gallery Problem (AGP)* [5, 6, 11, 12], which asks for a minimum cardinality set of generating points for visibility polygons such that every point in a simple polygon $\mathcal{P}$ has an unobstructed line-of-sight to at least one point in this set. We remark that the decision version of the AGP is well-known to be NP-hard [11], and was eventually shown to be $\exists\mathbb{R}$-complete both with and without the restriction that generating points for visibility polygons must be embedded on the host polygon boundary [1, 2].

Recently, Stade [18] established that the decision version of the AGP remains $\exists\mathbb{R}$-complete even under the moderated constraint that only boundary points of $\mathcal{P}$ must be covered by visibility polygons. Inspired in part by this result, we consider the fine-grained complexity of a much more constrained version of this problem, where visibility polygons must be induced by generating points strictly interior to $\mathcal{P}$, and must not only cover but also partition the boundary of $\mathcal{P}$ into pairwise interior-disjoint segments:

▶ **Definition 1.1.** *Boundary Visibility Partition Problem (BVPP).* Given a simple polygon $\mathcal{P}$, the BVPP asks for any set of generating points strictly interior to $\mathcal{P}$ (i.e., interior to but not on the boundary of $\mathcal{P}$) that induce a set of visibility polygons $\mathcal{V}$, whose union covers all

This work was supported by JSPS Kakenhi grants {25K21149, 23K28035, 23K18501, 21H05052}.

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

boundary points of $\mathcal{P}$ (i.e., the perimeter of $\mathcal{P}$) under the constraint that every point along the boundary of $\mathcal{P}$ either belongs to exactly one polygon $V \in \mathcal{V}$ or is a vertex point for all visibility polygons that contain it.

To establish algorithmic lower bounds for the BVPP, we first establish algorithmic lower bounds for the following variant of the problem of partitioning weighted items into two equal weight sets:

▶ **Definition 1.2.** *Partition with Positive Rational Weights (Partition-PRW).* Letting $S = \{w_1, \ldots, w_n\}$ be a set of positive rational weights for n distinct items, with each weight $w \in S$ expressed as a ratio of two integers, the objective of the Partition-PRW problem is to partition S into two subsets X_S and Y_S under the constraint that $\sum_{w_i \in X_S}(w_i) = \sum_{w_i \in Y_S}(w_i)$.

Here, after showing that no $2^{o(n)}$ time algorithm exists for the Partition-PRW problem with n items under the *Exponential Time Hypothesis (ETH)* and the standard *word Random-Access Machine (word RAM)* model (see Lemma 3.1), with the help of a nice property for the visibility of a vertical segment through a narrow vertical aperture (see Lemma 3.2) we establish the same bound for the BVPP defined on n vertex input simple polygons with rational vertex coordinates when the *real Random-Access Machine (real RAM)* model is assumed for operations involving geometric primitives (see Theorem 3.3).

2 Preliminaries

2.1 Polygons

We observe the standard definitions for a *polygon* and a *simple polygon* (see, e.g., [12]). Recall that a polygon is *star-shaped* if it contains a non-empty set of points, denoted a *kernel*, visible to all other points in the polygon. Recall also that a point q is *strictly interior* to a polygon $\mathcal{P}$ if it lies in the interior of $\mathcal{P}$ but not along its boundaries.

2.2 Visibility and visibility polygons

Concerning the notion of *visibility*, we say that two points in a polygon $\mathcal{P}$ are *visible* to one-another (or have a *line-of-sight* to one-another) if and only if they can be connected by an open segment lying within the closure for $\mathcal{P}$. In this context, the *visibility polygon* for a *generating point* q in a simple polygon is the set of all points interior to or on the boundary of the *host* simple polygon (i.e., polygon containing the generating point) visible to q.

2.3 Exponential Time Hypothesis (ETH)

Impagliazzo & Paturi's [7] *Exponential Time Hypothesis* (ETH) is the hypothesis that, for a k-SAT instance with n variables and m clauses, where $k \geq 3$ and $s_k = inf\{\delta \ : \ k\text{-SAT}$ can be solved in $2^{(\delta \cdot n)} \cdot poly(m)$ time$\}$, it holds that $s_k > 0$. Together with the sparsification lemma [8], the ETH can be reformulated as the statement that no $2^{o(n)}$ time algorithm exists for an n variable instance of 3-SAT with $m \in \mathcal{O}(n)$ clauses.

2.4 Word RAM and Real RAM models of computation

The standard *word Random-Access Machine (word RAM)* computation model assumes $\mathcal{O}(1)$ costs for storage, comparison, addition, subtraction, multiplication, and division of "machine word sized" integers (usually assumed to have a number of bits logarithmic in the overall

problem size). For the *real Random-Access Machine (real RAM)* model of computation [16], we assume $\mathcal{O}(1)$ costs for storage, comparisons, addition, subtraction, multiplication, and division of real numbers (but not modulus or rounding operations).

3 Algorithmic lower bounds for the BVPP

▶ **Lemma 3.1.** *Assuming the ETH and word RAM model, no $2^{o(n)}$ time algorithm can exist for the Partition-PRW problem with n distinct items.*

Proof. From Wojtczak [19] we have that the Partition-PRW problem with n distinct items is strongly NP-hard, and hence, that the problem does not admit a pseudo-polynomial time algorithm unless $P = NP$. While such an analysis is not performed by Wojtczak [19], we can establish the current algorithmic lower bounds for the Partition-PRW problem by looking more carefully at the overhead for this strong NP-hardness proof. Here it suffices to note that the reduction in question is transitively from an N variable and M clause instance ϕ of a variant of 3-SAT in which we are guaranteed that no variable occurs more than four times, and that the reduction has an overall asymptotic time complexity dominated by the time to generate $N + M$ distinct prime numbers of size $\geq N + 6$. From the prime number theorem, we also have that it will suffice to generate all prime numbers over the range $[N \cdot M + 6, 2(N \cdot M)]$ for N sufficiently large to obtain the required $N + M$ distinct primes to construct weights for the $n \in \mathcal{O}(N + M)$ distinct items.

Now observe that by "Lemma 2.1" of Kanj & Szeider [9], unless the ETH is false, no $2^{o(N)}$ time algorithm exists for an N variable and $\mathcal{O}(N)$ clause variant of SAT where each variable occurs at most three times, or using clause padding, an N variable and $\mathcal{O}(N)$ clause variant of 3-SAT where each variable occurs at most four times. In addition, observe that we can use the Sieve of Atkin [4] (which improves on the Sieve of Eratosthenes [17]) to find all prime numbers of size at most ζ in at most $\mathcal{O}\left(\zeta \cdot (\ln(\ln(\zeta)))^{-1}\right)$ time. Accordingly, we are able to reduce an N variable and $\mathcal{O}(N)$ clause instance of 3-SAT to an instance of the Partition-PRW problem with $n \in \mathcal{O}(N)$ weights in polynomial time. Thus, no $2^{o(n)}$ time algorithm can exist for the Partition-PRW problem unless the ETH is false. ◀

▶ **Lemma 3.2.** *If we let $v = (x, y)$ be the generating point for a visibility polygon V, and let $q \in \mathbb{R}$ (where $0 < q < x$) be the x-coordinate for a perfectly opaque vertical line containing a contiguous aperture (i.e., gap) of length $L_1 > 0$, then the length of the edge of V falling along a perfectly opaque vertical line at $x = 0$ will be $L_2 = L_1 \cdot \left(\frac{x}{x-q}\right)$.*

Proof. Let $v = (x, y)$, V, and q be as stated. In addition, let T_1 be a triangle induced by (x, y) and the two points bordering the aperture at $x = q$, and let T_2 be the triangle induced by (x, y) and the two vertex points of V at $x = 0$. Observe that, because V must contain only points that can be connected to v by a line segment passing through the aperture at $x = q$, T_1 and T_2 will be similar triangles. Accordingly, we have that $\frac{L_2}{x} = \frac{L_1}{x-q}$, which yields the current lemma. ◀ ◀

▶ **Theorem 3.3.** *For input n vertex simple polygons with rational vertex coordinates, assuming the ETH and the real RAM model for geometric primitives, no $2^{o(n)}$ time algorithm can exist for the BVPP.*

Proof. We proceed by giving a linear time reduction from an N item instance of the Partition-PRW problem, which by Lemma 3.1 will establish the current theorem. We assume the real RAM model for all operations involving geometric primitives (e.g., storing, retrieving, and

performing basic arithmetic operations on coordinates for vertices or generating points) in the specific context of the BVPP, and otherwise assume the word RAM model.

At a high level of abstraction, the idea is to create a polygon $\mathcal{P}$ serving as input for an instance of the BVPP, where we will use what we denote as a *torch gadget* to force the placement of generating points for visibility polygons at six specific locations. With the exception of two parallel and non-adjacent boundary lines along the left-hand-side of $\mathcal{P}$ – where we refer to the bottom and top instances of these boundary lines as the *bottom projection screen* and *top projection screen*, respectively – all other boundary points for $\mathcal{P}$ will fall along the boundary of exactly one of these visibility polygons, or else, correspond to a boundary point shared by exactly two of these visibility polygons.

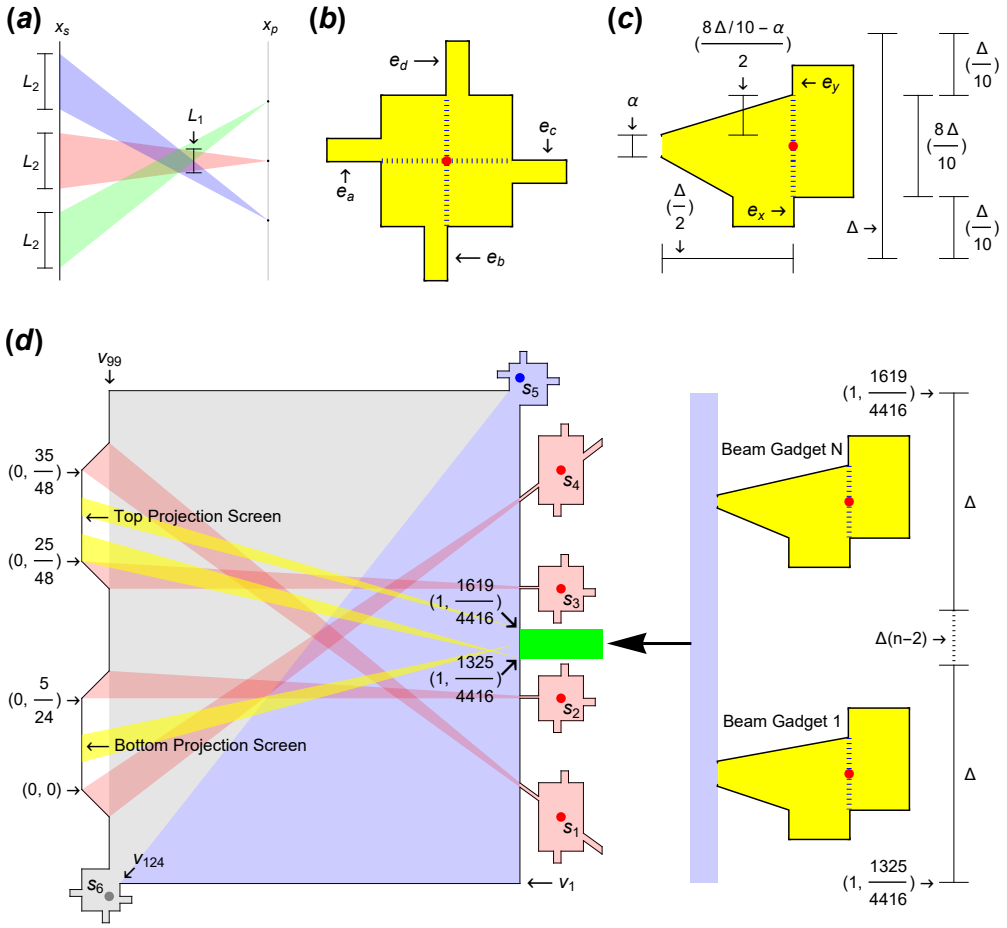


Figure 1 Illustrations to accompany the Theorem 3.3 proof argument; see the proof argument for further details.

Next, for each item of weight $w \in S$, we place a distinct instance of what we refer to as a *beam gadget* along the right-hand-side wall of $\mathcal{P}$, where this gadget will force a single visibility polygon generating point to be placed along a specified vertical line segment. This generating point will then induce a visibility polygon having a boundary overlapping the bottom or top projection screen of length proportional to the weight $w \in S$, and by moving the generating point along the extent of the aforementioned vertical line, this boundary can

be positioned anywhere along the bottom or top projection screens.

For visual intuition concerning this last point, we refer the reader to Fig. 1(a), wherein we show three visibility polygons (colored green, red, and blue) "radiating outwards" from narrow apertures or "holes" on the right-hand-side of the illustration, and where these apertures are constructed from perfectly vertical 1-dimensional lines (at the x-coordinate x_p). Noting that each visibility polygon has extent L_1 at a common horizontal position, we can make the general observation that, regardless of their angle with respect to the x-axis, all three visibility polygons will have a boundary of the same length L_2 on the left-hand-side wall (at the x-coordinate x_s). In particular, this is exactly maps to the scenario from Lemma 3.2 if we let $x_s = 0$ and consider the aperture in the vertical opaque line segment $x = q$ from Lemma 3.2 to be the common length L_1 vertical line for the green, red, and blue visibility polygons in Fig. 1(a). Here, we can observe that the relation $\frac{L_2}{x} = \frac{L_1}{x-q}$ we establish in Lemma 3.2 has no dependence on the y-coordinate for the visibility polygon propagating through the length L_1 aperture.

The idea is now that we can normalize the weights in S when encoding them as instances of beam gadgets to ensure that their sum equals the combined lengths of the bottom and top projection screens. This will then ensure the problem of splitting the items of S into two equal weight subsets, $S'_1 \subseteq S$ and $S'_2 \subseteq S$, is equivalent to the problem of finding generating point coordinates along each beam gadget line segment to ensure that all points along the bottom and top projection screens fall along the boundary of some visibility polygon.

To get into the specific details, we refer the reader to Fig. 1(b) for a generalized illustration of a torch gadget, where the rectangular indentations on the boundary of the polygon force the existence of a generating point for a visibility polygon (red dot) at the intersection of the two (blue dashed) lines. In particular, for some rotation or reflection of the torch gadget shown in Fig. 1(b), we require that: (t1) the line segments labeled e_a and e_c are parallel; (t2) the line segments labeled e_b and e_d are parallel; and (t3) e_a, e_b, e_c, and e_d are the bottom, right, top, and left borders, respectively, of rectangle protrusions from a rectangular central chamber. Here, requirements (t1) through (t3) will then force the torch gadget to be star shaped, having a kernel consisting of a single point exactly at the indicated intersection of the two (blue dashed) lines in Fig. 1(b). Under the constraints of the BVPP, these requirements will also ensure that this kernel is the only possible location for the placement of a visibility polygon generating point, as it will otherwise be impossible to cleanly partition the perimeter of the central rectangular chamber.

We also refer the reader to Fig. 1(c) for an illustration of the beam gadget – centered on a vertical slot of length Δ, with an "aperture" width of α, and with the other specified dimensions – where the geometry of the gadget forces the existence of a single generating point for a visibility polygon falling anywhere along the (dashed blue) line aside from its endpoints. In particular, as the segments labeled e_x and e_y are required to be parallel in any beam gadget, under the constraints of the BVPP, the only way to ensure a clean partitioning of the boundary of the beam gadget is to place a single visibility polygon generating point along the (dashed blue) line in Fig. 1(c) connecting these segments.

Our full construction can be seen in Fig. 1(d) where six instances of the torch gadget force the existence of six generating points – illustrated with colored dots labeled $\{s_1, s_2, s_3, s_4, s_5, s_6\}$ – for visibility polygons covering all extreme points of the illustrated host polygon $\mathcal{P}$ under the intersection constraints enforced by the BVPP, with the sole exception of the edges labeled as corresponding to the bottom and top projection screens. For clarity, we use red, gray, and blue colors to disambiguate the visibility polygons for the six torch gadget generating points, and color each visibility polygon in accordance with the

color of its generating point.

Here, each of the N beam gadgets is placed along the section of the host polygon $\mathcal{P}$ boundary between the points $\left(1, \frac{1325}{4416}\right)$ and $\left(1, \frac{1619}{4416}\right)$ – along the left-hand-side of the (solid green) block, which also serves to illustrate that the beam gadgets can have arbitrary rightward horizontal lengths – and as shown on the right-hand-side of Fig. 1(d), each takes up a "slot" of vertical extent $\Delta = N^{-1} \cdot \left(\left(\frac{1619}{4416}\right) - \left(\frac{1325}{4416}\right)\right) = \left(\frac{49}{736 \cdot N}\right)$. Next, letting $M = \sum_{w_i \in S} (w_i)$, for each item of weight $w \in S$, we let $w' = w \cdot \left(\frac{5}{12 \cdot M}\right)$, and subsequently specify $\alpha = \left(\frac{49 \cdot w'}{1472N + 49}\right)$ for a distinct instance of one of the N total beam gadgets. Noting that the bottom and top projection screens have a combined horizontal extent of $\left(\frac{5}{12}\right)$, and invoking Lemma 3.2, we can observe that: (1) this will ensure the combined vertical extent of all beam gadget produced visibility polygons at the same x-coordinate as the bottom and top projection screens will also equal $\left(\frac{5}{12}\right)$; and (2) that the host polygon we have constructed will have $n \in \mathcal{O}(N)$ total vertices.

Finally, let $\mathcal{P}'$ be a simplified 124 vertex instance of the host polygon $\mathcal{P}$ without the N beam gadgets, where we consider the boundary between the points $\left(1, \frac{1325}{4416}\right)$ and $\left(1, \frac{1619}{4416}\right)$ to simply be a part of an edge between the torch gadgets hosting visibility polygon generating points s_2 and s_3. Due to space limitations, we ask the reader to verify for themselves that it is possible to embed $\mathcal{P}'$ in the plane such that all vertices have rational coordinates. It remains to observe that the geometric operations used to construct the N beam gadgets for $\mathcal{P}$ are closed under the rationals. ◀

References

1. M. Abrahamsen, A. Adamaszek, and T. Miltzow. The art gallery problem is ∃ℝ-complete. *Proc. 50th STOC*, pages 65–73, 2018. doi:10.1145/3188745.3188868.
2. M. Abrahamsen, A. Adamaszek, and T. Miltzow. The art gallery problem is ∃ℝ-complete. *J. ACM*, 69(1):4:1–4:70, 2021. doi:10.1145/3486220.
3. T. T. Allen. Polya's orchard problem. *Amer. Math. Monthly*, 93(2):98–104, 1986. doi:10.2307/2322700.
4. A. O. L. Atkin and D. J. Bernstein. Prime sieves using binary quadratic forms. *Math. Comput.*, 73(246):1023–1030, 2004. doi:10.1090/S0025-5718-03-01501-1.
5. V. Chvátal. A combinatorial theorem in plane geometry. *J. Comb. Theory Ser. B*, 18(1):39–41, 1975. doi:10.1016/0095-8956(75)90061-1.
6. S. Fisk. A short proof of Chvátal's watchman theorem. *J. Comb. Theory Ser. B*, 24(3):374, 1978. doi:10.1016/0095-8956(78)90059-X.
7. R. Impagliazzo and R. Paturi. On the complexity of k-SAT. *J. Comput. Syst. Sci.*, 62(2):367–375, 2001.
8. R. Impagliazzo, R. Paturi, and F. Zane. Which problems have strongly exponential complexity? *J. Comput. Syst. Sci.*, 63(4):512–530, 2001. doi:10.1006/jcss.2001.1774.
9. I. Kanj and S. Szeider. Parameterized and subexponential-time complexity of satisfiability problems and applications. *Theoret. Comput. Sci.*, 607(Part 3):282–295, 2015. doi:10.1016/j.tcs.2015.08.029.
10. C. P. Kruskal. The orchard visibility problem and some variants. *J. Comput. Syst. Sci.*, 74(4):587–597, 2008. doi:10.1016/j.jcss.2007.06.004.
11. D. Lee and A. Lin. Computational complexity of art gallery problems. *IEEE Trans. Inf. Theory*, 32(2):276–282, 1986. doi:10.1109/TIT.1986.1057165.
12. J. O'Rourke. *Art gallery theorems and algorithms*. Oxford University Press: New York, NY, 1987.

13 J. O'Rourke. Efficient visibility blockers in Polya's orchard problem. MathOverflow, 2011. https://mathoverflow.net/q/62922 (version: 2017-03-31).

14 J. O'Rourke. A random variation on Polya's orchard problem. MathOverflow, 2014. https://mathoverflow.net/q/179749 (version: 2017-04-13).

15 G. Pólya. *Zahlentheoretisches und wahrscheinlichkeitstheoretisches über die sichtweite im walde*, volume 27. 1918.

16 M. I. Shamos. Computational geometry. *Ph.D. dissertation, Yale University*, pages 1–236, 1978.

17 J. Sorenson. An introduction to prime number sieves. *Tech. Rep. #909, Department of Computer Science, University of Wisconsin-Madison (Madison, WI)*, pages 1–13, 1990. http://digital.library.wisc.edu/1793/59248.

18 J. Stade. The point-boundary art gallery problem is $\exists\mathbb{R}$-hard. *Proc. 41st SoCG*, pages 74:1–74:23, 2025. doi:10.4230/LIPIcs.SoCG.2025.74.

19 D. Wojtczak. On strong NP-completeness of rational problems. *Proc. 13th CSR*, pages 308–320, 2018. doi:10.1007/978-3-319-90530-3_26.

Fault-Tolerance and Oriented Dilation of the Greedy Triangulation

Prosenjit Bose[1], Kevin Buchin[2], and Antonia Kalb[2]

1 **Carleton University, Ottawa, Canada**
jit@scs.carleton.ca
2 **TU Dortmund University, Germany**
kevin.buchin, antonia.kalb@tu-dortmund.de

Abstract

An undirected graph $G = (P, E)$ is a *k-edge fault-tolerant t-spanner*, when for every edge set $E' \subset E$ of size $|E'| = k$, the shortest path in $G \setminus E'$ between any pair of points is at most a factor t longer than their shortest path in $K(P) \setminus E'$, where $K(P)$ denotes the complete undirected graph on P. An oriented graph $\overrightarrow{G}$ is an *oriented t-spanner* on a point set P contains for each pair $p, q \in P$ a shortest closed walk through p and q of length of at most a factor t longer than their shortest cycle in $K(P)$.

We present the first results on plane fault-tolerant spanners: For n points in the Euclidean plane, the greedy triangulation is a 1-edge fault-tolerant $\mathcal{O}(n)$-spanner for arbitrary point sets and an $\mathcal{O}(1)$-spanner for convex point sets. Further, we extend previous results on plane oriented spanners for convex point sets by orienting the greedy triangulation yielding a $\mathcal{O}(n)$-spanner in general.

1 Introduction

A geometric spanner offers an efficient approximation of the complete graph, maintaining short paths between pairs of points with only a small subset of its edges. The quality of this approximation is measured by the *dilation* of a graph $G = (P, E)$, defined as the maximum ratio of the shortest path length $|d_G(p, q)|$ to their Euclidean distance $|pq|$ over all $p, q \in P$. In addition to small dilation, many applications require further structural properties such as planarity [5], directions [2, 7] or resilience to the failure of connections [17, 18].

In this paper, we analyze and correlate two spanner models: *oriented spanners*, where each edge is a one way connection, and *fault-tolerant spanners*, which maintain short paths even faced with the failure of a fixed number of arbitrary connections. Until now, for both models no plane constructions were known for arbitrary point sets. This paper reveals new properties of the greedy triangulation. We show that it is a fault-tolerant spanner under the failure of one edge, with dilation linear in the number of points, and that an appropriate orientation yields an oriented spanner with the same dilation.

1.1 Definitions and Known Results

Notations We use $|\Delta_{pqr}|$ to denote the perimeter of the triangle spanned by the points p, q, r. If p and q are distinct points in P, then $\Delta^*(p, q)$ denotes a shortest cycle through p and q in the complete undirected graph $K(P)$ on P. Since this paper considers point sets in the Euclidean plane, this is a minimum perimeter triangle in P containing p and q. For an edge pq of a triangulation T, let $\Delta_T(pq)$ be a minimum perimeter face of T adjacent to pq.

Constant Dilation Triangulations Das and Joseph [9] proved that every triangulation that fulfills the α-diamond-property has constant dilation.

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

▶ **Definition.** *(α-diamond property) Let α be a real number with $0 < \alpha < \pi/2$. Let $G = (P, E)$ be a plane undirected graph. We say that G* fulfills the α-diamond property*, when for each edge $e \in E$ at least one of the two isosceles triangles with base edge e and base angle α, does not contain any points of P.*

The currently best upper bound on the dilation t of such a triangulation is $t \leq \frac{8(\pi-\alpha)^2}{\alpha^2 \sin^2(\alpha/4)}$ [4]. The prominent triangulations that fulfill the α-diamond property are the Delaunay triangulation for $\alpha = \pi/4$ [13], the minimum weight triangulation for $\alpha = \pi/4.6$ [12] and the greedy triangulation for $\alpha = \pi/6$ [4].

Greedy Triangulation The *greedy triangulation* of a point set is obtained by considering all pairs of points by increasing distance, and by adding the straight-line edge if it does not intersect any of the edges already added. Computing the greedy triangulation for a set of n non-convex points takes $\mathcal{O}(n \log n)$ time. For a convex point set, if the order of the points along the convex hull is given, this can be done in linear time [14].

Fault-Tolerant Spanners In communication as well as road networks, the system should be resilient to edge failures, e.g. a connection is turned off or a road is closed for construction. Despite the failure of one or more vertices or edges, the shortest path between any two vertices should be short relative to the shortest path possible "under the circumstances" [18].

▶ **Definition** (Fault-tolerant spanner)**.** *Given $k \in \mathbb{N}$, a graph $G = (P, E)$ is called a k-edge fault-tolerant t-spanner if*

$$t = \max_{\substack{E' \subseteq E,\ |E'| \leq k \\ p,q \in P}} \frac{|d_{G \setminus E'}(p, q)|}{|d_{K(P) \setminus E'}(p, q)|},$$

where $G \setminus E'$ is the graph $G' = (P, E \setminus E')$.

Since their introduction in 1998 [15], fault-tolerant spanners have gained a lot of interest [1, 3, 19]. Lukovszki [17] proved that every spanner that is resilient to vertex faults is also resilient to edge faults. There is a simple algorithm to transform any t-spanner into a k-fault-tolerant t-spanner [18]. Further, for a fixed k, we can compute a k-fault-tolerant spanner with a linear number of edges [15, 16]. However, none of these constructions guarantees a plane graph.

Oriented Spanners While geometric spanners have been researched for decades, directed versions have only been considered more recently [2, 7]. This is surprising since, in many applications, edges may be directed or even oriented if two-way connections are not permitted. Examples for one-way networks include transportation infrastructures with one-way streets or rail tracks, air traffic with predefined unidirectional airways and communication systems enforcing asymmetric data transmission to reduce interference or enhance security.

▶ **Definition** (Oriented spanner)**.** *Let $\overrightarrow{G}$ be an oriented graph with vertex set P. The* oriented dilation *of two distinct points $p, q \in P$ is defined as*

$$\mathrm{odil}_{\overrightarrow{G}}(p, q) = \frac{|C_{\overrightarrow{G}}(p, q)|}{|\Delta^*(p, q)|}.$$

where $C_{\overrightarrow{G}}(p, q)$ is the shortest closed walk in $\overrightarrow{G}$ through p and q. We call $\overrightarrow{G}$ an oriented t-spanner *if $t = \max_{p,q \in P} \mathrm{odil}_{\overrightarrow{G}}(p, q)$.*

While there are several results on oriented spanners with a linear number of edges [7, 8], plane oriented spanners have only been considered in [7]. The authors showed that computing a plane oriented spanner on a given point set with minimum dilation is at least as hard as computing the minimum dilation triangulation; its hardness is still open [10, 11], but it is conjectured to be NP-hard [6]. For convex point sets, they offered an orientation of the greedy triangulation that yields an oriented $\mathcal{O}(1)$-spanner. Further, they presented a non-convex point set, where any orientation of the greedy triangulation is an oriented $\Omega(n)$-spanner.

1.2 Contributions

Since every plane spanner has at most $3n-6$ edges, any triangulation is at most 5-connected. Thus, the question of the existence of a plane k-edge fault-tolerant spanner is meaningful only for $k < 5$. For convex points, since every triangulation contains a vertex of degree 2, dilation under the failure of $k > 1$ edges is unbounded. Therefore, in this paper, we restrict the problem to spanners resilient to exact one edge failure. The challenges presented by even $k = 1$ highlight the complexity of this problem.

Oriented and 1-edge fault-tolerant spanners are quite related models: Both, the dilation under the failure of an edge pq and the oriented dilation of two points p, q, depend on the perimeter of the smallest triangle of p and q in P.

Thus, in Section 2, we bound the perimeter of a face of a greedy triangulation by the minimum perimeter triangle of two points which are adjacent to this face. In Section 3, we use this bound to show that the greedy triangulation is a 1-edge fault-tolerant $\mathcal{O}(n)$-spanner. For convex point sets, its dilation is constant. In Section 4, we present an orientation of the greedy triangulation that yields an oriented $\mathcal{O}(n)$-spanner for arbitrary point sets.

2 Bounding the Perimeter of a Face of the Greedy Triangulation

In [7], the authors oriented the greedy triangulation to obtain an oriented $\mathcal{O}(1)$-spanner for convex point sets. They bound the oriented dilation of a pair p, q with $pq \in E$ as follows:

1. They show the existence of a set of "blockers" $b_1, \dots, b_k$ where pb_i *blocks* qb_{i-1}, i.e. pb_i is the shortest edge in E intersecting pq. They choose b_0 s.t. $pb_0 \in E$ and $|pb_0| = \mathcal{O}(|pq|)$.
2. They show that the length of the blockers is increasing linearly in the number of blockers.
3. By showing that there is only a constant number of blockers of length $\mathcal{O}(|pq|)$, they bound the length of the last blocker pb_k by $\mathcal{O}(|pq|)$.
4. They present an orientation $\overrightarrow{T}$ of T such that $|C_{\overrightarrow{T}}(p, q)| \leq |\Delta_{pqb_k}|$.

We generalize and extend their results for non-convex point sets (compare to Figure 1):

- Due to convexity, all blockers in Step 1 are adjacent to p. We define a different sequence adjacent to p whose edges satisfy the bounds as in Step 2, even for non-convex point sets.
- We show that the length of the last blocker pb_k is linear in the degree of p.

▶ **Lemma 2.1.** *Let $T = (P, E)$ be a greedy triangulation and $pq \in E$. It holds*

$$|\Delta_T(pq)| = \mathcal{O}(\deg_T(p) + \deg_T(q)) \cdot |\Delta^*(p, q)|.$$

Lemma 2.1 bounds the perimeter of $\Delta_T(pq)$ by a multiple of the minimum perimeter triangle of p and q. As this factor depends on the degree of p and q, it can be linear in the number of points. Analog to Step 3 in [7], we show a constant bound for convex point sets:

▶ **Lemma 2.2.** *Let $T = (P, E)$ be a greedy triangulation of a convex point set P and $pq \in E$. It holds $|\Delta_T(pq)| = \mathcal{O}(|pq|)$.*

An equivalent of Step 4 is done in Section 4 leading to a plane oriented $\mathcal{O}(n)$-spanner.

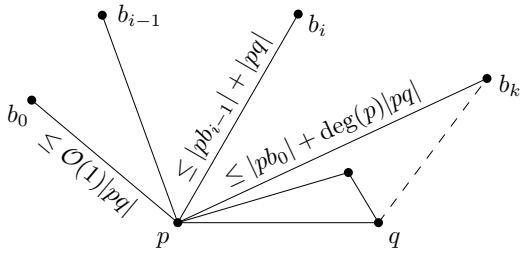


Figure 1 Sketch of the proof of Lemma 2.1: We define a sequence of points $b_1, \ldots, b_k$, such that $pb_i \in E$ and $|pb_i| \leq |pq| + |pb_{i-1}|$. The perimeter of the triangle Δ_{pqb_k} upper bounds $|\Delta_T(pq)|$.

3 Plane 1-Edge Fault-Tolerant Spanners

To analyze the 1-edge fault-tolerance of a triangulation, we use the following statement:

▶ **Lemma 3.1.** *Let $T = (P, E)$ be a triangulation that fulfills the α-diamond property. The dilation t of T under the failure of an edge $pq \in E$ is $t = \mathcal{O}\left(\frac{|\Delta_T(pq)|}{|\Delta^*(p,q)|}\right)$.*

Proof. Let $\varphi = \frac{|\Delta_T(pq)|}{|\Delta^*(p,q)|}$. Note that $\varphi \geq 1$.

The dilation of the pair pq under the failure of pq is

$$\frac{|d_{T\setminus pq}(p,q)|}{|d_{K(P))\setminus pq}(p,q)|} = \frac{|\Delta_T(pq)| - |pq|}{\min\limits_{r\in P\setminus\{p,q\}} |pr| + |rq|} \leq \frac{|\Delta_T(pq)|}{|\Delta^*(p,q)| - |pq|} = \frac{\varphi|\Delta^*(p,q)|}{|\Delta^*(p,q)| - |pq|} \leq 2\varphi.$$

Now, we bound the dilation of a pair $p'q' \neq pq$ under the failure of pq, which is $\frac{|d_{T\setminus pq}(p',q')|}{|p'q'|}$.

Let $d_T(p', q')$ be a shortest path from p' to q' in T. Due to α-diamond property, it holds $|d_T(p', q')| = \mathcal{O}(1) \cdot |p'q'|$.

If $pq \notin d_T(p', q')$, then we have $|d_T(p', q')| = \mathcal{O}(|p'q'|) \leq \varphi \cdot \mathcal{O}(|p'q'|)$.

If $pq \in d_T(p', q')$, w.l.o.g. we assume that this path from p' to q' visits p before q. So, we have $|d_T(p', q')| = |d_T(p', p)| + |pq| + |d_T(q, q')|$. This give us

$$|d_{T\setminus pq}(p', q')| \leq |d_T(p', p)| + |\Delta_T(pq)| - |pq| + |d_T(q, q')| \leq |d_T(p', q')| + \varphi|\Delta^*(p, q)|.$$

Since $\Delta^*(p, q)$ is the minimum perimeter triangle of p and q, it holds that

$$|\Delta^*(p, q)| \leq |\Delta_{pqq'}| \leq 2(|pq| + |qq'|) \leq 2(|pq| + |d_G(q, q')|) \leq 2|d_T(p', q')| = \mathcal{O}(|p'q'|).$$

Thus, the dilation of the pair $p'q'$ under the failure of pq is

$$\frac{|d_{T\setminus pq}(p', q')|}{|p'q'|} \leq \frac{|d_T(p', q')| + \varphi|\Delta^*(p, q)|}{|p'q'|} = \frac{\mathcal{O}(|p'q'|) + \varphi \cdot \mathcal{O}(|p'q'|)}{|p'q'|} = \mathcal{O}(1) \cdot \varphi. \qquad \blacktriangleleft$$

As the greedy triangulation fulfills the α-diamond property and the maximum degree is $n - 1$, combining Lemmas 2.1 and 3.1 gives us:

▶ **Theorem 3.2.** *The greedy triangulation is a 1-edge fault-tolerant $\mathcal{O}(n)$-spanner.*

We complement this result with a lower bound of $\Omega(n)$: The greedy triangulation given in [7, Figure 18], where every orientation has oriented dilation$\Omega(n)$, is also a 1-edge fault-tolerant $\Omega(n)$-spanner.

Combining Lemmas 2.2 and 3.1 leads to a constant dilation for convex point sets:

▶ **Theorem 3.3.** *The greedy triangulation is a 1-edge fault-tolerant $\mathcal{O}(1)$-spanner for convex point sets.*

4 Plane Oriented Spanners

In [7], the authors showed that there is an orientation of the greedy triangulation that yields an oriented $\mathcal{O}(1)$-spanner for convex point sets. Further, they presented a non-convex point set, where any orientation of the greedy triangulation of this point set is an oriented $\Omega(n)$-spanner. We complement these results with a linear upper bound.

The following greedy algorithm orients an undirected triangulation such that the resulting orientation is a 2-approximation relatively to the input.

1. Sort the $O(n)$ triangles of the triangulation in ascending order by their perimeter.
2. In this order, orient each triangle clockwise or anti-clockwise if the already oriented edges admit it; otherwise, we skip this triangle.
3. At the end of the algorithm, orient all remaining edges arbitrarily.

This algorithm is a modification of an algorithm from [7] which was used to achieve a 2-spanner by orienting a complete graph. Adapting the proof in [7] we can show:

▶ **Lemma 4.1.** *Let $T = (P, E)$ be a triangulation on n points. In $\mathcal{O}(n \log n)$ time, we can compute an orientation $\overrightarrow{T}$ of T, such that for each $pq \in E$ holds $|C_{\overrightarrow{T}}(p, q)| \leq 2|\Delta_T(pq)|$.*

Orienting a greedy triangulation with this algorithm yields in an oriented $\mathcal{O}(n)$-spanner:

▶ **Theorem 4.2.** *Given a set P of n points, a plane oriented $\mathcal{O}(n)$-spanner on P can be computed in $\mathcal{O}(n \log n)$ time.*

Proof. Let $T = (P, E)$ be the greedy triangulation of P and $\overrightarrow{T}$ its orientation computed by the greedy algorithm described above, given the faces of T as input triangles. Computing $\overrightarrow{T}$ takes $\mathcal{O}(n \log n)$ time. For two distinct points $p, q \in P$, we show that $\operatorname{odil}(p, q) = \mathcal{O}(n)$.

For $pq \in E$, as the degree of any point is in $\mathcal{O}(n)$, combining Lemmas 2.1 and 4.1 gives

$$|C_{\overrightarrow{T}}(p, q)| \leq 2|\Delta_T(pq)| = \mathcal{O}(n) \cdot |\Delta^*(p, q)|.$$

For $pq \notin E$, we consider a shortest (undirected) path $d_T(p, q)$ from p to q in T. Let $p_0, p_1, \ldots, p_k$ be the points along this path with $p_0 = p$ and $p_k = q$. Due to Lemma 4.1, for each edge $p_i p_{i+1} \in E$ along this path exists a shortest closed walk in $\overrightarrow{T}$ of length $|C_{\overrightarrow{T}}(p_i, p_{i+1})| \leq 2|\Delta_T(p_i p_{i+1})|$. The union of all these closed walks along the path $d_T(p, q)$ is a closed walk containing p and q (compare to Figure 2). Applying Lemma 2.1 bounds

$$\begin{aligned} |C_{\overrightarrow{T}}(p, q)| &\leq \sum_{p_i p_{i+1} \in d_T(p,q)} |C_{\overrightarrow{T}}(p_i, p_{i+1})| \leq \sum_{p_i p_{i+1} \in d_T(p,q)} 2|\Delta_T(p_i p_{i+1})| \\ &= \sum_{p_i p_{i+1} \in d_T(p,q)} \mathcal{O}(\deg(p_i) + \deg(p_{i+1})) \cdot |\Delta^*(p_i, p_{i+1})|. \end{aligned}$$

For all $1 \leq i \leq k - 1$, we have

$$|\Delta^*(p_i, p_{i+1})| \leq |\Delta_{p_{i-1} p_i p_{i+1}}| \leq 2|p_{i-1} p_i| + 2|p_i p_{i+1}| \leq 2|d_T(p, q)|.$$

Since the greedy triangulation fulfills the α-diamond property [9], it holds that $|d_T(p, q)| = \mathcal{O}(1) \cdot |pq|$. We summarize

$$\begin{aligned} |C_{\overrightarrow{T}}(p, q)| &\leq \sum_{p_i p_{i+1} \in d_T(p,q)} \mathcal{O}(\deg(p_i) + \deg(p_{i+1})) \cdot |\Delta^*(p_i, p_{i+1})| \\ &\leq \sum_{p_i p_{i+1} \in d_T(p,q)} \mathcal{O}(\deg(p_i) + \deg(p_{i+1})) \cdot 2|d_T(p, q)| \\ &\leq \sum_{p_i \in d_T(p,q)} \mathcal{O}(\deg(p_i)) \cdot |pq| \leq |pq| \cdot \sum_{p' \in P} \mathcal{O}(\deg(p')) = \mathcal{O}(|E|) \cdot |pq|, \end{aligned}$$

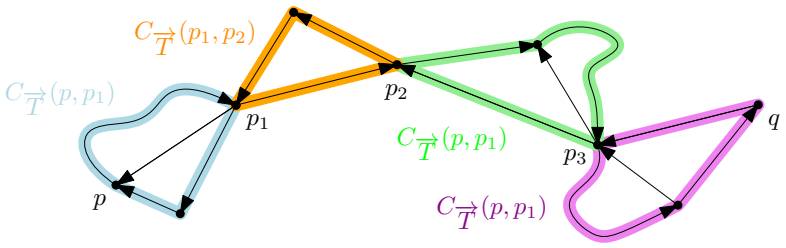


Figure 2 Sketch of the proof of Theorem 4.2. We orient T such that for each undirected triangular face of T exists a shortest closed walk in the orientation $\overrightarrow{T}$ of bounded length. The union of these closed walks along the undirected path $d_T(p, q)$ is a closed walk through p and q.

where the last step uses the handshaking lemma. As $|E| = \mathcal{O}(n)$ for triangulations, we have

$$\operatorname{odil}(p, q) = \frac{|C_{\overrightarrow{T}(p,q)}|}{|\Delta^*(p, q)|} \leq \frac{\mathcal{O}(n) \cdot |pq|}{|\Delta^*(p, q)|} = \mathcal{O}(n).$$ ◀

5 Open Questions

In this paper, we gained new insights on the greedy triangulation. We bounded the perimeter of a face adjacent to an edge pq in the triangulation by a multiple of the minimum perimeter triangle in P containing p and q, where the factor is linear in the degree of p and q. For convex point sets, it is bounded by a constant factor of the length of pq.

We used this property to achieve new results on computing plane fault-tolerant and oriented spanners. It is open whether this property is useful for further models, for example spanners, where we consider the edge whose failure minimizes the dilation.

We proved that the greedy triangulation leads to a 1-edge fault-tolerant $\mathcal{O}(n)$-spanner and an oriented $\mathcal{O}(n)$-spanner. The question arises whether other triangulations that fulfill the α-diamond property have linear dilation. The same point sets that provide lower bounds on the oriented dilation of the minimum weight, Delaunay and minimum dilation triangulation (see [7]) also lead to lower bounds on the fault-tolerance of these triangulations. Thus, we conclude that the α-diamond property is not sufficient for plane fault-tolerant spanners. This raises the question of whether plane fault-tolerant or oriented spanners with constant dilation even exist in general. If not, what is the lowest dilation that we can guarantee?

Acknowledgments. We thank Carolin Rehs and Sampson Wong for helpful discussions.

References

1 Mohammad Ali Abam, Mark de Berg, Mohammad Farshi, and Joachim Gudmundsson. Region-fault tolerant geometric spanners. *Discrete Comput. Geom.*, 41(4):556–582, 2009. Also at SODA 2007. doi:10.1007/S00454-009-9137-7.

2 Hugo A. Akitaya, Ahmad Biniaz, and Prosenjit Bose. On the spanning and routing ratios of the directed Θ_6-graph. *J. Comput. Geom.*, 105-106:101881, 2022. Also at WADS 2021. doi:10.1016/J.COMGEO.2022.101881.

3 Greg Bodwin, Bernhard Haeupler, and Merav Parter. Fault-tolerant spanners against bounded-degree edge failures: Linearly more faults, almost for free. In *Annu. ACM-SIAM Sympos. Discrete Algorithms (SODA)*, pages 2609–2642, 2024. doi:10.1137/1.9781611977912.93.

4 Prosenjit Bose, Aaron Lee, and Michiel H. M. Smid. On generalized diamond spanners. In *Proc. 10th Internat. Sympos. Algorithms and Data Struct.*, volume 4619 of *Lecture Notes in Computer Science*, pages 325–336. Springer-Verlag, 2007. doi:10.1007/978-3-540-73951-7_29.

5 Prosenjit Bose and Michiel Smid. On plane geometric spanners: A survey and open problems. *Comput. Geom. Theory Appl.*, 46(7):818–830, 2013. doi:10.1016/j.comgeo.2013.04.002.

6 Aléx Brandt, Miguel Gaiowski, Cid de Souza, and Pedro de Rezende. Minimum dilation triangulation: Reaching optimality efficiently. In *Proc. 26th Canad. Conf. Comput. Geom. (CCCG)*, 2014.

7 Kevin Buchin, Joachim Gudmundsson, Antonia Kalb, Aleksandr Popov, Carolin Rehs, André van Renssen, and Sampson Wong. Oriented spanners. *Algorithmica*, 88(1), 2026. Also at ESA 2023. doi:10.1007/S00453-025-01342-8.

8 Kevin Buchin, Antonia Kalb, Anil Maheshwari, Saeed Odak, Carolin Rehs, Michiel Smid, and Sampson Wong. Computing Oriented Spanners and Their Dilation. In Oswin Aichholzer and Haitao Wang, editors, *Proc. 41st Internat. Sympos. Comput. Geom. (SoCG)*, volume 332 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 27:1–27:17. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2025. doi:10.4230/LIPIcs.SoCG.2025.27.

9 Gautam Das and Deborah Joseph. Which triangulations approximate the complete graph? In *Int. Symp. on Optimal Algorithms (OA)*, volume 401 of *Lecture Notes in Computer Science*, pages 168–192. Springer-Verlag, 1989. doi:10.1007/3-540-51859-2_15.

10 David Eppstein. Spanning trees and spanners. In *Handbook of Computational Geometry*, pages 425–461. Elsevier, 2000. doi:h10.1016/B978-044482537-7/50010-3.

11 Panos Giannopoulos, Rolf Klein, Christian Knauer, Martin Kutz, and Dániel Marx. Computing geometric minimum-dilation graphs is NP-hard. *Internat. J. Comput. Geom. Appl.*, 20(2):147–173, 2010. Also at GD 2006. doi:10.1142/S0218195910003244.

12 Robert L. (Scot) Drysdale III, Scott A. McElfresh, and Jack Snoeyink. On exclusion regions for optimal triangulations. *Discret. Appl. Math.*, 109(1-2):49–65, 2001. doi:10.1016/S0166-218X(00)00236-5.

13 J. Mark Keil and Carl A. Gutwin. Classes of graphs which approximate the complete euclidean graph. *Discrete Comput. Geom.*, 7:13–28, 1992. doi:10.1007/BF02187821.

14 Christos Levcopoulos and Andrzej Lingas. Fast algorithms for greedy triangulation. *BIT*, 32(2):280–296, 1992. doi:10.1007/BF01994882.

15 Christos Levcopoulos, Giri Narasimhan, and Michiel H. M. Smid. Efficient algorithms for constructing fault-tolerant geometric spanners. In *Proc. 30th Annu. ACM Sympos. Theory Comput. (STOC)*, pages 186–195, 1998. doi:10.1145/276698.276734.

16 Christos Levcopoulos, Giri Narasimhan, and Michiel H. M. Smid. Improved algorithms for constructing fault-tolerant spanners. *Algorithmica*, 32(1):144–156, 2002.

17 Tamas Lukovszki. New results of fault tolerant geometric spanners. In Frank Dehne, Arvind Gupta, Jörg-Rüdiger Sack, and Roberto Tamassia, editors, *Proc. 6th Internat. Sympos. Algorithms and Data Struct.*, volume 1663 of *Lecture Notes in Computer Science*, pages 193–204. Springer-Verlag, 1999. doi:10.1007/3-540-48447-7_20.

18 Giri Narasimhan and Michiel H. M. Smid. *Geometric Spanner Networks*. Cambridge University Press, 2007. doi:10.1017/CBO9780511546884.

19 Merav Parter, Asaf Petruschk, Shay Sapir, and Elad Tzalik. Parks and recreation: Color fault-tolerant spanners made local. In *Annu. ACM-SIAM Sympos. Discrete Algorithms (SODA)*, pages 4061–4094, 2025. doi:10.1137/1.9781611978322.139.

Approximation Depth of Convex Polytopes

Egor Bakaev[1], Florestan Brunck[1], and Amir Yehudayoff[1,2]

1 **University of Copenhagen**
2 **Technion-IIT**

Abstract

We study approximations of polytopes in the standard model for computing polytopes using Minkowski sums and (convex hulls of) unions. Specifically, we study the ability to approximate a target polytope by polytopes of a given depth. Our main results imply that simplices can only be "trivially approximated". On the way, we obtain a characterization of simplices as the only "outer additive" convex bodies.

1 Introduction

There are numerous classical approximation problems in convex geometry, such as approximating a given convex set by polytopes and approximating a given polytope by specific polytopes. These problems, in their many forms, have been studied for many years with various types of motivation (see [4, 7, 8, 11] and references within).

Shephard and others, for example, considered the following approximation problem (see [14] and references within). The Minkowski sum of two sets K, L in Euclidean space is defined as $K + L = \{x + y : x \in K, y \in L\}$. A target convex set T is approximated by a collection $\mathcal{K}$ of convex sets if there is a sequence of sets (K_j) such that each K_j is a finite sum of elements of $\mathcal{K}$ so that $K_j \to T$ as $j \to \infty$ in the Hausdorff metric.

As we describe next, Shephard proved that an indecomposable T can only be "trivially approximated". A set $K \subset \mathbb{R}^n$ is indecomposable if it is "prime with respect to Minkowski summation" (see definition below). A homothet of $K \subset \mathbb{R}^n$ is a set of the form $H = x + \lambda K$ where $x \in \mathbb{R}^n$ and $\lambda \geq 0$ is called the coefficient of homothety. Denote by $\mathcal{H}_K$ the set of all homothets of K.

▶ **Definition 1.1.** A set $K \subset \mathbb{R}^n$ is indecomposable if for every $L_1, \ldots, L_m \subset \mathbb{R}^n$ such that $K = L_1 + \ldots + L_m$, it holds that $L_1, \ldots, L_m \in \mathcal{H}_K$.

Indecomposable polytopes were studied in many works (see e.g. [6, 14, 15]). It is known, e.g, that if K is a polytope so that all of its two-dimensional faces are triangles then K is indecomposable [15]. Indecomposability was also recently used to study the expressivity of monotone ReLU neural networks and input convex neural networks (ICNNs); see [3, 19]. Returning to the approximation problem mentioned above, Shephard proved that an indecomposable T can be approximated by the class $\mathcal{K}$ only if $\mathcal{K}$ contains a homothet of T.

1.1 Constructing polytopes

We study a non-asymptotic version of the approximation problem, which is motivated by the expressivity of neural networks (see [3, 9, 10] and references within). In other words, we care about approximating the target polytope up to a fixed error (and not error tending to zero).

We consider the following model for constructing polytopes. The two operations are: Minkowski sum and union. For two sets $L, K \subset \mathbb{R}^n$, let $K * L = \mathsf{conv}(K \cup L)$ where conv

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

denotes convex hull. Polytopes of depth zero are points: $\mathcal{P}_{n,0} = \{\{x\} : x \in \mathbb{R}^n\}$. Polytopes of depth $d > 0$ are defined inductively as

$$\mathcal{P}_{n,d} = \{(K_1 * L_1) + \ldots + (K_m * L_m) : m \in \mathbb{N}, K_j, L_j \in \mathcal{P}_{n,d-1}\}.$$

The standard name for polytopes in $\mathcal{P}_{n,1}$ is zonotopes.

We can now define the depth complexity of a polytope $P \subset \mathbb{R}^n$ as the minimum d so that $P \in \mathcal{P}_{n,d}$. The depth complexity of a polytope is always finite. In fact, if P has m vertices then its depth complexity is at most $\lceil \log_2 m \rceil$. This bound is sometimes not sharp; e.g., the depth complexity of cubes is one.

This model has a one-to-one correspondence with the following neural network model for computing convex functions. The input gates compute linear functions on $\mathbb{R}^n$. Inner gates compute functions of the form $g(x) = \sum_j c_j \max\{a_j(x), b_j(x)\}$ where for every j, it holds that $c_j > 0$ and a_j, b_j were previously computed. A network can thus be represented as a directed acyclic graph. The number of hidden layers in the network is the maximum number of inner gates in an input-output directed path.

The correspondence between polytopes and convex functions is seen via support functions. Every closed convex set $P \subset \mathbb{R}^n$ defines a support function $h_P : \mathbb{R}^n \to \mathbb{R}$ via

$$h_P(x) = \max\{\langle x, y\rangle : y \in P\}.$$

Support functions have many important properties (see e.g. [8]). In particular, there is a one-to-one correspondence between P and h_P. This correspondence translates to the computational setting: $P \in \mathcal{P}_{n,d}$ iff h_P is computed by a neural network with d hidden layers. There is a long sequence of related works on the "functional side" studying monotonicity in the context of neural networks; see [3, 5, 12, 16, 17] and references within.

We are interested in the depth complexity of *approximating* a given polytope. We measure the quality of an approximation using two measures of distance (defined below).[1]

A particularly important polytope in this setting is the simplex. A standard geometric representation of the n-dimensional simplex Δ^n is in $\mathbb{R}^n$ as the orthogonal projection of $\mathsf{conv}\{e_1, \ldots, e_{n+1}\}$ to the space orthogonal to $(1, 1, \ldots, 1) \in \mathbb{R}^{n+1}$, where $e_1, \ldots, e_{n+1} \in \mathbb{R}^{n+1}$ are the standard unit vectors. For concreteness, in this work Δ^n refers to this geometric representation of the simplex. Most of our results, however, hold for arbitrary simplices (which are affine images of the geometric simplex Δ^n).

The depth complexity of the simplex Δ^n is known to be $d = \lceil \log_2(n+1) \rceil$. The upper bound is trivial and the lower bound was proved by Valerdi [19]. The depth complexity of the simplex is particularly important because if the depth complexity of Δ^n is d then every piecewise linear function on $\mathbb{R}^{n-1}$ can be computed with d hidden layers [2, 20].

It is well known that simplices are indecomposable (e.g., [15]). Our main finding is the stronger statement that simplices can only be "trivially approximated". For example, we show that if for some depth parameter d, there is a sequence of polytopes (P_j) in $\mathcal{P}_{n,d}$ such that $P_j \to \Delta^n$ as $j \to \infty$ then $d \geq \lceil \log_2(n+1) \rceil$.

[1] As opposed to Shephard's result, in our non-asymptotic setting, some indecomposable polytopes can be approximated non-trivially. The reason is that, in dimension $n \geq 3$, the set of indecomposable polytopes is dense [13], so every polytope $P \subset \mathbb{R}^n$ is close to an indecomposable polytope $Q \subset \mathbb{R}^n$ (for example, we can build Q that is close to P such that all of its two-dimensional faces are triangles). If we start with a zonotope $P \in \mathcal{P}_{n,1}$, then we get an indecomposable Q that can be easily approximated.

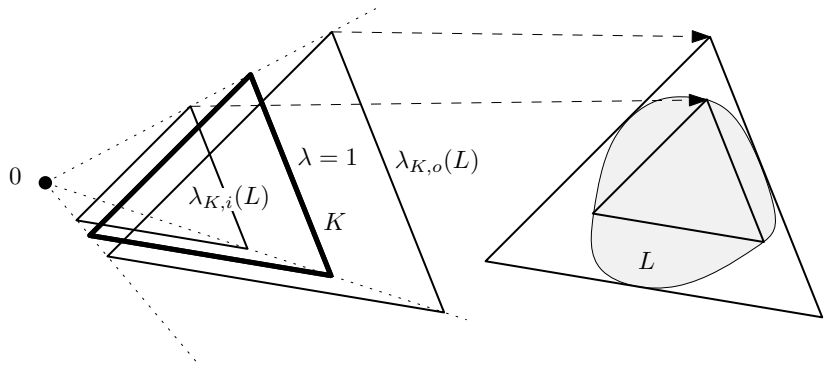


Figure 1 The in-out distance. The in-out distance with respect to K is defined as $D_{io}(L;K) = \lambda_{K,o}(L) - \lambda_{K,i}(L)$. We can think of $\lambda_{K,o}(L)$ as "the size of the smallest homothet of K containing L", and of $\lambda_{K,i}(L)$ as "the size of the largest homothet of K contained in L".

1.2 Distances

We measure the quality of approximations through the following two notions of distance. We start with the in-out distance; see illustration in Figure 1. For $H \in \mathcal{H}_K$, denote by $\lambda(H) = \lambda_K(H)$ the coefficient of homothety of H. Define outer and inner coefficients as follows. For $L, K \subset \mathbb{R}^n$, the outer coefficient of L with respect to K is[2]

$$\lambda_{K,o}(L) = \inf\{\lambda_K(H) : H \in \mathcal{H}_K, H \supseteq L\}.$$

The inner coefficient of L with respect to K is

$$\lambda_{K,i}(L) = \sup\{\lambda_K(H) : H \in \mathcal{H}_K, H \subseteq L\}.$$

A set $L \subset \mathbb{R}^n$ is called a convex body if it is convex, compact and has non-empty relative interior.[3] If $L \subseteq \mathbb{R}^n$ is a convex body then $0 < \lambda_{\Delta,o}(L) < \infty$.

▶ **Definition 1.2.** The in-out distance is defined by[4] $D_{io}(L;K) = \lambda_{K,o}(L) - \lambda_{K,i}(L)$.

It is somewhat similar to the Banach–Mazur distance, but it is not invariant under scaling or invertible linear maps. It is, in fact, positively homogeneous: for all $\lambda > 0$, $D_{io}(\lambda L;K) = \lambda D_{io}(L;K)$. Consequently, the in-out distance depends on the scale of L. The smaller L is, the smaller $D_{io}(L;K)$ is.

The second notion of distance, the empty corners distance, is defined only for the simplex. Consider the simplex $\Delta = \Delta^n$. For $H \in \mathcal{H}_\Delta$, the vertices of Δ are in one-to-one correspondence to the vertices of H, so we can identify them and denote them by $V(\Delta) = V(H)$. For a set $L \subset \mathbb{R}^n$, for $H \in \mathcal{H}_\Delta$ so that $L \subseteq H$ and for $v \in V(\Delta)$, the size of the empty corner of L with respect to v is (see Figure 2)

$$E_v(L;H) = \sup\left\{\lambda_\Delta(F) : F \in \mathcal{H}_\Delta, v \in V(F), F \subseteq H, F \cap L = \emptyset\right\};$$

it is zero when the set of F's is empty. In the plane, for example, L is some planar convex body, H is a triangle containing L, v is a vertex of H, and F is the largest similar triangle

[2] It is ∞ when the set of H's is empty.
[3] In some works, the term "body" means non-empty interior.
[4] When the expression is $\infty - \infty$, it is not defined.

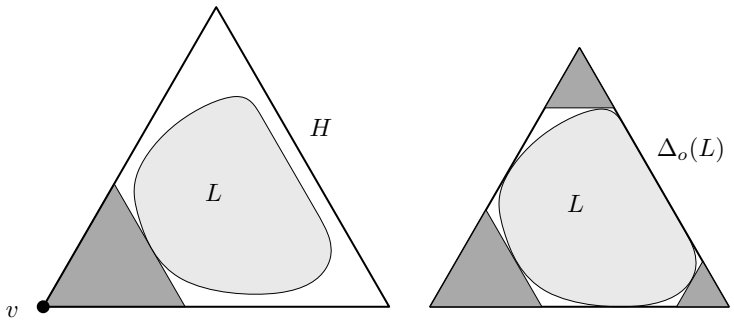


Figure 2 The empty corner distance. *On the left:* The size of the empty corner of L with respect to the vertex v is denoted by $E_v(L;H)$ where H is some homothet of Δ containing L. *On the right:* The empty corner distance of a convex body L is denoted by $D_e(L)$. It is the normalized sum of the sizes of all empty corners of the smallest homothet of Δ containing L.

touching v and (essentially) disjoint from L. The total empty corner size is

$$E(L;H) = \sum_{v \in V(\Delta)} E_v(L;H).$$

For a bounded $L \subset \mathbb{R}^n$, let $\Delta_o(L)$ denote the unique $H \in \mathcal{H}_{\Delta^n}$ so that $H \supseteq L$ and $\lambda_\Delta(H) = \lambda_{\Delta,o}(L)$. In other words, it is the smallest homothet of Δ containing L. Uniqueness holds because if $H, H' \in \mathcal{H}_\Delta$ are such that $H \cap H' \neq \emptyset$ then $H \cap H' \in \mathcal{H}_\Delta$.

▶ **Definition 1.3.** The empty corner distance of a convex body $L \subset \mathbb{R}^n$ is

$$D_e(L) = D_e(L;\Delta^n) = \frac{E(L;\Delta_o(L))}{\lambda_{\Delta,o}(L)}.$$

The empty corner distance is invariant to scale: for all $\lambda > 0$, $D_e(L) = D_e(\lambda L)$. It trivially holds that $0 \leq D_e(L) \leq n+1$. It is sometimes convenient to work with the co-distance

$$co\text{-}D_e(L) = co\text{-}D_e(L;\Delta^n) := n + 1 - D_e(L).$$

The following simple observation partially justifies the term "distance".

▶ Observation 1.4. For every convex body $L \subset \mathbb{R}^n$,

$$L \in \mathcal{H}_\Delta \iff D_{io}(L;\Delta^n) = 0 \iff D_e(L;\Delta^n) = 0.$$

1.3 Results

We start with a couple of theorems showing that simplices are stably indecomposable. The theorems show that the Minkowski sum operation cannot decrease the distance to a simplex. The theorems are stronger than Shephard's inapproximability result mentioned above for the special case of simplices.

▶ **Theorem 1.5.** *If $L = L_1 + \ldots + L_m$ where $L_1, \ldots, L_m \subset \mathbb{R}^n$ are convex bodies then*

$$D_{io}(L;\Delta^n) \geq \max\{D_{io}(L_j;\Delta^n) : j \in [m]\}.$$

▶ **Theorem 1.6.** *If $L = L_1 + \ldots + L_m$ where $L_1, \ldots, L_m \subset \mathbb{R}^n$ are convex bodies then*

$$co\text{-}D_e(L) \leq \max\{co\text{-}D_e(L_j) : j \in [m]\}.$$

The following theorem describes the effect of the union operation on the empty corner distance.

▶ **Theorem 1.7.** *If $L = L_1 * L_2$ where $L_1, L_2 \subset \mathbb{R}^n$ are convex bodies then*

$$co\text{-}D_e(L) \leq co\text{-}D_e(L_1) + co\text{-}D_e(L_2).$$

Next, we calculate the distance of the simplex from centrally symmetric sets. A set $L \subseteq \mathbb{R}^n$ is centrally symmetric if there is $p \in \mathbb{R}^n$ so that for all $x \in \mathbb{R}^n$ we have $p + x \in L$ iff $p - x \in L$. The point p is called the center of symmetry of L. It is worth mentioning that some works measured how far is a given convex body from being centrally symmetric using inscribed simplices (see e.g. [18] and references within). Surprisingly, the distance is the same for all centrally symmetric sets.

▶ **Lemma 1.8.** *If $L \subset \mathbb{R}^n$ is a centrally symmetric convex body then $D_e(L) = n - 1$.*

We can now draw the following conclusion that says that simplices can only be trivially approximated. The next theorem fully describes the best approximation of the simplex by depth-d polytopes. For a depth parameter $d > 0$, let[5]

$$D_e(\mathcal{P}_{n,d}) := \min\{D_e(P; \Delta^n) : P \in \mathcal{P}_{n,d} \setminus \mathcal{P}_{n,0}\}.$$

▶ **Theorem 1.9.** *For $0 < d < \lceil \log_2(n+1) \rceil$, we have $D_e(\mathcal{P}_{n,d}) = n + 1 - 2^d$.*

Theorem 1.9 is stronger than the $\lceil \log_2(n+1) \rceil$ lower bound on the depth complexity of Δ^n proved in [19]. It says that we cannot even approximate Δ^n in depth smaller than $\lceil \log_2(n+1) \rceil$. For example, if the polytope P has depth complexity $d < \lceil \log_2(n+1) \rceil$ then $D_e(P) \geq 1$.

The previous three theorems concern the empty corner distance. To complete the picture, the following lemma shows that (with the proper normalization) the empty corner distance bounds from below the in-out distance. So, the inapproximability results also carry on to the in-out distance. We also provide a bound in the other direction.

▶ **Lemma 1.10.** *If $L \subset \mathbb{R}^n$ is a convex body such that $\lambda_{\Delta,o}(L) = 1$ then*

$$D_e(L; \Delta^n) \leq n \cdot D_{io}(L; \Delta^n).$$

▶ **Lemma 1.11.** *If $L \subset \mathbb{R}^n$ is a convex body such that $\lambda_{\Delta,o}(L) = 1$ then*

$$D_e(L; \Delta^n) \geq D_{io}(L; \Delta^n).$$

Both lemmas are sharp (see remarks after their proofs). Somewhat surprisingly, the proof of Lemma 1.11 relies on a Kadets-type result proved by Akopyan and Karasev [1].

1.4 A characterization of simplices

The first three theorems stated above are based on the fact that the simplex satisfies the following additivity property.

▶ **Definition 1.12.** A convex body $K \subset \mathbb{R}^n$ is called outer additive if for every $L = L_1 + L_2$ where $L_1, L_2 \subset \mathbb{R}^n$ are convex bodies, it holds that $\lambda_{K,o}(L) = \lambda_{K,o}(L_1) + \lambda_{K,o}(L_2)$.

[5] $P \notin \mathcal{P}_{n,0}$ because points are not convex bodies. As the theorem shows, the minimum is attained.

The additivity of K does not rely on the "position" of K; it is invariant under invertible affine maps. Because it is always true that $\lambda_{K,o}(L) \leq \lambda_{K,o}(L_1) + \lambda_{K,o}(L_2)$, the additivity of K is equivalent to $\lambda_{K,o}(L) \geq \lambda_{K,o}(L_1) + \lambda_{K,o}(L_2)$.

The fact that simplices are outer additive easily follows from the additivity of support functions. Are simplices the only outer additive convex bodies?

▶ **Theorem 1.13.** *A convex body is outer additive iff it is a simplex.*

References

1 Arseniy Akopyan and Roman Karasev. Kadets-type theorems for partitions of a convex body. *Discrete & Computational Geometry*, 48:766–776, 2012.

2 Raman Arora, Amitabh Basu, Poorya Mianjy, and Anirbit Mukherjee. Understanding deep neural networks with rectified linear units. In *International Conference on Learning Representations*, 2018.

3 Egor Bakaev, Florestan Brunck, Christoph Hertrich, Daniel Reichman, and Amir Yehudayoff. On the depth of monotone ReLU neural networks and ICNNs. *arXiv preprint arXiv:2505.06169*, 2025.

4 Károly Böröczky Jr. Approximation of general smooth convex bodies. *Advances in Mathematics*, 153(2):325–341, 2000.

5 Hennie Daniels and Marina Velikova. Monotone and partially monotone neural networks. *IEEE Transactions on Neural Networks*, 21(6):906–917, 2010.

6 William J Firey and Branko Grünbaum. Addition and decomposition of convex polytopes. *Israel Journal of Mathematics*, 2(2):91–100, 1964.

7 Peter M Gruber. Approximation of convex bodies. In *Convexity and its Applications*, pages 131–162. Springer, 1983.

8 Branko Grünbaum. *Convex Polytopes*. Springer-Verlag, New York, 2003.

9 Christian Haase, Christoph Hertrich, and Georg Loho. Lower bounds on the depth of integral ReLU neural networks via lattice polytopes. *arXiv:2302.12553*, 2023.

10 Christoph Hertrich, Amitabh Basu, Marco Di Summa, and Martin Skutella. Towards lower bounds on the depth of ReLU neural networks. *Advances in Neural Information Processing Systems*, 34:3336–3348, 2021.

11 Michael Kallay. Indecomposable polytopes. *Israel Journal of Mathematics*, 41:235–243, 1982.

12 Dan Mikulincer and Daniel Reichman. Size and depth of monotone neural networks: Interpolation and approximation. *IEEE Transactions on Neural Networks and Learning Systems*, 2024.

13 Rolf Schneider. *Convex bodies: the Brunn–Minkowski theory*, volume 151. Cambridge university press, 2013.

14 G. C. Shephard. Approximation problems for convex polyhedra. *Mathematika*, 11(1):9–18, 1964. doi:10.1112/S0025579300003430.

15 Geoffrey C. Shephard. Decomposable convex polyhedra. *Mathematika*, 10(2):89–95, 1963.

16 Joseph Sill. Monotonic networks. *Advances in neural information processing systems*, 10, 1997.

17 Aishwarya Sivaraman, Golnoosh Farnadi, Todd Millstein, and Guy Van den Broeck. Counterexample-guided learning of monotonic neural networks. *Advances in Neural Information Processing Systems*, 33:11936–11948, 2020.

18 Gabor Toth. Asymmetry of convex sets with isolated extreme points. *Proceedings of the American Mathematical Society*, 137(1):287–295, 2009.

19 Juan L. Valerdi. On minimal depth in neural networks. *arXiv:2402.15315*, 2024.

20 Shuning Wang and Xusheng Sun. Generalization of hinging hyperplanes. *IEEE Transactions on Information Theory*, 51(12):4425–4431, 2005.

Range Counting Oracles for Extent Problems*

Sanghwa Han[1] and Eunjin Oh[2]

1 **Department of Computer Science and Engineering, POSTECH**
H.Sanghwa@postech.ac.kr
2 **Department of Computer Science and Engineering, POSTECH**
eunjin.oh@postech.ac.kr

Abstract

We are given access to a set of points in the plane through an oracle that returns the number of points contained in a given query rectangle. Our goal is to design algorithms that solve geometric extent problems using a small number of oracle queries. We present near-optimal algorithms for several fundamental problems, including the computation of the width, diameter, farthest neighbor, and minimum enclosing ball.

1 Introduction

Recent applications in domains such as spatial databases generate data at scales where scanning the entire input is prohibitively expensive, motivating the development of sublinear-time approximation algorithms. We focus specifically on the range counting oracle model, where algorithms interact with the data solely through a black-box oracle that returns the number of points within an axis-aligned query rectangle, introduced by [14, 10]. This model aligns well with modern data-management pipelines, as counting queries are efficiently supported by spatial indices (e.g., R-trees, quadtrees) and facilitate privacy-preserving analysis by exposing only aggregate statistics instead of sensitive point coordinates.

In this paper, we present algorithms for computing geometric extent measures, such as the width, diameter, farthest neighbor (FN) and minimum enclosing ball (MEB), in the range oracle model. These measures are fundamental primitives for quantifying the global shape and spread of data [2, 8]. These primitives serve as essential in computational geometry and arise in wide range of applications, including collision detection, shape analysis, motion planning, and data clustering [5, 11, 12, 15]. For instance, the width plays a role in measuring the flatness of data, while the diameter and MEB provide bounds on the spatial scale of the input. Since exact computation of these measures can be costly on massive datasets, developing efficient approximation algorithms is a central challenge in the field.

Our results We first propose an algorithm that constructs an *s-approximate convex hull*, which will be defined later, using only $O(s)$ range counting queries for any parameter s ($1 \leq s \leq n$), where n denotes the number of input points. Using this structure, we derive approximations for several extent measures. Our main results are summarized as follows:

Related work Approximating extent measures has been extensively studied in sublinear setting, particularly under the streaming and coreset models [3, 4, 1, 13]. While effective,

* Supported by Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No.RS-2024-00440239, Sublinear Scalable Algorithms for Large-Scale Data Analysis) and the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No.RS-2024-00358505).

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

Table 1 Summary of our results. Here, Δ denotes the size of the domain, n denotes the number of points. Then s is a trade-off parameter between the additive error and query complexity.

Problem	Approximation guarantee	Query complexity	
		Upper bound	Lower bound
Width	additive $O(\Delta/s)$	$O(s)$	$\Omega(s)$
Diameter	multiplicative $(1 \pm \epsilon)$	$O(\log \Delta + 1/\epsilon)$	$\Omega(\log \Delta + 1/\epsilon)$
FN	multiplicative $(1 \pm \epsilon)$	$O(\log \Delta + 1/\epsilon)$	$\Omega(\log \Delta + 1/\epsilon)$
MEB	multiplicative $(1 \pm \epsilon)$	$O(\log \Delta + 1/\epsilon)$	$\Omega(\log \Delta + 1/\epsilon)$

these approaches typically rely on direct access to point coordinates or the ability to sample points uniformly. Our setting shares conceptual similarities with geometric probing [6, 9], where the goal is to reconstruct a geometric shape using intersection queries (probes) without viewing the object. Our work also relates to geometric property testing, where the objective is to determine if a point set satisfies a certain property (e.g., convexity) using sublinear queries [16, 7]. Recently the range counting oracle model has gained attention for solving geometric optimization problems without accessing individual coordinates [14, 10].

Preliminaries We consider a set P of n points in the discrete space $[\Delta]^2$, where Δ denotes the size of the domain. We cannot access P directly; Instead, we access P through the range counting oracle model that returns the number of points of P contained in any arbitrary query axis-aligned rectangle. In this model, the efficiency of an algorithm is measure by the number of queries used by the algorithm.

To develop sublinear algorithms, we impose a grid over the domain $[\Delta]^2$, determined by a given resolution parameter s ($1 \leq s \leq n$). We partition the domain into s vertical columns $C_1, \ldots, C_s$ and s horizontal rows $R_1, \ldots, R_s$. The intersection of column C_i and row R_j forms a unique grid cell $c_{i,j}$, which is a square of side length $w = \Delta/s$. For any non-empty cell $c_{i,j}$, we use its center for its representative point, denoted as $p_{\mathrm{rep}}(c_{i,j})$. The maximum Euclidean distance between any point $p \in P \cap c_{i,j}$ and its representative is bounded by:

$$\|p - p_{\mathrm{rep}}(c_{i,j})\| \leq \frac{\sqrt{2}}{2} w = O(\Delta/s).$$

Finally, we formally define the notion of an approximate convex hull, which is central to our results. Let $d_H(A, B)$ denote the Hausdorff distance between two convex sets A and B. Formally,

$$d_H(A, B) = \max \left\{ \max_{a \in A} \min_{b \in B} \|a - b\|, \ \max_{b \in B} \min_{a \in A} \|a - b\| \right\}.$$

▶ **Definition 1.1** (s-approximate convex hull)**.** Let P be a set of points in the plane contained in the domain of side length D and $s \geq 1$ be a resolution parameter. An s-approximate convex hull of P is a convex set $\widetilde{\mathsf{CH}}(P)$ such that the Hausdorff distance between the true convex hull $CH(P)$ and $\widetilde{\mathsf{CH}}(P)$ satisfies:

$$d_H(CH(P), \widetilde{\mathsf{CH}}(P) \leq f(D, s)$$

where $f(D, s)$ is an error bound. In this paper, our construction achieves $f(D, s) = O(D/s)$.

Due to the space limit, we omit some proofs and details. All missing proofs and details can be found in the appendices.

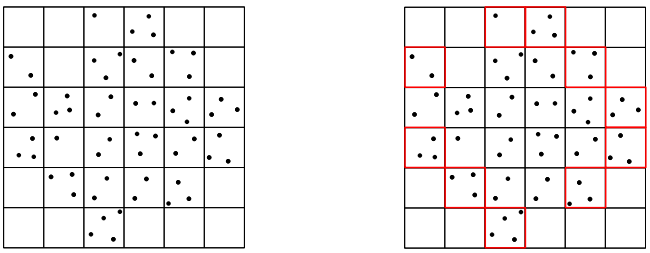
2 Common Algorithm: Computing an Approximate Convex Hull

Our goal in this section is to construct an s-approximate convex hull $\widetilde{\mathsf{CH}}(P)$ using only a sublinear number of queries. We observe that such an approximation can be constructed from the set of representative points of the extreme cells. In Section 2.1, we define extreme cells and show that the convex hull of the representative points of extreme cells constitutes a $\widetilde{\mathsf{CH}}(P)$. In Section 2.2, we describe how to compute the extreme cells efficiently in the oracle model and analyze the query complexity.

2.1 Definition and Properties of Extreme Cells

We formalize the definition of the extreme cells that will be used to compute $\widetilde{\mathsf{CH}}(P)$. A grid cell is non-empty if it contains a point of P, and empty otherwise. We call a non-empty grid cell $c_{i,j}$ an *extreme cell* if there exists one of the four quadrants determined by $c_{i,j}$ that contains no non-empty cell other than $c_{i,j}$ itself. Specifically, the four quadrants determined by $c_{i,j}$ are defined as follows:

- the top-left quadrant consists of cells $c_{i',j'}$ with $i' \leq i$ and $j' \geq j$;
- the top-right quadrant consists of cells $c_{i',j'}$ with $i' \geq i$ and $j' \geq j$;
- the bottom-left quadrant consists of cells $c_{i',j'}$ with $i' \leq i$ and $j' \leq j$; and
- the bottom-right quadrant consists of cells $c_{i',j'}$ with $i' \geq i$ and $j' \leq j$.



A B

Figure 1 In the sub-figure A, a point set $P \subseteq [\Delta]^2$ is given. In the sub-figure B, an extreme cell c (red square) is shown.

We let $\mathcal{C}_{\text{ext}}$ be the set of all non-empty cells $c_{i,j}$ such that, for at least one of these quadrants, there is no other non-empty cell $c_{i',j'} \neq c_{i,j}$ inside that quadrant. See Figure 1. An extreme cell is a maximal element under the dominance order induced by one of the four quadrants. This definition identifies the outer envelope of the occupied cells on the grid; in particular, the representative points of $\mathcal{C}_{\text{ext}}$ will be sufficient to construct an s-approximate convex hull. Let P_{ext} denote the set of representative points of the cells in $\mathcal{C}_{\text{ext}}$.

▶ **Lemma 2.1.** *The convex hull of P_{ext} is an s-approximate convex hull of P.*

2.2 Computing All Extreme Cells

We now present the main algorithm that computes the set of extreme cells $\mathcal{C}_{\text{ext}}$ using only $O(s)$ queries. Recall from Section 2.1 that the convex hull of the representative points of $\mathcal{C}_{\text{ext}}$ constitutes an s-approximate convex hull, denoted as $\widetilde{\mathsf{CH}}(P)$.

In the following, we show how to compute the top-left part of the boundary of $\widetilde{\mathsf{CH}}(P)$ only. The other parts can be computed analogously. We process the columns from left to

right in order to identify the extreme cells forming the upper chain of $\widetilde{\mathsf{CH}}(P)$. We begin by finding the topmost non-empty cell c in the leftmost non-empty column, which can be done using $O(\log s)$ range counting queries by binary search. The cell c is an extreme cell and its representative is the leftmost point of $\widetilde{\mathsf{CH}}(P)$. We maintain the horizontal line ℓ containing the top edge of the recently found extreme cell. We initialize ℓ with respect to c. We then consider the remaining columns one by one.

Let C' denote the current column. Using a single range counting query, we check whether C' contains any point of P lying strictly above ℓ. If not, then C' contains no extreme cell, and we proceed to the next column. Otherwise, C' contains at least one extreme cell. In this case, we search within C' starting from the cell whose bottom edge lies on ℓ and moving upward. For each cell c' encountered during this search, we test whether there exists any point of P in C' lying strictly above c'. The first cell for which the answer is negative is an extreme cell. Once such a cell c' is found, we update ℓ to be the horizontal line containing the top edge of c', and continue with the next column.

▶ **Lemma 2.2.** *We can find all extreme cells using $O(s)$ range counting queries.*

In conclusion, the proposed algorithm efficiently computes an s-approximate convex hull $\widetilde{\mathsf{CH}}(P)$. Based on this construction, we demonstrate in the subsequent sections that $\widetilde{\mathsf{CH}}(P)$ suffices to approximate geometric extent measures within the desired error bound.

3 Width Problem

In this section, we investigate the problem of estimating the width of a point set $P \subseteq [\Delta]^2$. The width of P, denoted as $\mathsf{width}(P)$, is defined as the minimum distance between two parallel supporting lines of the convex hull of P. We can obtain an $O(\Delta/s)$-additive approximation of $\mathsf{width}(P)$ with $O(s)$ queries and show that the query complexity of $O(s)$ is tight.

3.1 Sublinear Algorithm for Width Problem

The width depends only on the convex hull of P: interior points do not affect the supporting lines that define $\mathsf{width}(P)$. Therefore, it suffices to work with $\widetilde{\mathsf{CH}}(P)$. In this section, we show that the Hausdorff closeness between $\mathsf{CH}(P)$ and $\widetilde{\mathsf{CH}}(P)$ translates into an $O(\Delta/s)$ additive error in width. This implies the following: Given a set P of n points in $[\Delta]^2$ and a parameter $s \geq 1$, we can estimate the width of P with an additive error of $O(\Delta/s)$ using $O(s)$ range counting queries.

The estimation procedure is straightforward. First, we construct $\widetilde{\mathsf{CH}}(P)$ using the algorithm from Section 2. We then compute the width of the $\widetilde{\mathsf{CH}}(P)$ using the rotating calipers algorithm [17], which requires no additional oracle queries because the coordinates of P_{ext} are explicitly known. Consequently, as the query cost is incurred only in the construction of $\widetilde{\mathsf{CH}}(P)$, the overall query complexity is $O(s)$.

▶ **Lemma 3.1.** *Let A and B be two convex sets in $\mathbb{R}^2$. If their Hausdorff distance satisfies $d_H(A, B) \leq \delta$, then:*

$$|\mathit{width}(A) - \mathit{width}(B)| \leq 2\delta.$$

By applying Lemma 3.1 with $A = \mathsf{CH}(P)$ and $B = \widetilde{\mathsf{CH}}(P)$, we conclude that our algorithm has an additive error of $O(\Delta/s)$.

3.2 Lower Bound for Width Problem

In this section, we show that the query complexity of $O(s)$ achieved by our algorithm is tight. That is, for any parameter s, any randomized algorithm that estimates the width of a point set P in $[\Delta]^2$ within an additive error of $O(\Delta/s)$ requires $\Omega(s)$ range counting queries.

To establish a more general impossibility result, we extend our analysis beyond deterministic approaches to encompass randomized algorithms. We demonstrate that the $\Omega(s)$ query requirement is an inherent limit of the model, holding even when algorithms are permitted to make probabilistic choices. To formalize this randomized lower bound, we apply Yao's Minimax Principle [18]. This principle implies that if we can exhibit a probability distribution $\mathcal{D}$ over input instances such that any deterministic algorithm requires $\Omega(s)$ queries to succeed with probability at least $2/3$, then any randomized algorithm also requires $\Omega(s)$ queries to achieve success probability at least $2/3$. Therefore, to establish the randomized lower bound, it suffices to demonstrate a distribution $\mathcal{D}$ where no deterministic algorithm using at most s queries can achieve an additive error within Δ/s with a success probability of at least $2/3$.

Grid and Gadget Construction We partition the domain $[\Delta]^2$ into a grid of $8s \times 8s$ cells and focus specifically on the sequence of disjoint cells $\mathcal{C} = \{c_1, \ldots, c_m\}$ located along the main diagonal $y = x$, where $m = \Theta(s)$. We define two types of gadgets within a cell c_i. A near gadget contains points aligned strictly along the line $y = x$; its local width perpendicular to the main diagonal is 0. A far gadget contains points aligned along $y = -x$; it creates a geometric extent of $\Theta(\Delta/s)$ perpendicular to the main diagonal. See Figure 2 as follows.

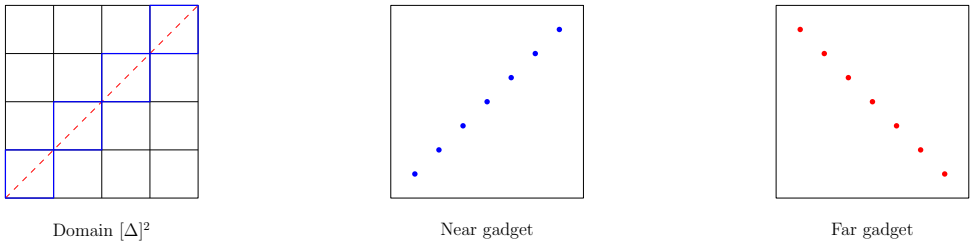


Figure 2 We partition $[\Delta]^2$ into $8s \times 8s$ cells. On the sequence of disjoint cells located along the $y = x$, we place either the near gadget or far gadget.

Input Distribution $\mathcal{D}$ We construct the distribution $\mathcal{D}$ as follows. With probability $1/2$, every cell $c_i \in \mathcal{C}$ contains a near gadget. In this case, all points in the entire domain lie on the line $y = x$, resulting in a global width of 0. With probability $1/2$, we select one index $t \in \{1, \ldots, m\}$ uniformly at random. The cell c_t contains the far gadget, while all other cells contain the near gadget. In this instance, a single far gadget creates a difference, resulting in a width of $\Theta(\Delta/s)$. To distinguish between a width of 0 and $\Theta(\Delta/s)$, any deterministic algorithm must determine whether the input contains a far gadget.

Analyze the probability Let $\mathcal{A}$ be a deterministic algorithm that satisfies the condition and consider $\mathcal{A}$ as a decision tree of depth s, where each node corresponds to a range query. The number of points in a cell is invariant regardless of the type of gadget; a query Q can distinguish two instances only if the corners of Q are contained in the interior of c_t. Since the cells in $\mathcal{C}$ are arranged along the main diagonal and Q is an axis-aligned rectangle, the corner of Q can intersect the interior of at most two cells in $\mathcal{C}$. Because the target index t is chosen

uniformly at random from $m = \Theta(s)$ candidates, the probability that any single query hits c_t is bounded by $O(1/s)$. If we use s queries, the probability that any of the s queries detects the far gadget is strictly less than a constant $(1 - (1 - \frac{2}{8s})^s = 1 - (1 - \frac{1}{4s})^s < \frac{1}{3})$. By the law of total probability, $\Pr[success] \leq \frac{1}{2} \cdot 1 + \frac{1}{2} \cdot \Pr[detect far] < \frac{1}{2} + \frac{1}{2} \cdot \frac{1}{3} = \frac{2}{3}$. Consequently, the algorithm fails to distinguish the input from the all-near instance (width 0) with sufficient probability, proving that $\Omega(s)$ queries are necessary to achieve a success probability of at least 2/3.

4 Approximating Other Extents Measures

Due to space constraints, we defer the full details of the algorithms and lower bounds for the diameter, farthest neighbor (FN), and minimum enclosing ball (MEB) problems to the appendices.

Here, we briefly sketch the algorithm used to achieve a $(1 + \epsilon)$-approximation for these measures using $O(\log \Delta + 1/\epsilon)$ queries. Unlike the width problem, these measures admit a multiplicative approximation guarantee. This is because the true values of the diameter, FN, and MEB are strictly bounded by a constant factor of the point set's bounding box size.

Taking advantage of this geometric property, our approach proceeds in these steps: We first compute the axis-aligned bounding box $\mathcal{B}$ of P using $O(\log \Delta)$ range counting queries. To compute $\mathcal{B}$, we perform binary searches to determine the minimum and maximum coordinates of P along each axis, which requires $O(\log \Delta)$ queries in total. Let L be its maximum side length. This step establishes the geometric scale, as L serves as a constant-factor lower bound for the exact measures. We then restrict our domain to $\mathcal{B}$ and construct the approximate convex hull $\widetilde{\mathsf{CH}}(P)$ using the algorithm of Section 2 with a resolution parameter $s = \Theta(1/\epsilon)$. This takes $O(1/\epsilon)$ queries and introduces an additive error of $O(L/s) = O(\epsilon L)$. Finally, we compute the extent measure on the $O(s)$ vertices of $\widetilde{\mathsf{CH}}(P)$ using known algorithms without requiring additional queries. Since the additive error is $O(\epsilon L)$, this directly translates into a $(1 + \epsilon)$-approximation for the point set P.

References

1 Pankaj K Agarwal, Sariel Har-Peled, et al. Geometric approximation via coresets. *Combinatorial and computational geometry*, 52(1):1–30, 2005.

2 Pankaj K Agarwal, Sariel Har-Peled, and Kasturi R Varadarajan. Approximating extent measures of points. *Journal of the ACM (JACM)*, 51(4):606–635, 2004.

3 Pankaj K Agarwal and R Sharathkumar. Streaming algorithms for extent problems in high dimensions. *Algorithmica*, 72(1):83–98, 2015.

4 Alexandr Andoni and Huy L Nguyen. Width of points in the streaming model. In *Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 447–452. SIAM, 2012.

5 Gill Barequet and Sariel Har-Peled. Efficiently approximating the minimum-volume bounding box of a point set in three dimensions. *Journal of Algorithms*, 38(1):91–109, 2001.

6 Richard Cole and Chee K Yap. Shape from probing. *Journal of Algorithms*, 8(1):19–38, 1987.

7 Artur Czumaj and Christian Sohler. Property testing with geometric queries. In *European Symposium on Algorithms*, pages 266–277. Springer, 2001.

8 Pooya Davoodi, Michiel Smid, and Freek Van Walderveen. Two-dimensional range diameter queries. In *Latin American Symposium on Theoretical Informatics*, pages 219–230. Springer, 2012.

9 David Dobkin, Herbert Edelsbrunner, and Chee K Yap. Probing convex polytopes. In *Proceedings of the Eighteenth Annual ACM Symposium on Theory of Computing*, pages 424–432. SIAM, 1986.

10 Anne Driemel, Morteza Monemizadeh, Eunjin Oh, Frank Staals, and David P Woodruff. Range counting oracles for geometric problems. In *41st International Symposium on Computational Geometry (SoCG 2025)*, pages 42–1. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2025.

11 Christos Faloutsos and King-Ip Lin. Fastmap: A fast algorithm for indexing, data-mining and visualization of traditional and multimedia datasets. In *Proceedings of the 1995 ACM SIGMOD international conference on Management of data*, pages 163–174, 1995.

12 James T Klosowski, Martin Held, Joseph SB Mitchell, Henry Sowizral, and Karel Zikan. Efficient collision detection using bounding volume hierarchies of k-dops. *IEEE transactions on Visualization and Computer Graphics*, 4(1):21–36, 1998.

13 Piyush Kumar, Joseph SB Mitchell, and E Alper Yildirim. Approximate minimum enclosing balls in high dimensions using core-sets. *Journal of Experimental Algorithmics (JEA)*, 8(1):1–1, 2003.

14 Morteza Monemizadeh. Facility location in the sublinear geometric model. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2023)*, pages 6–1. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2023.

15 Jia Pan, Christian Lauterbach, and Dinesh Manocha. g-planner: Real-time motion planning and global navigation using gpus. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 1245–1251, 2010.

16 Luis Rademacher and Santosh Vempala. Testing geometric convexity. In *International Conference on Foundations of Software Technology and Theoretical Computer Science*, pages 469–480. Springer, 2004.

17 Godfried T Toussaint. Solving geometric problems with the rotating calipers. In *Proc. IEEE Melecon*, page A10. IEEE, 1983.

18 Andrew Chi-Chin Yao. Probabilistic computations: Toward a unified measure of complexity. In *18th Annual Symposium on Foundations of Computer Science (sfcs 1977)*, pages 222–227. IEEE Computer Society, 1977.

How many times can two minimum spanning trees cross?

Todor Antić[1], Morteza Saghafian[2], Maria Saumell[3], Felix Schröder[1], Josef Tkadlec[4], and Pavel Valtr[1]

1 **Department of Applied Mathematics, Faculty of Mathematics and Physics, Charles University, Czech Republic**
{todor,schroder,valtr}@kam.mff.cuni.cz
2 **Institute of Science and Technology Austria, Klosterneuburg, Austria**
morteza.saghafian@ist.ac.at
3 **Department of Theoretical Computer Science, Faculty of Information Technology, Czech Technical University in Prague, Czech Republic**
maria.saumell@fit.cvut.cz
4 **Computer Science Institute, Faculty of Mathematics and Physics, Charles University, Czech Republic**
josef.tkadlec@iuuk.mff.cuni.cz

Abstract

Let P be a generic set of n points in the plane, and let $P = R \cup B$ be a coloring of P in two colors. We are interested in the number of crossings[1] between the minimum spanning trees (MSTs) of R and B, denoted by $\mathrm{cr}(R, B)$. We define the *bicolored MST crossing number* of P, denoted by cr-MST(P), as the maximum value of $\mathrm{cr}(R, B)$ taken over all colorings (or partitions) $P = R \cup B$, i.e., cr-MST$(P) = \max_{P=R\cup B}(\mathrm{cr}(R, B))$. We prove a linear upper bound for cr-MST(P) when P is generic. If P is dense or in convex position, we provide linear lower bounds.

Related Version arXiv:2601.20060

1 Introduction

When two sets of points are placed in the plane, how can we tell whether they are well separated or thoroughly mixed? This intuitive question arises in a variety of contexts, from clustering and spatial statistics to ecology and spatial biology; see [10, 13, 14]. One natural approach is to compare the structures that arise when each subset is considered independently, for instance, by constructing a spanning tree for each set and examining how those trees interact. One notable robust example that has recently attracted attention is the MST-ratio, defined as the sum of the lengths of the minimum spanning trees of the individual point sets divided by the length of the minimum spanning tree of their union. Intuitively, a higher MST-ratio indicates a greater degree of spatial mixing between the two point sets, see [2, 5, 7].

Among such approaches, a particularly simple measure, suggested in [2], considers the number of edge crossings between the minimum spanning trees (MSTs) of disjoint point sets. Intuitively, when the sets are well separated, their MSTs tend to avoid each other; conversely, the more the MSTs cross, the more spatially mixed the point sets are. This makes MST crossings a natural geometric proxy for mixedness and motivates us to study this measure. The crossings of minimum spanning trees have also been studied in [11], where it is proved that two MSTs cross at most $8n$ times where n is the total number of points.[2]

[1] We say that two edges create a *crossing* if the interiors of the corresponding segments intersect.
[2] However, the proof of one of the key lemmas is missing from the paper. See Section 1.1 for details.

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

We say that a planar point set is in *general position* if no three of its points lie on a line. We say that it is in *generic position with respect to minimum spanning trees*, or simply *generic*, if it is in general position and any subset of it has a unique minimum spanning tree. For instance, any set in general position with pairwise distinct distances is generic.

Let R, B be two disjoint point sets in the plane such that $R \cup B$ is generic. We denote by $\mathrm{cr}(R, B)$ the number of crossings between the minimum spanning trees of R and B, where two edges create a *crossing* if the interiors of the corresponding segments intersect. For a set P of points in the plane in generic position, we denote by $\text{cr-MST}(P)$ the maximum value of $\mathrm{cr}(R, B)$, taken over all colorings (or partitions) $P = R \cup B$, and we call this parameter the *bicolored MST crossing number* of P.

1.1 Our Results

We establish the following upper and lower bounds on the bicolored MST crossing number.

▶ **Theorem 1.1.** *There exists a constant $c > 0$ such that, for every generic set P of n points in the plane,* $\text{cr-MST}(P) < cn$.

Theorem 1.1 appears in [11],[3] but the authors did not include a proof of a crucial lemma [[11], Lemma 3], and we also did not succeed in clarifying the proof with them. We include our proof of the lemma with a different constant (Lemma 2.2) for the sake of completeness.

▶ **Theorem 1.2.** *If P is a generic set of $n > 5$ points in the plane,* $\text{cr-MST}(P) \geq 1$.

The constant 5 in Theorem 1.2 is tight. See Figure 1. The same example shows that $\text{cr-MST}(P)$ is not monotone in $n = |P|$: Removing the topmost point yields a point set for which the bicolored MST crossing number increases to 1.

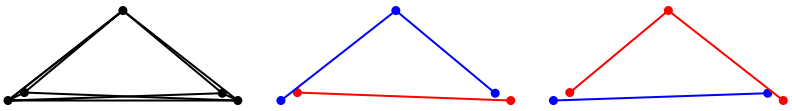


Figure 1 A set of 5 points with bicolored MST crossing number equal to 0. The middle and right pictures show the two MSTs for two different colorings. Interestingly, removing the top point increases the bicolored MST crossing number to 1.

We can strengthen the lower bound in two particular cases: (i) P is in convex position; (ii) the ratio between the maximum and minimum distances between the points in P is bounded (we call such sets *dense*).

1.2 Notation

For a point set X, we use $\mathrm{CH}(X)$ to denote the convex hull of X and $\partial\,\mathrm{CH}(X)$ to denote the boundary of $\mathrm{CH}(X)$. If X is generic, we use T_X to denote the MST of X.

Missing proofs can be found in the full version of the paper [3].

[3] Even though it is not stated explicitly, the authors of [11] also assume that point sets have a unique minimum spanning tree.

2 The general case

Before proving Theorem 1.1, we need the following two technical lemmas.

▶ **Lemma 2.1.** *Let $a, b, c, d \in \mathbb{R}^2$ be four points such that the line segments ab and cd cross at a point x. Suppose that $|ab|, |cd| \geq 2$, $|ax| \leq |xb|$, $|cx| \leq |xd|$, and $\angle axc \leq 1°$. Then $\max\{|ab|, |cd|\} \geq \max\{|ac|, |bd|\} + 0.5$.*

▶ **Lemma 2.2.** *Let P, S be two disjoint point sets. Then there is a constant $c > 0$, independent from P and S, such that any edge e of T_P crosses at most c edges of T_S whose length is at least the length of e.*

Proof. Let e be an edge of T_P and assume that $|e| = 2$ and that e is horizontal. Let $C > 2$ be a constant. Assume that e is crossed by at least $720C + 2$ edges of T_S whose length is at least the length of e. Then, at least $360C + 1$ of those edges have half or more of their length in the same halfplane determined by the line through e. Associate to each such edge e' a tuple (χ, α), where $\chi \in (0, 2)$ is the distance from the left endpoint of e to the crossing point between e and e' and $\alpha \in (0, \pi)$ is the size of the angle defined by the portion of the plane lying above e and to the right of e'. By the pigeonhole principle we can find two such edges $e_1 = uv, e_2 = xy$ with tuples (χ_1, α_1) and (χ_2, α_2) such that $|\chi_1 - \chi_2| \leq \frac{1}{C}$ and $|\alpha_1 - \alpha_2| \leq 1°$. In particular, the lines through e_1 and e_2 cross at an angle less than $1°$. Let us assume that u and x are on the same side of e and that this is the side that contains the shorter portions of e_1, e_2. Now, translate the edge $e_2 = xy$ by distance at most $\frac{1}{C} < 0.5$, so that it crosses e at the same point as e_1. Call this translate $x'y'$. By Lemma 2.1 we see that

$$|ux| \leq |ux'| + |xx'| \leq \max\{|uv|, |x'y'|\} = \max\{|uv|, |xy|\}.$$

Similarly, we see that

$$|vy| \leq |vy'| + |y'y| \leq \max\{|uv|, |x'y'|\}.$$

Therefore, one of the edges uv, xy is the longest edge of the cycle $uvyx$ in the complete geometric graph on S and cannot be in T_S. We obtain a contradiction. ◀

We note that the proof of Lemma 2.2 is quite simple, but the constant we obtain is at least 1442, which is significantly higher than the constant 8 claimed in [[11], Lemma 3].

We now finalize the proof of Theorem 1.1.

Proof of Theorem 1.1. Consider a coloring $P = R \cup B$. For each edge e in T_R, let $\phi_R(e)$ be the number of crossings between e and an edge e' of T_B such that $|e'| \geq |e|$. We define ϕ_B similarly. Clearly, $\mathrm{cr}(R, B) \leq \sum_{e \in T_R} \phi_R(e) + \sum_{e \in T_B} \phi_B(e)$. By Lemma 2.2, ϕ_R and ϕ_B are upper bounded by a constant c, so the result follows. ◀

We next prove Theorem 1.2.

Proof of Theorem 1.2. We consider two cases.

Case 1 The convex hull of P has at least 4 vertices.

We color two non adjacent vertices of the convex hull of P red and color all of the other points of P blue. Then T_R consists of only one edge that splits B into two parts, so at least one edge of T_B crosses it. See Figure 2 (left).

Case 2 The convex hull of P has three vertices.

Let a, b, c be the three vertices of the convex hull of P. Assume that ab is at least as long as the other edges of the convex hull, ab is horizontal and a is to the left of b.

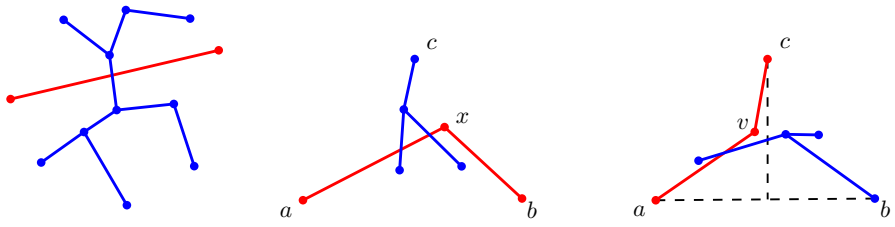


Figure 2 Colorings of the proof of Theorem 1.2.

Case 2.1 There is a point $x \in P$, $x \neq c$, such that $\triangle axb$ contains a point of P in the interior.
We color a, x, b red and all other points of P blue. T_R has two edges: ax and bx. Let Y be the set of points of P in the interior of $\triangle axb$. At least one edge of T_B connects a (blue) point of Y to a blue point that is not in Y, and this edge crosses ax or bx. See Figure 2 (middle).

Case 2.2 $\triangle axb \cap (P \setminus \{a, x, b\}) = \emptyset, \forall x \in (P \setminus \{a, b, c\})$.
Sort the points in $(P \setminus \{a, b, c\})$ by their x-coordinate (note that the assumptions of this case guarantee that no two points in $P \setminus \{a, b, c\}$ have the same x-coordinate). Let v be either the middle point or (if $n - 3$ is even) one of the two middle points in this ordering. Since $n > 5$, there are points of $(P \setminus \{a, b, c\})$ both to the left and right of v. Assume that the x-coordinate of v is smaller than or equal to the x-coordinate of c. This implies that ac is the longest edge of $\triangle(a, v, c)$. Now, we color a, v, c red and all the other points blue. By the same argument as in Case 2.1, there is a crossing between T_R and T_B. See Figure 2 (right).

◀

3 Special cases

In this section, we prove that the lower bound on cr-MST(P) given in Theorem 1.2 can be greatly improved in two significant particular cases: point sets in convex position and dense point sets.

3.1 Convex Point Sets

If the point set is in convex position, we can prove the following linear lower bound:

▶ **Theorem 3.1.** *Let P be a generic set of n points in the plane in convex position. Then* cr-MST(P) $\geq \lfloor \frac{n}{2} \rfloor - 1$.

Proof. We color P using an alternating coloring along $\partial\,\mathrm{CH}(P)$ (see Figure 3). Then $||R| - |B|| \in \{0, 1\}$. Assume that $|R| \geq |B|$. Then T_R, together with $\partial\,\mathrm{CH}(P)$, splits $CH(P)$ into cells such that no two blue points are in the same cell. Thus, any edge of T_B crosses at least one edge between two cells, and this edge is necessarily an edge of T_R. Since T_B has $\lfloor \frac{n}{2} \rfloor - 1$ edges, we conclude that $\mathrm{cr}(R, B) \geq \lfloor \frac{n}{2} \rfloor - 1$. ◀

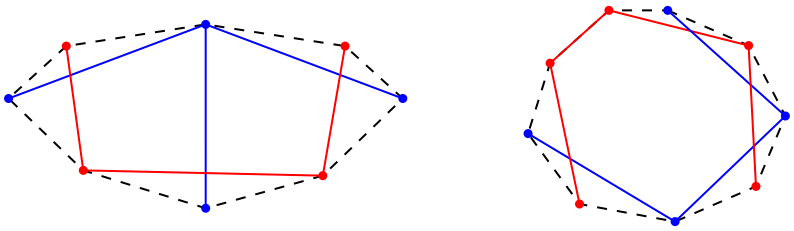


Figure 3 Coloring of the proof of Theorem 3.1.

3.2 Dense Point Sets

Next, we focus on the so-called *dense* point sets, that is, (generic) sets with bounded ratio between the maximum and minimum distances defined by the points. Various problems for dense point sets have been studied since the late 1980's [1, 4, 6, 7, 8, 9, 12, 15, 16, 17].

Formally, let P be a set of n points in generic position such that $\min_{a,b\in P}(d(a,b)) = 1$ (this can always be achieved by rescaling the set). The diameter of P is known to be at least $c\sqrt{n}$, where $c > 0$ is a constant. If n is sufficiently large, $c \geq \frac{\sqrt{2}\sqrt[4]{3}}{\sqrt{\pi}}$ [15]. We say that P is α-*dense* if the diameter of P is less than $\alpha\sqrt{n}$, for some constant $\alpha > 0$.

We define the *minimum bounding box* of P to be a minimum area square with integer-length edges that contains P. We further denote the area of the minimum bounding box of P by $B(P)$. Then, we define the *density* of P as $\phi(P) = \frac{|P|}{B(P)}$. This notion of density is more useful for our purposes and is closely related to the original definition of α-density.

▶ **Lemma 3.2.** *Let P be a generic α-dense set of n points in the plane for some $\alpha > 1$. Then $\phi(P) \geq \frac{1}{\lceil\alpha\rceil^2}$.*

The following corollary can be obtained by an easy packing argument.

▶ **Corollary 3.3.** *For any $\delta \geq 3$, there is a number $k(\delta) \in \mathbb{N}$ such that there is no set P of more than $k(\delta)$ points in the plane with $\phi(P) = \delta$.*

For a set of points P and integer k we say that P can *fill in* a $k \times k$ grid if we can place a (possibly rescaled) copy of P inside the square $[0, k]^2$ in such a way that each subsquare of the grid $\{0, 1, \ldots, k\} \times \{0, 1, \ldots, k\} \subset \mathbb{Z} \times \mathbb{Z}$ contains at least one point of P in its interior.

▶ **Lemma 3.4.** *Let P be a generic set of points in the plane. If there is a subset $P' \subseteq P$ that can fill in a 101×101 grid, there is a coloring of P' such that T_R and T_B cross at least once, regardless of how we color the points of P outside the 101×101 grid.*

In order to apply Lemma 3.4, we need the following key observation.

▶ **Lemma 3.5.** *Let P be a generic set of n points in the plane such that $\phi(P) = \delta$ and let $k \in \mathbb{N}$ be a positive integer. Then, P fills in a $k \times k$ grid or there is a subset $Q \subset P$ such that $\phi(Q) \geq (1 + \frac{1}{k^2-1})\delta$ and $|Q| \geq \frac{1}{k^2-1}|P|$.*

Lemma 3.4 and Lemma 3.5 yield the following:

▶ **Lemma 3.6.** *Let P be a generic α-dense set of $n > k(2)(101^2 - 1)^\zeta$ points in the plane, where $\alpha > 1$, $\zeta = \log_{1+\frac{1}{101^2-1}}(2\alpha^2)$ and $k(2)$ as in Corollary 3.3. Then, there is a coloring of P such that $\mathrm{cr}(R, B) \geq 1$.*

Proof. Since P is α-dense, $\phi(P) \geq \frac{1}{\lceil \alpha \rceil^2}$ (Lemma 3.2). We repeatedly apply Lemma 3.5 to P with $k = 101$. If, in some iteration, we obtain a point set that fills in a 101×101 grid, we apply Lemma 3.4 on this set and obtain a crossing. Note that in the coloring from Lemma 3.4 all points inside the set except for two are colored blue. We color the rest of P also blue.

It remains to argue that, if we repeatedly apply Lemma 3.5 to P with $k = 101$ at most $\log_{1+\frac{1}{101^2-1}}(3\alpha^2)$ times, in some iteration we obtain a point set that fills in a 101×101 grid. Suppose, for the sake of contradiction, that this is not true. After the last iteration we would obtain a point set Q with $\phi(Q) > 3$ and $|Q| > cn$, for $c = (101^2 - 1)^{-\log_{1+\frac{1}{101^2-1}}(3\alpha^2)}$. However, this is impossible if $n > k(3)/c$, where $k(3)$ is the value from Corollary 3.3. ◀

We are now ready to state and prove the main result of this subsection.

▶ **Theorem 3.7.** *Let P be a generic α-dense set of n points for $\alpha \geq \frac{\sqrt{2}\sqrt[4]{3}}{\sqrt{\pi}}$. If n is sufficiently large, then* cr-MST$(P) \geq cn$ *for some constant $c > 0$ that depends only on α.*

Proof. Since P is α-dense, it is contained in a square Q of side-length $\alpha\sqrt{n}$. We define $r = k(2)(101^2 - 1)^{\log_{1+\frac{1}{101^2-1}}(2\alpha^2)}$ and $x = \sqrt{r}\alpha^2$. We next split Q into $\frac{n\alpha^2}{x^2} = \frac{n}{r\alpha^2}$ cells of side-length x. By a standard argument for dense point sets (see, e.g., Lemmas 12 and 13 in [7]), we get that, if n is sufficiently large, there are $\Omega(n)$ cells with at least r points inside each of them; we call these cells *rich*. We choose a subset of rich cells in such a way that if a particular cell is chosen, none of the 24 cells closest to it will be chosen. In this way, we can again choose $\Omega(n)$ rich cells. Finally, for each rich cell C, $|P \cap C| \geq r$ and the diameter of $|P \cap C|$ is at most $\sqrt{2r}\alpha^2$. Thus, $P \cap C$ is $\sqrt{2}\alpha^2$-dense and we can apply Lemma 3.6 to obtain a crossing inside each of the chosen rich cells. Note that, in the obtained coloring, in each such cell exactly two points are red. Finally, we color the rest of the set P blue. ◀

4 Concluding remarks

There are many questions still left to explore about the bicolored MST crossing number.

The first and most important problem, in our opinion, is improving the lower bound for generic point sets. Currently, the best lower bound is 1, but we actually conjecture that cr-MST$(P) \in \Omega(n)$ for any generic set P of n points in the plane.

In the other direction, it would be interesting to improve the constant from Lemma 2.2 and determine its exact value. We remark that this constant is at least 6, as is witnessed by a configuration described in [3].

From a complexity-theoretic point of view, it would be interesting to determine whether the bicolored MST crossing number of a given point set can be computed in polynomial time. We conjecture that this is not possible.

Acknowledgments. T. Antić, M. Saumell, F. Schröder and P. Valtr were supported by grant no. 23-04949X of the Czech Science Foundation (GAČR). J. Tkadlec was supported by GAČR project 25-17221S and Charles Univ. projects UNCE 24/SCI/008, PRIMUS 24/SCI/012.

References

1 Noga Alon, Meir Katchalski, and William R. Pulleyblank. The maximum size of a convex polygon in a restricted set of points in the plane. *Discrete & Computational Geometry*, 4(3):245–251, June 1989. doi:10.1007/bf02187725.

2 Afrouz Jabal Ameli, Faezeh Motiei, and Morteza Saghafian. On the MST-ratio: Theoretical Bounds and Complexity of Finding the Maximum, 2025. arXiv:2409.11079.

3 Todor Antić, Morteza Saghafian, Maria Saumell, Felix Schröder, Josef Tkadlec, and Pavel Valtr. How many times can two minimum spanning trees cross?, 2026. arXiv:2601.20060.

4 Boris Bukh and Dong Zichao. Convex polytopes in restricted point sets in $\mathbb{R}^d$. *Advances in Combinatorics*, January 2025. doi:10.19086/aic.2025.1.

5 Sebastiano Cultrera di Montesano, Ondřej Draganov, Herbert Edelsbrunner, and Morteza Saghafian. The Euclidean MST-Ratio for Bi-Colored Lattices. In *GD 2024*, volume 320 of *LIPIcs*, pages 3:1–3:23. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024. doi:10.4230/LIPIcs.GD.2024.3.

6 Adrian Dumitrescu and János Pach. Partitioning complete geometric graphs on dense point sets into plane subgraphs. In *GD 2024*, volume 320 of *LIPIcs*, pages 9:1–9:10. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024. doi:10.4230/LIPICS.GD.2024.9.

7 Adrian Dumitrescu, János Pach, and Géza Tóth. Two trees are better than one. *SIAM Journal on Discrete Mathematics*, 39(3):1883–1893, 2025. doi:10.1137/24M1685365.

8 Adrian Dumitrescu and Csaba D. Tóth. Finding points in convex position in density-restricted sets, 2022. arXiv:2205.03437.

9 Herbert Edelsbrunner, Pavel Valtr, and Emo Welzl. Cutting dense point sets in half. *Discrete & Computational Geometry*, 17(3):243–255, April 1997. doi:10.1007/pl00009291.

10 Mikhail Binnewies et. al. Understanding the tumor immune microenvironment (time) for effective therapy. *Nature Medicine*, 24:541–550, 2018.

11 Mikio Kano, Criel Merino, and Jorge Urrutia. On plane spanning trees and cycles of multicolored point sets with few intersections. *Information Processing Letters*, 93(6):301–306, 2005. doi:10.1016/J.IPL.2004.12.003.

12 István Kovács and Géza Tóth. Dense point sets with many halving lines. *Discrete & Computational Geometry*, 64(3):965–984, March 2019. doi:10.1007/s00454-019-00080-3.

13 J.L. Maître, H. Berthoumieux, S.F. Krens, G. Salbreux, F. Jülicher, E. Paluch, and C.P. Heisenberg. Adhesion functions in cell sorting by mechanically coupling the cortices of adhering cells. *Science*, 338(6104):253–256, 2012.

14 G. Palla, D. S. Fischer, A. Regev, and F. J. Theis. Spatial components of molecular tissue biology. *Nature Biotechnology*, 40:308–318, 2022.

15 Pavel Valtr. Convex independent sets and 7-holes in restricted planar point sets. *Discrete & Computational Geometry*, 7(2):135–152, February 1992. doi:10.1007/bf02187831.

16 Pavel Valtr. *Planar point sets with bounded ratios of distances*. Phd thesis, Free University Berlin, 1994.

17 Pavel Valtr. Lines, line-point incidences and crossing families in dense sets. *Combinatorica*, 16(2):269–294, June 1996. doi:10.1007/bf01844852.

On geodesic disks enclosing many points*

Prosenjit Bose[1], Guillermo Esteban[2], David Orden[2], Rodrigo I. Silveira[3], and Tyler Tuttle[1]

1 **School of Computer Science, Carleton University, Canada**
jit@scs.carleton.ca, TylerTuttle@cmail.carleton.ca
2 **Departamento de Física y Matemáticas, Universidad de Alcalá, Spain**
{g.esteban, david.orden}@uah.es
3 **Departament de Matemàtiques, Universitat Politècnica de Catalunya, Spain**
rodrigo.silveira@upc.edu

Abstract

Let $\Pi(n)$ be the largest number such that for every set S of n points in a polygon P, there always exist two points $x, y \in S$, where every geodesic disk containing x and y contains $\Pi(n)$ points of S. We establish a lower bound for $\Pi(n)$, and show that $\lceil \frac{n}{5} \rceil + 1 \leq \Pi(n)$. We also show that there always exist two points $x, y \in S$ such that every geodesic disk with x and y on its boundary contains at least $\frac{n}{7+\sqrt{37}} \approx \lceil \frac{n}{13.1} \rceil$ points both inside and outside the disk. For the special case where the points of S are restricted to be the vertices of a geodesically convex polygon we give a tight bound of $\lceil \frac{n}{3} \rceil + 1$. We provide the same tight bound when we only consider geodesic disks having x and y as diametral endpoints.

Related Version arXiv:2506.06477 and [6]

1 Introduction

Given a set S of n points in the plane in general position—no three of them are collinear and no four of them are cocircular—there always exist two points $u, v \in S$ such that any disk that contains u and v also contains a constant fraction of S. Neumann-Lara and Urrutia [13] proved that this constant fraction is at least $\lceil \frac{n-2}{60} \rceil$. This bound was improved in a series of papers [9, 10] culminating in the best known lower bound of about $\frac{n}{4.7}$ by Edelsbrunner et al. [8]. Hayward et al. [10] gave an upper bound by constructing a set of n points such that for every pair of points u, v, there exists a disk with u, v on its boundary containing less than $\lceil \frac{n}{4} \rceil$ points, thereby showing that $\lceil \frac{n}{4} \rceil + 1$ is an upper bound for the problem. In addition, they studied the problem for point sets in convex position, giving a tight bound of $\lceil \frac{n}{3} \rceil + 1$. Another version of the problem is to consider disks in the plane having the points u and v as diametral endpoints [1]. In this case, a tight bound of $\lceil \frac{n}{3} \rceil + 1$ for both the convex and non-convex cases was shown.

The best known lower bound of $\left(\frac{1}{2} - \frac{1}{\sqrt{12}}\right) n + O(1) \approx \frac{n}{4.7}$ was obtained in multiple ways, see [7, 8, 14]. Edelsbrunner et al. [8] were the first to show this using techniques related to the order-k Voronoi diagram. Ramos and Viaña [14] used known results about j-facets of point sets in $\mathbb{R}^3$. In fact, these results allowed Ramos and Viaña to prove that there is always a pair of points such that any disk through them has at least $\frac{n}{4.7}$ points both inside and outside the disk.

We note that all the upper bounds in the Euclidean setting translate directly to the geodesic setting by enclosing the constructions in a large enough triangle. Thus, in this

* G. E., D. O. and R. S. are partially supported by grant PID2023-150725NB-I00 funded by MICIU/AEI/10.13039/501100011033. P. B. and T. T. are supported in part by NSERC.

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

Variant (see Def. 1.1)	Euclidean setting		Geodesic setting	
	Lower bound	Upper bound	Lower bound	Upper bound
$\Pi(n)$	$\lfloor \frac{n}{4.7} \rfloor$ [8]	$\lceil \frac{n}{4} \rceil + 1$ [10]	$\lceil \frac{n}{5} \rceil + 1$ (here)	$\lceil \frac{n}{4} \rceil + 1$
$\overline{\Pi}(n)$	$\lceil \frac{n}{3} \rceil + 1$ [10]	$\lceil \frac{n}{3} \rceil + 1$ [10]	$\lceil \frac{n}{3} \rceil + 1$ (here)	$\lceil \frac{n}{3} \rceil + 1$
$\Pi^{in\text{-}out}(n)$	$\lfloor \frac{n}{4.7} \rfloor$ [14]	$\lceil \frac{n}{4} \rceil + 1$ [10]	$\lceil \frac{n}{13.1} \rceil + 1$ (here)	$\lceil \frac{n}{4} \rceil + 1$
$\Pi^{diam}(n)$	$\lceil \frac{n}{3} \rceil + 1$ [1]	$\lceil \frac{n}{3} \rceil + 1$ [1]	$\lceil \frac{n}{3} \rceil + 1$ (here)	$\lceil \frac{n}{3} \rceil + 1$
$\overline{\Pi}^{diam}(n)$	$\lceil \frac{n}{3} \rceil + 1$ [1]	$\lceil \frac{n}{3} \rceil + 1$ [1]	$\lceil \frac{n}{3} \rceil + 1$ (here)	$\lceil \frac{n}{3} \rceil + 1$

Table 1 Summary of our results, comparing the Euclidean and geodesic settings.

paper, we focus mainly on the lower bounds, and address all variants of the problem in the case where S is a set of n points inside a simple polygon P. In addition, throughout the paper, we consider that $n \geq 21$, so that all our results hold. We summarize all our results and compare them with the Euclidean setting in Table 1.

1.1 Preliminaries and notation

Given a pair of points u, v in a (simple) polygon P, the *geodesic* (or *shortest*) *path* from u to v in P is denoted by $g(u, v)$. It is well-known that $g(u, v)$ is a polygonal chain. The length of this path is the sum of the lengths of its edges and is denoted by $|g(u, v)|$.

A subset P' of P is called *geodesically convex* if, for all pairs of points $u, v \in P'$, the geodesic path $g(u, v)$ is contained in P'. The *geodesic convex hull* of a set S of points in P is the geodesically convex set of minimum area that contains S. We say that S is in *geodesically convex position* if every point of S is on the boundary of the geodesic convex hull of S. A *geodesic triangle* on three points $a, b, c \in P$, denoted $\triangle(a, b, c)$, is a weakly-simple polygon whose boundary consists of $g(a, b)$, $g(b, c)$, and $g(c, a)$.

A *geodesic disk* centered at $c \in P$ with radius $r \geq 0$ is the set $D(c, r) = \{x \in P : |g(c, x)| \leq r\}$. When we refer to a disk *through* a set of points, we mean that the points are on the boundary of the disk. Given two points u and v, there always exists a geodesic disk through u and v centered at c_{uv}, the midpoint of $g(u, v)$, called the *diametral geodesic disk*.

A set S' of at least three points in P is *geodesically collinear* if $\exists x, y \in S'$ such that $S' \subset g(x, y)$, see set $\{x, c, y\}$ in Figure 1. Similarly, a set S'' of at least four points is *geodesically cocircular* if there exists a geodesic disk with S'' on its boundary, see set $\{w, x, y, z\}$ in Figure 1. A set S of n points in a polygon P is said to be in *general position* if no three points in S are geodesically collinear and no four points in S or the boundary of P are geodesically cocircular. In this paper we only consider sets S in general position in polygons P with the extra condition that there are no two points $x, y \in S$ and a reflex vertex p on the boundary of P such that $y \subset g(x, p)$.

Given two points $u, v \in S$, the bisector $b(u, v)$ is the set of all points that are equidistant to u and v. In this paper, as in [2], we assume that no bisector $b(u, v)$ passes through a vertex of P. Hence, $b(u, v)$ is a continuous curve connecting two points on the boundary of P.

▶ **Definition 1.1.** Let S be a set of n points in a polygon P.

- $\Pi(n)$ (resp., $\overline{\Pi}(n)$) is the largest integer such that, for every set (resp., geodesically convex set) S, there exist points $u, v \in S$ such that every geodesic disk through them contains $\Pi(n)$ (resp., $\overline{\Pi}(n)$) points of S.
- $\Pi^{in\text{-}out}(n)$ is the largest integer such that, for every set S, there exist points $u, v \in S$

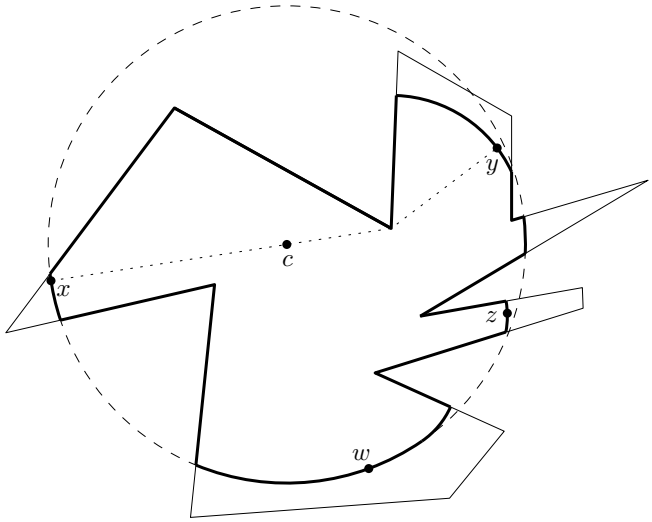


Figure 1 Geodesic disk centered at c, with radius $|g(c,z)|$, shown solid, and the equivalent Euclidean disk, shown dashed.

such that every geodesic disk through them has $\Pi^{in\text{-}out}(n)$ points of S, both inside and outside.

- $\Pi^{diam}(n)$ (resp., $\overline{\Pi}^{diam}(n)$) is the largest integer such that, for every set (resp., geodesically convex set) S, there exist points $u, v \in S$ such that $D\left(c_{uv}, \frac{|g(u,v)|}{2}\right)$ contains $\Pi^{diam}(n)$ (resp., $\overline{\Pi}^{diam}(n)$) points of S.

1.2 Some differences between Euclidean and geodesic disks

The problem of studying the largest number of points contained in any geodesic disk through two points inside a polygon is distinct from the Euclidean case. As illustrated in Figure 1, a geodesic disk of radius r in a polygon P can be strictly contained in a Euclidean disk of radius r. As such, in this setting, it is unclear whether there always exists a pair of points such that any geodesic disk through them can contain the same number of points as in the Euclidean setting.

Also, the geodesic metric, unlike the Euclidean metric, does not necessarily have bounded doubling dimension since it depends on the number of reflex vertices of P. Another difference is that the geodesic bisector of a pair of points inside a simple polygon can intersect a line segment more than once [5].

The order-k Voronoi diagrams were one of the tools used to prove the lower bounds in the Euclidean setting, see, e.g., [7, 8, 14]. However, some important properties of order-k Voronoi diagrams do not extend to the geodesic setting. For instance, in the Euclidean setting, the bisector of a pair of points is a line, whereas in the geodesic setting, the bisector is a finite curve. Moreover, this means that in the Euclidean case, three points in general position uniquely define a disk with those points on the boundary. On the contrary, given three points inside a polygon P, there may not exist a point in P equidistant to all three. Thus, there are cases where three points do not define a geodesic disk with the three points on the boundary [3].

Another difference between the two settings is that the well-known lifting transformation

that maps a point $(x, y) \in \mathbb{R}^2$ to the point $(x, y, x^2 + y^2) \in \mathbb{R}^3$ used in [14] to obtain the lower bound of $\frac{n}{4.7}$ is no longer applicable in the geodesic setting. The exact number of edges of the order-k Voronoi diagram is known in the Euclidean case [12]. The formula for this exact number of edges was fundamental to Edelsbrunner et al.'s lower bound proof [8]. However, only upper bounds are known in the geodesic case [4]. In fact, this is one of the reasons why the lower bounds we obtain in the geodesic setting are not as strong as in the Euclidean setting.

2 The general case

In this section we provide a generalization to the geodesic setting of lower bounds that are known for the original problem in the plane. In particular, we generalize results from [8].

▶ **Theorem 2.1.** *For any set S of n points in a polygon, $\Pi(n) \geq \lceil \frac{n}{5} \rceil + 1$.*

Proof. In this proof, order-k diagrams refer to order-k geodesic Voronoi diagrams. Recall that a geodesic disk through two points containing $k-1$ points in its interior has its center on an edge of the order-k diagram. Consider the set of all disks through two points $p, q \in S$ in the polygon. These disks are all centered on the bisector $b(p, q)$, and there are at most $n-2$ points on the bisector where a third point of S is on the boundary of the disk. This partitions the bisector into *at most* $n-1$ pieces such that any circle through p and q centered on any point belonging to a fixed piece s contains in its interior the same number of points $\rho(s)$. The pieces s such that $\rho(s) = k-1$ are precisely the edges of the order-k diagram.

Now consider some fixed bisector $b(p, q)$. The edges of the order-1 through order-k diagrams on $b(p, q)$ form a number of connected components (which is strictly less than the total number of edges of all the diagrams). Let λ_k be the number of connected components formed by *all* the bisectors of the order-1 through order-k diagrams. The goal is to find the largest value of k such that $\lambda_k < \binom{n}{2}$. Every bisector is completely covered by a single connected component when $k = n-2$. When $\lambda_k < \binom{n}{2}$, by the pigeonhole principle there exists a bisector $b(p, q)$ that does not have any edges of the order-1 through order-k diagrams. Therefore, every geodesic disk with p and q on its boundary must contain in its interior at least k points.

We derive a formula for λ_k in terms of the number of edges in the order-1 to the order-k diagram by considering how the pieces must "pair up", in the sense that a piece s with $\rho(s) = i$ can only be adjacent to pieces s' with $\rho(s') = i \pm 1$. If a piece s with $\rho(s) = k$ is adjacent to two pieces s_1 and s_2 with $\rho(s_1) = \rho(s_2) = k-1$, the number of connected components is decreased by one. If s is adjacent to only one piece s_1 with $\rho(s_1) = k-1$, it extends an existing component. Otherwise, if it is not adjacent to any piece s_1 with $\rho(s_1) = k-1$, it creates a new connected component. Let ϵ_a, ϵ_b and ϵ_c be, respectively, the number of pieces s with $\rho(s) = k$ of the first, second, and third type. Then we have

$$\lambda_k = \lambda_{k-1} + \epsilon_c - \epsilon_a \tag{1}$$

Let e_k be the number of edges in an order-k diagram and let S_k be the number of edges of the order-k diagram that intersect the boundary of the polygon. The total number of endpoints of all order-k edges is given by $2e_k - S_k$ since the edges that intersect the boundary of the polygon only contribute one endpoint. The edges intersecting the boundary are analogous to the unbounded edges in the Euclidean setting. Moreover, we note that $e_k = \epsilon_a + \epsilon_b + \epsilon_c$.

Let ϕ_i be the number of unpaired endpoints in λ_i. Consider all the unpaired endpoints of the connected components of λ_{k-1}. These endpoints can only come from edges in the

order-$k-1$ diagram since all edges of the order-i diagrams for $i \in \{1,\ldots,k-2\}$ are all paired. Therefore, these unpaired endpoints can only pair up with edges from the order-k diagram, which implies that $\phi_{k-1} = 2\epsilon_a + \epsilon_b$. Thus, from Equation (1) and the fact that $e_k = \epsilon_a + \epsilon_b + \epsilon_c$, we have that $\lambda_k = \lambda_{k-1} + e_k - \phi_{k-1}$. If we iterate over k, we get that $\lambda_k = (e_k - \phi_{k-1}) + (e_{k-1} - \phi_{k-2}) + \ldots + (e_2 - \phi_1) + e_1 = \sum_{i=1}^{k} e_i - \sum_{i=1}^{k-1} \phi_i$. Note that ϕ_{i-1} is the number of unpaired endpoints in λ_{i-1}, which are the paired endpoints of λ_i. Therefore, $\phi_i + \phi_{i-1} = 2e_i - S_i$. We use this to rewrite the equation for λ_k in terms of e_i and S_i.

$$\begin{aligned}
\lambda_k &= \sum_{i=1}^{k} e_i - [(\phi_{k-1} + \phi_{k-2}) + (\phi_{k-3} + \phi_{k-4}) + \ldots] \text{ (end of sum depends on parity of } k) \\
&= \sum_{i=1}^{k} e_i - [(2e_{k-1} - S_{k-1}) + \ldots + (2e_{2\text{-}or\text{-}1} - S_{2\text{-}or\text{-}1})] \text{ (depending on parity of } k) \\
&= \sum_{i=1}^{k} (-1)^{k-i} e_i + (S_{k-1} + S_{k-3} + \ldots + S_{2\text{-}or\text{-}1}) \ \ (S_2 \text{ or } S_1 \text{ depending on parity of } k)
\end{aligned}$$

To complete the bound on λ_k, we need to bound the size of the order-j diagrams. Exact bounds on the size of order-j Euclidean Voronoi diagrams are known but do not apply since unlike the Euclidean case, the geodesic order-j Voronoi diagram has no unbounded edges. We can, however, establish upper and lower bounds using abstract Voronoi diagrams [11] since an order-j geodesic Voronoi diagram is an order-j abstract Voronoi diagram. Hence, we have that the number of edges $e_j = 3(F_j - 1) - S_j$, where $F_j = 2jn - j^2 - n + 1 - \sum_{i=1}^{j-1} S_i$ is the number of faces of an order-j geodesic Voronoi diagram, see [4, Lemmas 15 and 16]. Thus, $e_j = 6jn - 3j^2 - 3n - 3\sum_{i=1}^{j-1} S_i - S_j$. Then we get

$$e_j - e_{j-1} = 6n - 6j + 3 - 2S_{j-1} - S_j. \tag{2}$$

In the final calculation, as in the proof of the lower bound in [8], we distinguish the case that k is even from k odd. However, in both cases we have that $\lambda_k = 3kn - \frac{3}{2}k^2 - \frac{3}{2}k - \sum_{i=1}^{k} S_i$. We use the fact that $j(j+1) \leq \sum_{i=1}^{j} S_i \leq j(2n-j-1)$, see [4, Lemma 11], which implies that $\lambda_k \leq 3kn - \frac{3}{2}k^2 - \frac{3}{2}k - k(k+1) = 3kn - \frac{5}{2}k^2 - \frac{5}{2}k$, which is a quadratic polynomial in k. We note that $\lambda_k < \binom{n}{2}$ whenever $k < \frac{n}{5}$, i.e., $k \leq \lceil \frac{n}{5} \rceil - 1$. Then, as mentioned above, the pigeonhole principle implies that every geodesic disk with p and q on its boundary must contain in its interior at least $k \leq \lceil \frac{n}{5} \rceil - 1$ points. Thus, we get that $\Pi(n) \geq \lceil \frac{n}{5} \rceil + 1$, with the plus one accounting for p and q. ◀

2.1 Other variants in the geodesic setting

In this section we provide a summary of the lower bounds that we obtained for some of the variants of the original problem in the geodesic setting. In particular, we generalize results from [1, 10, 14]. Due to space restrictions, all details are deferred to the full version.

First, the bound on $\Pi^{in\text{-}out}(n)$ is obtained by modifying the previous proof of Theorem 2.1. In addition to the parameter λ_k, we also consider the parameter $\overline{\lambda}_{n-k}$, the number of connected components formed by *all* the bisectors of the order-$(n-k)$ through order-$(n-1)$ diagrams. Then, the goal is to find the largest value of k such that $\lambda_k + \overline{\lambda}_{n-k} < \binom{n}{2}$ so, for any k that is less than approximately $\frac{n}{13.08}$, there exists a pair of points $\{p, q\}$ such that any geodesic disk with p and q on its boundary will contain at least k points and at most $n-k$ points.

Second, we borrow some of the ideas from [10] to find, among a set of points in geodesically convex position, a pair of points such that any geodesic disk through them contains at least

$\lceil \frac{n}{3} \rceil + 1$ points of S, showing that $\overline{\Pi}(n) \geq \lceil \frac{n}{3} \rceil + 1$. In order to prove it, we use the fact that if there is no enclosing geodesic disk through three points of S, then there exist two points of S such that any geodesic disk through them is an enclosing geodesic disk.

Finally, we consider the problem of proving a lower bound on the values of $\Pi^{diam}(n)$ and $\overline{\Pi}^{diam}(n)$, generalizing the results in [1] for the Euclidean setting. We prove that, among a set S of n points, there is a pair of points such that the diametral geodesic disk through them contains at least $\lceil \frac{n}{3} \rceil + 1$ points of S. In order to obtain these lower bounds, we are going to use the fact that there exists an enclosing geodesic disk D with (i) three points x, y, z on the boundary, and the center of D inside the geodesic triangle through x, y and z, or (ii) two points x, y on the boundary, and the center of D at the midpoint of $g(x, y)$.

3 Conclusion

We study different variants of the following question: Given a set S of n points in a polygon, does there always exist two points $x, y \in S$, such that every geodesic disk containing x and y contains a constant fraction of the points of S? This question and related ones have been studied intensely in the Euclidean setting [1, 7, 8, 9, 10, 13, 14] and in this work we consider their geodesic versions. We note that all the solutions to these problems lend themselves to polynomial time algorithms since the characterizations are based on standard structures in Computational Geometry such as order-k Voronoi diagrams. An obvious open problem is the following: Can our lower bounds be improved? Basically, some of our geodesic lower bounds are not as strong as in the Euclidean case. It would be interesting to determine whether there is a separation between the bounds in the Euclidean and geodesic case. Finally, the main open problem, which has been open since 1988, is whether we can find a tight bound for the original question.

References

1 Jin Akiyama, Y Ishigami, Masatsugu Urabe, and Jorge Urrutia. On circles containing the maximum number of points. *Discrete Mathematics*, 151(1-3):15–18, 1996.

2 Boris Aronov. On the geodesic Voronoi diagram of point sites in a simple polygon. In *Proceedings of the third annual symposium on Computational geometry*, pages 39–49, 1987.

3 Boris Aronov, Steven Fortune, and Gordon Wilfong. The furthest-site geodesic Voronoi diagram. *Discrete & Computational Geometry*, 9:217–255, 1993.

4 Cecilia Bohler, Panagiotis Cheilaris, Rolf Klein, Chih-Hung Liu, Evanthia Papadopoulou, and Maksym Zavershynskyi. On the complexity of higher order abstract Voronoi diagrams. *Computational Geometry*, 48(8):539–551, 2015.

5 Prosenjit Bose, Anthony D'Angelo, and Stephane Durocher. Approximating the smallest k-enclosing geodesic disc in a simple polygon. In *WADS*, volume 14079 of *Lecture Notes in Computer Science*, pages 179–192. Springer, 2023.

6 Prosenjit Bose, Guillermo Esteban, David Orden, Rodrigo I. Silveira, and Tyler Tuttle. On Geodesic Disks Enclosing Many Points. In Pat Morin and Eunjin Oh, editors, *19th International Symposium on Algorithms and Data Structures (WADS 2025)*, volume 349 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 10:1–10:20, Dagstuhl, Germany, 2025. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. URL: `https://drops.dagstuhl.de/entities/document/10.4230/LIPIcs.WADS.2025.10`, `doi:10.4230/LIPIcs.WADS.2025.10`.

7 Mercè Claverol, Clemens Huemer, and Alejandra Martínez-Moraian. On circles enclosing many points. *Discrete Mathematics*, 344(10):112541, 2021.

8 Herbert Edelsbrunner, Nany Hasan, Raimund Seidel, and Xiao Jun Shen. Circles through two points that always enclose many points. *Geometriae Dedicata*, 32(1):1–12, 1989.

9 Ryan Hayward. A note on the circle containment problem. *Discrete & Computational Geometry*, 4(3):263–264, 1989.

10 Ryan Hayward, David Rappaport, and Rephael Wenger. Some extremal results on circles containing points. *Discrete & Computational Geometry*, 4(3):253–258, 1989.

11 Rolf Klein. *Concrete and Abstract Voronoi Diagrams*, volume 400 of *Lecture Notes in Computer Science*. Springer, 1989.

12 Der-Tsai Lee. On k-nearest neighbor Voronoi diagrams in the plane. *IEEE transactions on computers*, 100(6):478–487, 1982.

13 Victor Neumann-Lara and Jorge Urrutia. A combinatorial result on points and circles on the plane. *Discrete Mathematics*, 69(2):173–178, 1988.

14 Pedro A Ramos and Raquel Viaña. Depth of segments and circles through points enclosing many points: a note. *Computational Geometry*, 42(4):338–341, 2009.

Centered flips of non-crossing, perfect matchings

Lisa Kudlik[1] **and Alexandra Wesolek**[2]

1 **Department of Mathematics, Technical University Berlin, Berlin, Germany**
`lisa.kudlik@gmail.com`
2 **Department of Mathematics, Technical University Berlin, Berlin, Germany**
`wesolek@tu-berlin.de`

Abstract

We study the diameter of flip graphs H_n whose node set is given by crossing-free drawings of perfect matchings. Specifically, we consider straight-line perfect matchings on a set of $2n$ vertices, with n odd, that are equidistantly placed on the unit circle which is centered at the origin. A centered flip of a matching is the replacement of two edges a and b by two edges c, d such that a,b,c,d form a crossing-free quadrilateral that contains the origin. Milich, Mütze and Pergel showed a lower bound of $n-1$ and an upper bound of $11n-29$ for the diameter of H_n. For $n \geq 5$, we improve the lower bound to $\frac{3}{2}n-1$ and for $n \geq 3$ we improve the upper bound to $6.5n-16$.

1 Introduction

Let $\mathcal{C}$ be a set of configurations. A *flip* is an allowed (local) operation that transforms a configuration $C \in \mathcal{C}$ into another configuration $C' \in \mathcal{C}$. The associated *flip graph* has vertex set $\mathcal{C}$, with an edge between two configurations C_1 and C_2 whenever one flip can transform C_1 into C_2. Flip graphs are of considerable interest as they promote the understanding of the relationships between different objects and their transformation possibilities (see [4] for a summary of geometric flip graphs). Topics that are widely discussed are the existence of Hamiltonian paths or cycles in flip graphs [8, 9] and the diameter of flip graphs [3, 5, 11, 13]. Hamiltonian paths are of interest because they provide a way of listing structures such that consecutively listed structures differ only by a small change (i.e. a flip). These enumerations are called Gray codes (which are surveyed in [12]). If there exists a Hamiltonian cycle in the flip graph, we have the choice of the starting element of the list. On the other hand, diameters of flip graphs are widely studied as they give an upper bound on the number of steps it takes to configure one object into another.

All matchings considered (unless explicitly stated otherwise) are perfect matchings, and we study the flip graph of non-crossing matchings on a set P of equidistantly placed points on the unit circle centered at the origin. A *general flip* of a non-crossing matching, shown in Figure 1, consists of removing two edges and adding two other edges such that the resulting geometric graph is a non-crossing matching. Flip graphs of perfect matchings of size n for general flips are well studied [2, 7]. For the path graph P, Hernando, Hurtado, and Noy [7] showed that the diameter of the corresponding flip graph is exactly $n-1$. In contrast, for an arbitrary point set in general position it remains a long-standing open problem whether the corresponding flip graph is even connected. A *centered flip* of a matching is the replacement of two edges a and b by two edges c and d such that a,b,c,d form a crossing-free quadrilateral that contains the origin. Note that a centered flip is also a general flip. In this paper, we study the more restricted centered flips, so in the following a *flip* is a centered flip if not specified otherwise.

Centered flips were first mentioned in the work of Felsner, Kleist, Mütze and Sering [6] on rainbow cycles of general flip graphs. These cycles are of interest as they correspond to

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

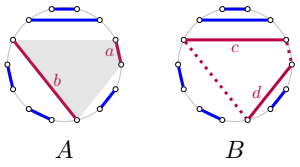


Figure 1 A centered flip from A to B is performed by replacing a, b with c, d.

balanced Gray codes [1]. In [6] it was shown that in a rainbow cycle of the flip graph of non-crossing matchings where flips are allowed to be general, in fact, all flips are centered. This motivates the study of centered flips.

We focus on the diameter of the flip graph H_n of non-crossing, perfect matchings of odd size n on P, where the flips are centered flips. Improving the bounds on its diameter given in [10], we give an upper bound of $6.5n - 16$ and a lower bound of $1.5n - 1$ on the diameter. In the article [10] by Milich, Mütze and Pergel a lower bound of $n - 1$ and an upper bound of $11n - 29$ for the diameter of H_n was proved (for odd n). For even $n \geq 4$, the flip graph H_n has multiple connected components and its diameter is therefore infinite.

1.1 Proof Overview

A standard proof technique to show that any matching A can be flipped to any matching B in s steps is to show that any matching A can be flipped to a fixed matching M in $\frac{s}{2}$ steps. In the work of Milich, Mütze and Pergel [10], the authors use two different matchings M_0 and M_1 to show that any matching can be flipped to either M_0 or M_1 in $4n - 12$ steps, which is how they attain their upper bound of $11n - 29$ (by adding the flips needed to go from M_0 to M_1). In order to describe M_0 and M_1, we define the *length* l of an edge ab in a matching M to be the minimum of the number of edges on the right or left side of the directed edge (a, b). An edge of length zero is called a *boundary edge*, see edge a in Figure 1. There are exactly two matchings whose edges all have length zero, which are exactly the mentioned matchings M_0 and M_1. In the following, v_i with $i \in \{1, ..., 2n\}$ are the points of P sorted clockwise around the circle. We define M_0 as the matching containing $\{v_{2n}, v_1\}$.

In fact, their proof can be adapted to get an upper bound of $8n - 20$ by showing that any matching can be flipped to M_0 in $4n - 10$ flips. The new idea, which we will focus on in this paper, is not to flip to a fixed intermediate matching, but rather to a matching M_φ chosen from a family of matchings. We show that there exists a matching M_φ from a set of n special matchings such that A and B can be flipped to M_φ in $3n - 9 + x$ and $3n - 9 + y$ steps, respectively, with $x + y \leq \frac{n}{2} + 2$, see Figure 2. The choice of the matching M_φ is tailored to the structural similarities of A and B. We can divide the flip-sequence into two different flip phases. In the first phase, we perform x flips starting from A and y flips starting from B, so that the resulting matchings A^* and B^* contain a common diagonal edge φ. A *diagonal edge* is a directed edge $\varphi = (v_i, v_j)$ between two antipodal points v_i, v_j on the unit circle. In the following it is convenient for us to work with directed edges in order to speak of a 'left' and a 'right' side of the edge. We say a matching *contains a directed edge* φ, if it contains the undirected edge with the same endpoints. We choose φ such that for x and y the inequality $x + y \leq \frac{n}{2} + 2$ holds. For the second phase we show that we need a maximum of $3n - 9$ steps from a matching containing φ to a specific matching M_φ which we define in the following. An important property of M_φ is that all but one edge cross φ and that it is diagonal-free, see Figure 2. We first define a partial matching $M_{\varphi,k}$ for all $k \leq \frac{n-3}{2}$, where a *partial matching* is a matching that is not perfect. These partial matchings are all submatching of M_φ.

For $\varphi^* = (v_1, v_{n+1})$ the partial matching $M_{\varphi^*,k}$ has the edges $b_i = \{v_{1-i}, v_{2+i}\}$ and $b'_i = \{v_{n-i}, v_{n+1+i}\}$, $i \in \{0, \dots, k\}$, where indices are taken modulo $2n$. If φ is obtained from φ^* through a rotation by angle θ around the origin then $M_{\varphi,k}$ is obtained from $M_{\varphi^*,k}$ through a rotation by angle θ around the origin. Note that $M_{\varphi,\frac{n-3}{2}}$ contains $2\left(\frac{n-1}{2}\right) = n-1$ edges. We let M_φ to be the unique matching containing $M_{\varphi,\frac{n-3}{2}}$.

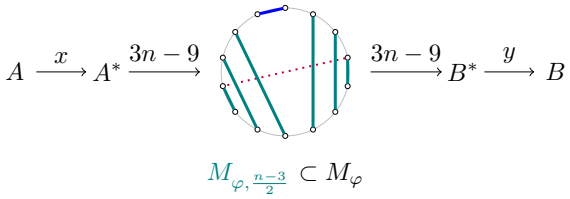


Figure 2 Proof idea for the upper bound $6.5n - 16$ where $x + y \leq \frac{n}{2} + 2$. The diagonal edge φ is depicted as a dotted line.

As mentioned, the article [7] deals with the diameter of G_n, which is the flip graph of non-crossing matchings on P where the flip operation is a general flip. It is shown that the diameter of G_n is exactly $n-1$. Therefore H_n, which is a spanning subgraph of G_n, has a diameter of at least $n-1$. We improve the lower bound to $1.5n-1$ by showing that any path between M_0 and M_1 has length at least $1.5n-1$ for $n \geq 5$. Since the matchings have no common edge, all n edges in M_0 must be removed and all n edges of M_1 must be added. We assign weight -1 to edges in M_0, weight $+1$ to edges in M_1, and weight 0 to all remaining edges. The total weight W_A of a matching A is defined as the sum of its edge weights, for example $W_{M_1} = n$. We consider only centered flips. A weight increase of 3 can occur only when $n = 3$. Hence, we assume $n \geq 5$, in which case a single flip changes the weight by at most 2. We show that if a flip with weight change 2 is performed, then either the resulting matching has a diagonal edge or the previous matching already had a diagonal edge. If the resulting matching does not have a diagonal edge, then the weight change in the next flip is at most 0. We get a lower bound by analyzing how frequently a weight change of 2 can happen by analyzing occurrences of diagonal edges. To relate the weight change to the length of the shortest path between two matchings A and B, we define

$$\xi_{A,B} = \frac{W_B - W_A}{\text{shortest path length from } A \text{ to } B}.$$

We show that $\xi(D_1, D_2) \leq \frac{4}{3}$ for two matchings D_1, D_2 which both have a diagonal edge (see 3). We use this to bound $\xi = \xi_{M_0,M_1}$ from above which yields a lower bound on the length of a shortest flip sequence from M_0 to M_1, namely $(W_{M_1} - W_{M_0})\frac{1}{\xi} = \frac{2n}{\xi}$. The *length* of a flip sequence is its number of flips.

2 Results on the upper bound

▶ **Theorem 2.1** (Upper Bound). *The diameter of H_n for odd $n \geq 3$ is at most $6.5n - 16$.*

In order to flip from a matching A to B, we modify these matchings in $x + y$ steps to make them more similar. The modified matchings A^* and B^* contain a common diagonal edge φ. Using Theorem 2.2 we can show that there exists x, y such that $x + y \leq \frac{n}{2} + 2$. For the second step, we use Theorem 2.4 to show that A^* and B^* can each be transformed into M_φ in at most $3n - 9$ steps.

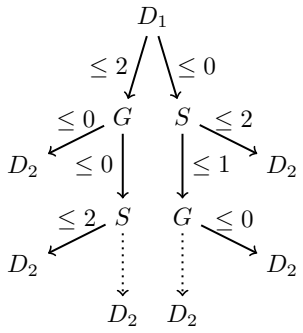


Figure 3 Depicted is a flip sequence from $D_1 \in \mathcal{D}$ to $D_2 \in \mathcal{D}$. An upper bound on the weight change for each step is indicated. For example, making a flip from a matching in $\mathcal{S}$ to a matching in $\mathcal{D}$ changes the weight by at most 2 and to a matching in G by at most 1.

The diagonal edge $\varphi = (v, w)$ from v to w can be divided into two line segments. We call the segment from v to the midpoint the *lower line segment* of φ, and the segment from the midpoint to w the *upper line segment* of φ. Note that a diagonal edge $d \neq \varphi$ intersects φ at the origin; in this case, we say that d intersects both the upper and lower line segments of φ.

2.1 Phase 1:

For a set of segments S let $\mathrm{cr}(S)$ be the number of properly intersecting pairs of segments of S, that is, intersections that do not occur at endpoints. Let A be a non-crossing matching and let $C = A \cup \varphi$ be the matching A with an added diagonal edge φ. Since A is intersection-free, all intersections in C involve φ, and we write $\mathrm{cr}(A, \varphi)$ instead of $\mathrm{cr}(C)$. Now we can define $\mathrm{cr_u}(A, \varphi)$ as the number of crossings between the upper line segment of φ and edges in A. For the number of crossing between the lower line segment of φ and edges in A we will write $\mathrm{cr}_l(A, \varphi)$.

We define the set of all diagonal edges as Φ and a function $h_A : \Phi \to \mathbb{N}$ which upper bounds the minimal number of flips needed to transform A to some non-crossing matching A', such that $A^* = A' \cup \varphi$ is intersection free, by which we mean intersections are at most at endpoints of φ. An edge is visible if the segments from the unit center to its endpoints do not intersect any other edge. To be more precise, let $\overline{A}$ be the plane graph obtained by adding all boundary edges to A. We call the *center face* of a matching A the face of $\overline{A}$ containing the origin. A *visible edge* of A is an edge which is contained in the center face of $\overline{A}$. A *visible vertex* is a vertex of a *visible edge*. The same definitions apply when A is a partial matching. In order to transform a matching A to A' such that $A^* = A' \cup \varphi$ is intersection-free, we do the following:
If φ is intersected by a diagonal edge in A, then the diagonal edge is flipped with a visible edge intersecting φ in the lower segment if $\mathrm{cr}_l(A, \varphi) > \mathrm{cr_u}(A, \varphi)$. If $\mathrm{cr}_l(A, \varphi) \leq \mathrm{cr_u}(A, \varphi)$ holds, then the diagonal edge can be flipped with an edge crossing φ in the upper segment. If φ is only crossed by another the diagonal edge, then the diagonal edge is flipped with an edge that shares an endpoint with φ. If φ is intersected but not by a diagonal edge, then there are exactly two visible edges, each of which either intersects φ or shares a common endpoint with φ. We flip those two visible edges. We repeat this process until both endpoints of φ are visible. Consequently, it follows that $A^* = A' \cup \varphi$ is intersection-free. We define

$$h_A(\varphi) = \max(\mathrm{cr}_l(A, \varphi), \mathrm{cr_u}(A, \varphi)). \tag{1}$$

$h_A(\varphi)$ is the minimum number of flips needed from A to some non-crossing matching A', such that $A^* = A' \cup \varphi$ is intersection free.

▶ **Theorem 2.2.** *Let A and B be matchings. Then there is at least one diagonal edge φ with,*

$$h_A(\varphi) + h_B(\varphi) < \frac{n}{2}.$$

We sketch the proof of Theorem 2.2 which will be available in the full version. We define a partial ordering on the set of matchings where $A \leq_h B$ if $h_A(\varphi) \leq h_B(\varphi)$ for every diagonal edge φ. We first study maximal matchings with respect to the ordering $\leq_h$. A non-centered flip of two edges a and b, as depicted in Figure 4, leads to a matching with edges c and d, which is a heavier matching with respect to $\leq_h$.

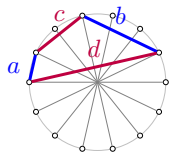
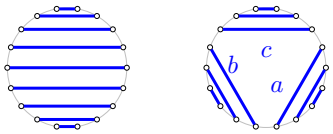


Figure 4 It shows that $\mathrm{cr}(a, \varphi_i) + \mathrm{cr}(b, \varphi_i) \leq \mathrm{cr}(c, \varphi_i) + \mathrm{cr}(d, \varphi_i)$ for every $i \in [1, n]$, leading to $\leq_h$-maximal matching, where all edges behind the visible ones are parallel to them.

From that we show that there are two possible structures for maximal matchings: either the edges are all parallel to each other (see Figure 5 (i)) or there are three visible edges a, b and c, where the edges behind them are parallel (see Figure 5 (ii)). We say that the latter is a *triangular matching* if the sum of the length of any two of a, b, c it at least $\frac{n-1}{2}$. We sketch the proof of Theorem 2.2, considering only $\leq_h$-maximal matchings. If A or B has the structure in Figure 5(i), the theorem is easy to prove, since the diagonal edge can be chosen as φ. Thus, assume both matchings have triangular structure. For any matching M with triangular structure, the average of $h_M(\phi)$ over all diagonal edges ϕ is less than $\frac{n}{4}$. Hence, there exists a diagonal edge φ such that A can be flipped to a diagonal-free matching in fewer than $\frac{n}{4}$ steps, same applys to B.



(i) (ii)

Figure 5 Only two possible structures for $\leq_h$-maximal matchings.(i) all parallel edges, (ii) triangular structure

Given two arbitrary matchings A and B, Theorem 2.2 states that there exists a diagonal edge φ such that the sets $A \cup \varphi$ and $B \cup \varphi$ are intersection-free after c steps, where $c \leq \frac{n}{2}$. Now it suffices to perform a single flip for each matching in order to incorporate the diagonal edge into each matching. To transition from two arbitrary matchings to two matchings A^* and B^*, that share a common diagonal edge φ, the maximum number of steps required is $\frac{n}{2} + 2$. We need this fact to prove Theorem 2.1.

2.2 Phase 2

To transform a matching A^* containing a diagonal edge φ into M_φ, we first modify A^* so that the following theorem can be applied repeatedly. The flip-sequence is shown in Figure 6 and has length 4.

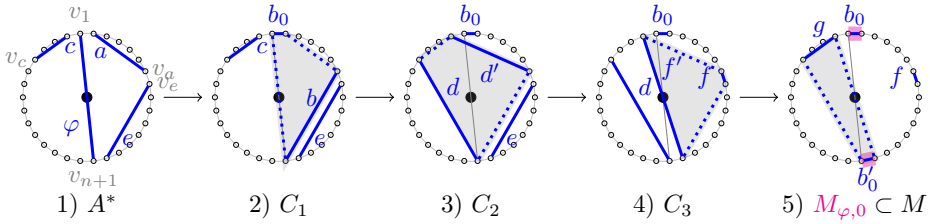


Figure 6 Flip sequence from a matching A^* with a diagonal edge φ to M containing $M_{\varphi,0}$.

▶ **Theorem 2.3.** *Let M be a matching without a diagonal edge, and let $M_{\varphi,k} \subset M$ for some diagonal edge φ and some $1 \leq k \leq \frac{n-5}{2}$, such that the visible edges of $M_{\varphi,k}$ are also visible in M. Then there exists a flip sequence of length at most 6 that transforms M into a matching M' with $M_{\varphi,k+1} \subset M'$, and the visible edges in $M_{\varphi,k+1}$ are visible in M'.*

The flip sequence of length 6 in Theorem 2.3 is visualized in Figure 7. It depicts how to add two new edges, b_k and b'_k, parallel to b_{k-1} and b'_{k-1} for $k \leq n-3$, respectively.

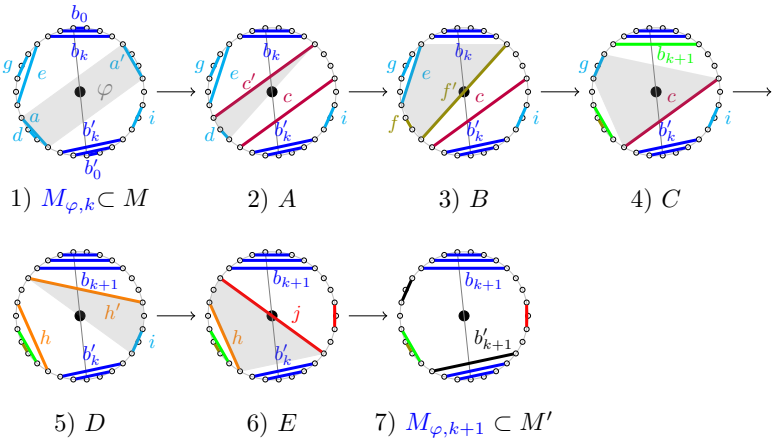


Figure 7 A depiction of how to flip from a diagonal-free matching containing $M_{\varphi,k}$ to $M_{\varphi,k+1}$ in 6 steps for $1 \leq k \leq \frac{n-5}{2}$, when the visible edges of $M_{\varphi,k}$ are also visible in the matching.

We apply Theorem 2.3 repeatedly until a matching containing $M_{\varphi,\frac{n-5}{2}}$ is obtained, which takes $\frac{n-5}{2}$ steps. We obtain M_φ once we reach $M_{\varphi,\frac{n-3}{2}}$, which can be reached from $M_{\varphi,\frac{n-5}{2}}$ with a flip sequence of length 2, as shown in Figure 8. Adding all steps from A^* to M, then to E_1 or E'_1, and finally to M_φ, we obtain

$$4 + 6\left(\frac{n-5}{2}\right) + 2 = 3n - 9.$$

▶ **Theorem 2.4.** *A matching A^* with a directed diagonal edge φ can be flipped in fewer than $3n-9$ flips to M_φ.*

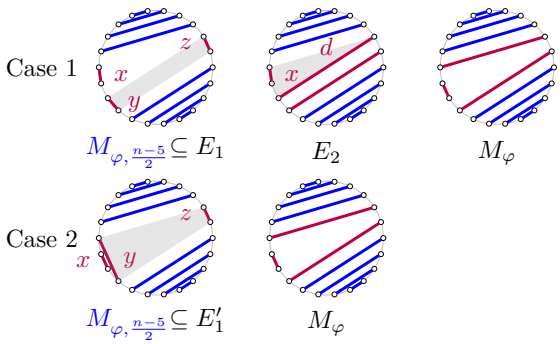


Figure 8 Two possible matchings E_1 and E_1' both containing $M_{\varphi,\frac{n-5}{2}}$ and the flip sequence from E_1 and E_1' to M_φ containing $M_{\varphi,\frac{n-3}{2}}$.

Acknowledgments. A.W. was supported by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy - The Berlin Mathematics Research Center MATH+ (EXC-2046/1, project ID:390685689).

References

1 Girish S Bhat and Carla D Savage. Balanced gray codes. *the electronic journal of combinatorics*, 3(1):R25, 1996.

2 Carla Binucci, Fabrizio Montecchiani, Daniel Perz, and Alessandra Tappini. Flipping matchings is hard. *arXiv preprint arXiv:2503.02842*, 2025.

3 Håvard Bakke Bjerkevik, Linda Kleist, Torsten Ueckerdt, and Birgit Vogtenhuber. Flipping non-crossing spanning trees. In *Proceedings of the 2025 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 2313–2325. SIAM, 2025.

4 Prosenjit Bose and Ferran Hurtado. Flips in planar graphs. *Computational Geometry*, 42(1):60–80, 2009. URL: `https://www.sciencedirect.com/science/article/pii/S0925772108000370`, `doi:10.1016/j.comgeo.2008.04.001`.

5 Nicolas Bousquet, Lucas De Meyer, Théo Pierron, and Alexandra Wesolek. Reconfiguration of plane trees in convex geometric graphs. In *40th International Symposium on Computational Geometry (SoCG 2024)*, pages 22–1. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2024.

6 Stefan Felsner, Linda Kleist, Torsten Mutze, and Leon Sering. Rainbow cycles in flip graphs. *SIAM Journal on Discrete Mathematics*, 34(1):1–39, 2020.

7 Carmen Hernando, Ferran Hurtado, and Marc Noy. Graphs of non-crossing perfect matchings. *Graphs and Combinatorics*, 18:517–532, 2002.

8 M.C. Hernando, F. Hurtado, A. Márquez, M. Mora, and M. Noy. Geometric tree graphs of points in convex position. *Discrete Applied Mathematics*, 93(1):51–66, 1999. 13th European Workshop on Computational Geometry CG '97. URL: `https://www.sciencedirect.com/science/article/pii/S0166218X99000062`, `doi:10.1016/S0166-218X(99)00006-2`.

9 Ferran Hurtado and Marc Noy. Graph of triangulations of a convex polygon and tree of triangulations. *Computational Geometry*, 13(3):179–188, 1999.

10 Marcel Milich, Torsten Mütze, and Martin Pergel. On flips in planar matchings. *Discrete Applied Mathematics*, 289:427–445, 2021.

11 Lionel Pournin. The diameter of associahedra. *Advances in Mathematics*, 259:13–42, 2014.

12 Carla Savage. A survey of combinatorial gray codes. *SIAM review*, 39(4):605–629, 1997.

13 Daniel Dominic Sleator, Robert Endre Tarjan, and William P Thurston. Rotation distance, triangulations, and hyperbolic geometry. In *Proceedings of the eighteenth annual ACM symposium on Theory of computing*, pages 122–135, 1986.

The Complexity of Extending Storylines with Minimum Local Crossing Number*

Alexander Dobler[1], Siddharth Gupta[2], Philipp Kindermann[3], Fabrizio Montecchiani[4], and Martin Nöllenburg[1]

1 **TU Wien, Austria.** adobler@ac.tuwien.ac.at, noellenburg@ac.tuwien.ac.at
2 **BITS Pilani, K K Birla Goa Campus, India.** siddharthg@goa.bits-pilani.ac.in
3 **Trier University, Germany.** kindermann@uni-trier.de
4 **University of Perugia, Italy.** fabrizio.montecchiani@unipg.it

Abstract

Storyline layouts visualize temporal interactions by drawing each character as an x-monotone curve and enforcing that the participants of every meeting form a contiguous vertical group. We study a drawing extension variant in which a layout of a sub-storyline is fixed and has to be extended by inserting missing characters while preserving all meeting constraints. We minimize the *local crossing number* χ, i.e., the maximum number of crossings along any single character. We prove that the problem is W[1]-hard parameterized by the number k of inserted characters plus the maximum number σ of active characters, in XP parameterized by σ and in FPT parameterized by $\sigma + \chi$.

1 Introduction

Storyline layouts (or *storylines* for short) are visualizations of temporal interaction patterns of a set of actors. They have been initially proposed to show the plots of novels and movies [20], but are more widely applicable for temporal co-authorship networks [1, 14], collaboration in software development [18], or rolling stock scheduling [9]. A storyline depicts time on the x-axis, shows a set of characters as x-monotone curves, and interactions (or *meetings*) as vertical character groupings at the corresponding point in time (see Fig. 1). Combinatorially, the primary degree of freedom in drawing a storyline is to determine valid vertical character orderings, where the characters in each meeting must form a contiguous interval, at all of the discrete meeting times. The predominant optimization goal in storylines is crossing minimization [10, 13, 17], yet other aspects, such as wiggle and whitespace minimization, have also been considered [9, 19, 21]. While most existing approaches target the minimization of the total number of crossings, we are more interested in minimizing the maximum local number of crossings per character. This is motivated by the aim to balance the required crossings (and consequently the visual quality) among the different characters and is related to the established concept of k-planarity and k-plane graph drawings, in which each edge may have at most k crossings [7, 15].

In this paper, we study local crossing minimization in storylines as a *drawing extension problem* [2, 3, 4, 6, 11, 12], i.e., we are given a (partial) storyline layout as well as a set of missing characters. The task is to optimally insert the missing characters into the partial layout with respect to the local crossing number of both the old and the new characters. We call this problem Local StoryLine Extension (LSLE). We take a parameterized complexity perspective and present two main results: Firstly, we show W[1]-hardness of LSLE, parameterized by the number k of new characters and the maximum number σ of

* This work started at the Bertinoro Workshop on Graph Drawing BWGD 2025.

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

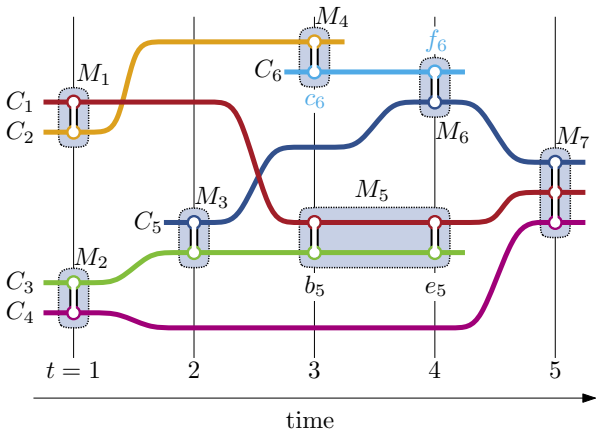


Figure 1 Sketch of a storyline layout with six characters, seven meetings, and five time instants.

characters per time instant, even if there are just two meetings with only new characters. Secondly, we present a dynamic program that yields an XP-algorithm parameterized by σ and an FPT-algorithm parameterized by $\sigma + \chi$.

Preliminaries. A *storyline* instance $\mathcal{S} = (\mathcal{C}, \mathcal{M})$ consists of a set $\mathcal{C} = \{C_1, C_2, \ldots, C_n\}$ of n *characters* and a set $\mathcal{M} = \{M_1, M_2, \ldots, M_m\}$ of m *meetings*. Each meeting $M_i \in \mathcal{M}$ is a triple $\langle \mathcal{C}(M_i), b_i, e_i \rangle$, where $\mathcal{C}(M_i) \subseteq \mathcal{C}$ is a subset of characters, $b_i \in \mathbb{N}$ is the *begin-time* of M_i and $e_i \in \mathbb{N}$ is the *end-time* of M_i (thus, $b_i \leq e_i$). We say that a meeting M_i is *active* at any time t with $b_i \leq t \leq e_i$, and that each character $C_j \in \mathcal{C}(M_i)$ *participates* in M_i. A common assumption is that a character cannot participate in two distinct meetings at the same time instant. For each character $C_j \in \mathcal{C}$ we define its *lifespan* $L_j = [c_j, f_j]$ as the interval between the begin-time b_i of the first meeting M_i that C_j participates in and the end-time e_l of the last meeting M_l it participates in. For any time instant in its lifespan, we say a character is *active*. Let A_t be the set of active characters at time t and let $\sigma_t = |A_t|$ be their number. By convention, we set to 1 the minimum c_j over all characters C_j, and we denote by τ the largest f_j over all characters C_j (i.e., $\mathcal{S}$ is defined over τ time instants).

In a *storyline layout* Γ (see Fig. 1), each character C_j is represented as an x-monotone curve from time instant c_j to time instant f_j. For each time instant $t \in [\tau] = \{1, 2, \ldots, \tau\}$ the active character curves in A_t are ordered vertically as a permutation π_t. Two basic combinatorial properties must be satisfied: (i) For each meeting M_i, the curves representing the characters in $\mathcal{C}(M_i)$ are *consecutive* in the permutations π_j for all time instants $j \in [b_i, e_i]$, i.e., they form a contiguous group between b_i and e_i; (ii) for meetings M_i with a duration $e_i - b_i > 0$ the characters $\mathcal{C}(M_i)$ cannot be crossed while M_i is active. For any two consecutive time instants i and $i+1$, two character curves *cross* if and only if they have a different relative order in π_i and π_{i+1}. The *local crossing number* of a storyline layout is the maximum number of crossings along any single character curve.

We are interested in extension problems. Let $\mathcal{S} = (\mathcal{C}, \mathcal{M})$ be a storyline instance. A *sub-storyline of $\mathcal{S}$* is a storyline instance $\mathcal{S}' = (\mathcal{C}' \subseteq \mathcal{C}, \mathcal{M}')$ such that if a meeting in $\mathcal{S}$ contains at least one character of $\mathcal{S}'$, then such a meeting exists also in $\mathcal{S}'$ (even if incomplete). More formally, for each meeting $M_i = \langle \mathcal{C}(M_i), b_i, e_i \rangle \in \mathcal{M}$ such that $\mathcal{C}(M_i) \cap \mathcal{C}' \neq \emptyset$, there is a meeting $M_i' \in \mathcal{M}'$ such that $M_i' = \langle \mathcal{C}(M_i) \cap \mathcal{C}', b_i, e_i \rangle$. For a storyline layout Γ of $\mathcal{S}$ we denote the induced *sub-storyline layout* $\Gamma[\mathcal{S}']$ as the layout Γ restricted to the characters and meetings in $\mathcal{S}'$. We focus on the following decision problem:

Local StoryLine Extension (LSLE)
Input: A storyline instance $\mathcal{S} = (\mathcal{C}, \mathcal{M})$, an integer χ, a storyline layout Γ' of a sub-storyline $\mathcal{S}' = (\mathcal{C}' \subseteq \mathcal{C}, \mathcal{M}')$ with local crossing number at most χ
Question: Does there exist a storyline layout Γ of $\mathcal{S}$ such that $\Gamma[\mathcal{S}'] = \Gamma'$ with local crossing number at most χ?

We define the following parameters relevant to quantify the complexity of LSLE.

- $n = |\mathcal{C}'|$, number of characters in the sub-storyline $\mathcal{S}'$
- $k = |\mathcal{C} \setminus \mathcal{C}'|$, number of new characters to be inserted
- τ, number of time instants of $\mathcal{S}$
- $\mu = |\mathcal{M} \setminus \mathcal{M}'|$, number of meetings with only new characters in $\mathcal{C} \setminus \mathcal{C}'$
- σ, maximum number of active characters at any time instant

Missing details and full proofs of statements marked by ($\star$) are found in the full version [8] of this abstract.

2 Hardness of Local StoryLine Extension

▶ **Theorem 2.1** ($\star$). *LSLE parameterized by $k + \sigma$ is* W*[1]-hard, even for $\mu = 2$.*

In order to prove Theorem 2.1, we exploit a reduction from the Unary Bin Packing problem. Here, we are given a set $I = \{x_1, x_2, \ldots, x_n\}$ of positive integers encoded in unary, and two integers B and K. The task is to decide whether I can be partitioned into K disjoint subsets (bins) such that the integers in each bin sum up to at most B. This problem has been proven to be W[1]-hard parameterized by the number of bins K [16]. It is not difficult to see that the same W[1]-hardness holds for Exact Unary Bin Packing (EUBP for short), where the sum of the integers in each bin must be exactly B (see, e.g., [5]). A necessary condition for an instance of EUBP to admit a solution is that $\sum_{j=0}^{n} x_j = K \cdot B$.

Overview. The idea is to transform an instance $\langle I, K, B\rangle$ of EUBP into an instance $\langle \mathcal{S}, \chi, \Gamma', \mathcal{S}'\rangle$ of LSLE as follows. The full storyline instance $\mathcal{S}$ and its sub-storyline $\mathcal{S}'$ differ by exactly $k = K$ characters. In order to extend the layout Γ' of $\mathcal{S}'$, the k new characters are forced to go through specific regions in Γ' in which they collect a controlled number of crossings. In particular, some special regions will force the new characters to collect a number of crossings that corresponds to an integer of I. Crucially, the structure of Γ' is such that, if a valid extension Γ exists, then the number of crossings collected by a single character over all special regions is exactly B. Moreover, no two characters can go through the same regions, which implies a partition of the integers (encoded by crossings) into bins (encoded by characters) such that the integers in each bin sum up to precisely B. We now proceed by describing the gadgets used to construct the instance $\langle \mathcal{S}, \chi, \Gamma', \mathcal{S}'\rangle$ of LSLE. These gadgets should be intended as parts of Γ' (and $\mathcal{S}'$) that will be eventually assembled together.

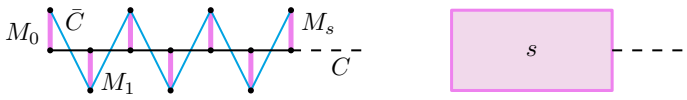


Figure 2 An s-saturator (left) and its schematic representation (right).

Saturator gadgets. An *s-saturator gadget*, for any integer s, consists of two characters C and $\bar{C}$, and $s + 1$ meetings $M_0, M_1, \ldots, M_s$, such that for each $i \in [s]$: $\mathcal{C}(M_i) = \{C, \bar{C}\}$,

and $b_i = e_i = b^* + i$, where b^* denotes the initial time instant when the gadget appears in the storyline. Also, in the corresponding layout in Γ', shown in Fig. 2, C and $\bar{C}$ cross s times. One of the two characters, say C, is called the *leading* character of the gadget, and it will participate in additional meetings beyond the gadget itself, receiving at most $\chi - s$ additional crossings. Instead, the curve representing $\bar{C}$ starts and ends within the gadget representation.

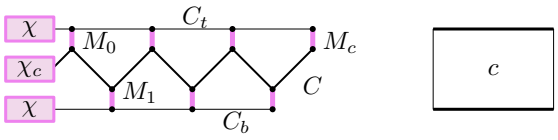


Figure 3 A c-channel (left) and its schematic representation (right).

Channel gadgets. A *c-channel gadget*, for a given integer c called *capacity*, consists of three characters C, C_t, C_b, and $c+1$ meetings $M_0, M_1, \ldots, M_c$, such that for each $i \in [c]$: $\mathcal{C}(M_i) = \{C, C_t\}$ if i is even or $\mathcal{C}(M_i) = \{C, C_b\}$ if i is odd, and $b_i = e_i = b^* + i$, where b^* denotes the initial time instant when the gadget appears in the storyline. We call C the *central character* of the gadget, while C_t and C_b are *boundary characters*. In addition, C_t and C_b are the leading characters of two χ-saturator gadgets, while C is the leading character of a χ_c-saturator gadget, with $\chi_c = \chi - c$; see Fig. 3. Intuitively, the gadget represents a region whose traversal implies crossing c times the central character.

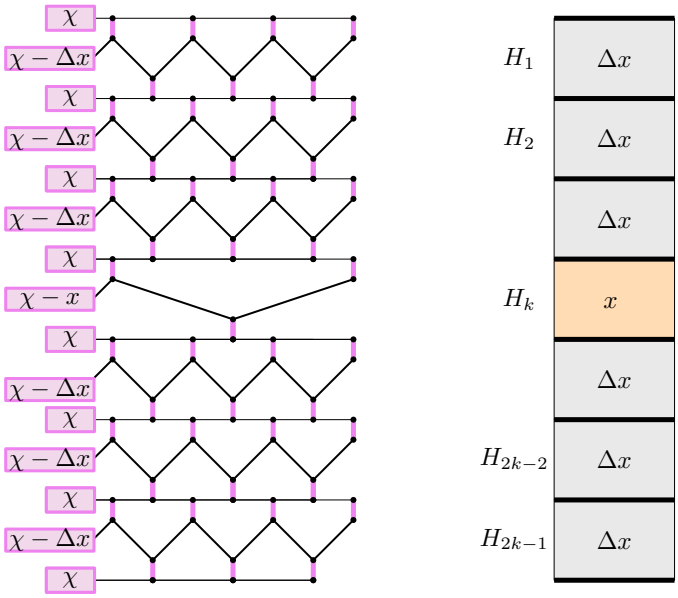


Figure 4 An x-column gadget (left) and its schematic representation (right). The sparse channel has a light orange background, the dense channels are grey.

Column gadgets. An *x-column gadget*, for an integer x, consists of $2k-1$ channel gadgets H_1, H_2, H_{2k-1} such that one boundary character of H_i coincides with one boundary character of H_{i+1}, for $i \in [2k-2]$. Moreover, for each H_i, the capacity is equal to $\Delta \cdot x$ if $i \neq k$, and to x if $i = k$. (We need $\Delta > 1$, the exact value is irrelevant and will be fixed later.) Indeed,

we call the channel H_k the *sparse channel* of the gadget, while all other channels are *dense*. The corresponding layout is shown in Fig. 4. Intuitively, each of the k new characters will be forced to traverse a distinct channel in the column gadget.

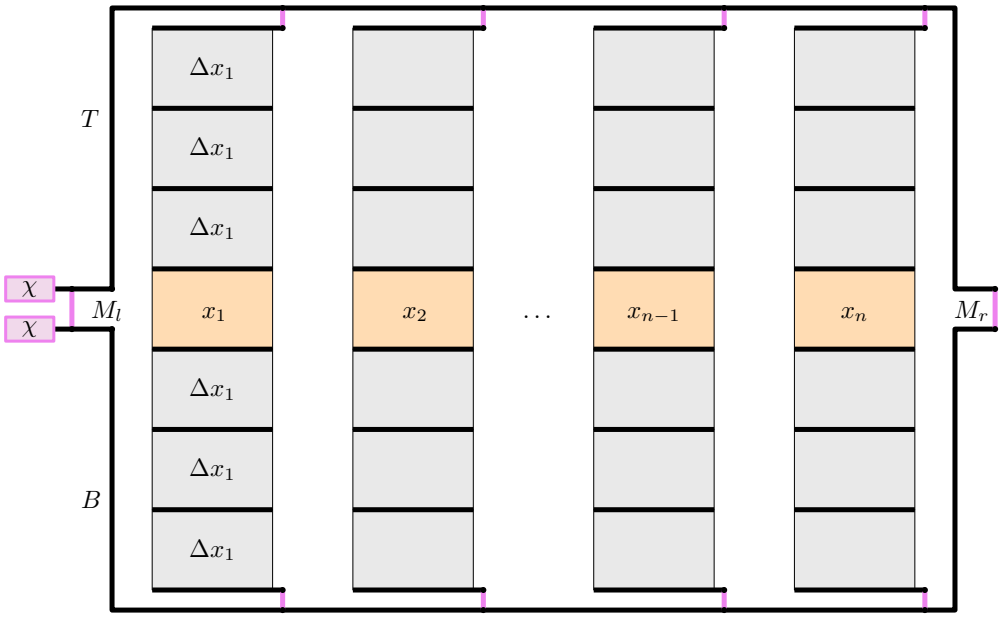


Figure 5 A schematic representation of Γ'.

Putting all together. Consider now n column gadgets $X_1, X_2, \dots, X_n$, such that the capacity of the sparse channel of X_i is equal to the integer $x_i \in I$, for each $i \in [n]$. Let T and B be two further characters that participate in two meetings M_l and M_r, such that: M_l occurs at the first time instant of $\mathcal{S}'$ (except for two saturator gadgets defined later), while M_r occurs at the last time instant; T and B are the only two characters involved in these two meetings. Moreover, T meets with one boundary character of the channel H_1 of the column X_i, for each $i \in [n]$. Analogously, B meets with one boundary character of the channel H_{2k-1} of the column X_i, for each $i \in [n]$. Also, T and B are the leading characters of two χ-saturators. This concludes the construction of Γ' and hence of $\mathcal{S}'$; see also Fig. 5.

Construction of $\mathcal{S}$. It remains to construct $\mathcal{S}$ from $\mathcal{S}'$, which implies introducing k new characters, which we denote by $N_1, N_2, \dots, N_k$, and the corresponding meetings. In particular, these new characters all meet together among them and with both T and B twice: once at the time instant right after the one of M_l, and again at the time instant just before M_r. Finally, we fix the values of Δ and χ as $\Delta = 2$ and $\chi = B + \Delta \cdot B \cdot (K - 1)$.

Idea of the reduction. The gadgets are now arranged in such a way that each new character N_i corresponds to one bin. Each new character must be routed through some sparse and some dense channels and the capacities are designed in such a way that a solution with local crossing number χ exists if and only if the collected sparse channels add up to the bin capacity B.

This concludes the reduction; an example of a solution of LSLE constructed from a solution of EUBP can be seen in Fig. 6.

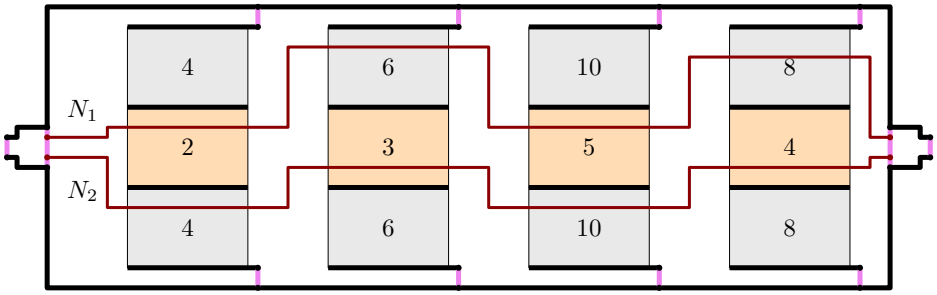


Figure 6 A solution Γ of LSLE constructed from a solution ℓ of EUBP. The instance of EUBP is $I = \{2, 3, 5, 4\}$, $K = 2$, and $B = 7$. The main parameters of Γ are $k = 2$, $\chi = 7 + 2 \cdot 7 = 21$.

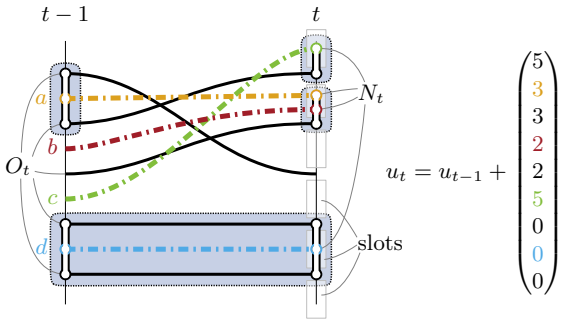


Figure 7 Illustration for the dynamic program. New characters are drawn dash-dotted.

3 An XP- and FPT-algorithm for Local StoryLine Extension

▶ **Theorem 3.1** ($\star$). *LSLE parameterized by σ can be solved in $\tau \cdot (\sigma + \chi)^{O(\sigma)}$ time.*

Let $\langle \mathcal{S}, \chi, \Gamma', \mathcal{S}' \rangle$ be an instance of LSLE. We use dynamic programming.

The DP states. Consider the discrete time instants $t = 1, \ldots, \tau$. Let $O_t = A_t \cap \mathcal{C}'$ be the set of *existing* active characters at time t and let $N_t = A_t \setminus O_t$ be the set of *new* active characters at t. In Γ', the characters in O_t appear in some fixed vertical order π_t, and partition the vertical line at time t into $|O_t| + 1 \leq \sigma_t + 1 \leq \sigma + 1$ *slots*; see Fig. 7.

A *placement* of the new characters at time t must specify which slot each new character occupies, as well as the relative order of the new characters within the same slot. Accordingly, we define a placement P_t as a pair $(s_t, \prec_t)$, where $s_t \colon N_t \to \{0, 1, \ldots, |O_t|\}$ assigns each new character to a slot, and $\prec_t$ is a permutation of N_t that aligns with the slot assignment, i.e., every character in slot i comes before every character in slot $j > i$. From $(s_t, \prec_t)$, we can obtain a single total order ρ_t of A_t for all characters at time t.

We process the time instants from left to right. We define a DP state at time t as a pair (P_t, u_t), where (i) P_t is a placement of the new characters at time t, and (ii) u_t is a vector that assigns to every character $C \in O_t$ the number $u_t(C)$ of crossings involving C that occur strictly before time t. The state (P_t, u_t) is *feasible* if there exists a partial extension of Γ' up to time t that realizes the placement P_t at time t, respects all meeting contiguity constraints for the meetings up to time t, and every character $C \in A_t$ has $u_t(C) \leq \chi$ crossings.

Filling the table. At time $t = 1$, we enumerate all placements P_1 of N_1 that are consistent with the meeting constraints, and set u_1 to the all-zero vector. At the final time $t = \tau$, we accept the instance if there exists any feasible state (P_τ, u_τ).

We now describe how to move from time $t-1$ to time t. Consider a feasible state (P_{t-1}, u_{t-1}). We enumerate all possible placements P_t of the active new characters N_t that (i) satisfy all meeting constraints at time t, and (ii) do not have any character cross an ongoing meeting between time $t-1$ and t that it does not participate in.

For every character $C \in A_{t-1} \cap A_t$, we can compute the number $cr_t(C)$ of crossings involving C that occur in the open strip $(t-1, t)$ from (P_{t-1}, P_t): this is exactly the number of characters whose relative vertical order with C differs between the two total orders.

We define the updated budget vector $u_t(C)$ for every character $C \in A_t$ as follows. If $C \in A_{t-1}$, then $u_t(C) = u_{t-1}(C) + cr_t(C)$; otherwise, the character C starts at time t, and we set $u_t(C) = 0$. If $u_t(C) \leq \chi$ for every $C \in A_t$, then we store (P_t, u_t) as a feasible state. All such transitions generate the set of feasible DP states at time t.

This concludes the sketch of the DP.

References

1 Storyline visualizations with ubiquitous actors. In D. Auber and P. Valtr, editors, *Graph Drawing and Network Visualization (GD'20)*, volume 12590 of *LNCS*, pages 324–332. Springer, 2020.

2 P. Angelini, G. Di Battista, F. Frati, V. Jelínek, J. Kratochvíl, M. Patrignani, and I. Rutter. Testing planarity of partially embedded graphs. *ACM Trans. Algorithms*, 11(4):32:1–32:42, 2015.

3 P. Angelini, I. Rutter, and T. P. Sandhya. Extending partial orthogonal drawings. *J. Graph Algorithms Appl.*, 25(1):581–602, 2021.

4 S. Bhore, R. Ganian, L. Khazaliya, F. Montecchiani, and M. Nöllenburg. Extending orthogonal planar graph drawings is fixed-parameter tractable. In E. W. Chambers and J. Gudmundsson, editors, *Computational Geometry (SoCG'23)*, volume 258 of *LIPIcs*, pages 18:1–18:16. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023.

5 V. Blazej, S. Jana, M. S. Ramanujan, and P. Strulo. On the parameterized complexity of eulerian strong component arc deletion. *Algorithmica*, 87(11):1669–1709, 2025.

6 T. Depian, S. D. Fink, R. Ganian, and M. Nöllenburg. The parameterized complexity of extending stack layouts. In S. Felsner and K. Klein, editors, *Graph Drawing and Network Visualization (GD'24)*, volume 320 of *LIPIcs*, pages 12:1–12:17. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024.

7 W. Didimo, G. Liotta, and F. Montecchiani. A survey on graph drawing beyond planarity. *ACM Computing Surveys*, 52(1):4:1–4:37, 2019.

8 A. Dobler, S. Gupta, P. Kindermann, F. Montecchiani, and M. Nöllenburg. The complexity of extending storylines with minimum local crossing number. *arXiv*, abs/2603.08340, 2026.

9 A. Dobler, T. Hegemann, M. Nöllenburg, and A. Wolff. Optimizing wiggle in storylines. In V. Dujmović and F. Montecchiani, editors, *Graph Drawing and Network Visualization (GD'25)*, volume 357 of *LIPIcs*, pages 39:1–39:17. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2025.

10 A. Dobler, M. Jünger, P. J. Jünger, J. Meffert, P. Mutzel, and M. Nöllenburg. Revisiting ILP models for exact crossing minimization in storyline drawings. In S. Felsner and K. Klein, editors, *Graph Drawing and Network Visualization (GD'24)*, volume 320 of *LIPIcs*, pages 31:1–31:19. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024.

11 E. Eiben, R. Ganian, T. Hamm, F. Klute, and M. Nöllenburg. Extending partial 1-planar drawings. In A. Czumaj, A. Dawar, and E. Merelli, editors, *Automata, Languages, and Programming (ICALP'20)*, volume 168 of *LIPIcs*, pages 43:1–43:19. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2020.

12 R. Ganian, T. Hamm, F. Klute, I. Parada, and B. Vogtenhuber. Crossing-Optimal Extension of Simple Drawings. In N. Bansal, E. Merelli, and J. Worrell, editors, *Automata, Languages, and Programming (ICALP'21)*, volume 198 of *LIPIcs*, pages 72:1–72:17. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021.

13 M. Gronemann, M. Jünger, F. Liers, and F. Mambelli. Crossing minimization in storyline visualization. In Y. Hu and M. Nöllenburg, editors, *Graph Drawing and Network Visualization (GD'16)*, volume 9801 of *LNCS*, pages 367–381. Springer, 2016.

14 T. Hegemann and A. Wolff. Storylines with a protagonist. In S. Felsner and K. Klein, editors, *Graph Drawing and Network Visualization (GD'24)*, volume 320 of *LIPIcs*, pages 26:1–26:22. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2024.

15 S.-H. Hong and T. Tokuyama, editors. *Beyond Planar Graphs*. Springer, 2020.

16 K. Jansen, S. Kratsch, D. Marx, and I. Schlotter. Bin packing with fixed number of bins revisited. *J. Comput. Syst. Sci.*, 79(1):39–49, 2013.

17 I. Kostitsyna, M. Nöllenburg, V. Polishchuk, A. Schulz, and D. Strash. On minimizing crossings in storyline visualizations. In E. Di Giacomo and A. Lubiw, editors, *Graph Drawing and Network Visualization (GD'15)*, volume 9411 of *LNCS*, pages 192–198. Springer, 2015.

18 M. Ogawa and K. Ma. Software evolution storylines. In A. C. Telea, C. Görg, and S. P. Reiss, editors, *Software Visualization (SoftVis'10)*, pages 35–42. ACM, 2010.

19 Y. Tanahashi, C.-H. Hsueh, and K.-L. Ma. An efficient framework for generating storyline visualizations from streaming data. *IEEE Trans. Visualization and Computer Graphics*, 21(6):730–742, 2015.

20 Y. Tanahashi and K.-L. Ma. Design considerations for optimizing storyline visualizations. *IEEE Trans. Visualization and Computer Graphics*, 18(12):2679–2688, 2012.

21 T. Tang, S. Rubab, J. Lai, W. Cui, L. Yu, and Y. Wu. iStoryline: Effective convergence to hand-drawn storylines. *IEEE Trans. Visualization and Computer Graphics*, 25(1):769–778, 2019.

On the edge surplus of 1-planar unit distance graphs over matchstick graphs*

Eliška Červenková[1] and Jan Kratochvíl[2]

1 **Department of Applied Mathematics, Faculty of Mathematics and Physics, Charles University, Prague, Czech Republic**
cervenel@kam.mff.cuni.cz, https://orcid.org/0009-0009-1357-8366

2 **Department of Applied Mathematics, Faculty of Mathematics and Physics, Charles University, Prague, Czech Republic**
honza@kam.mff.cuni.cz, https://orcid.org/0000-0002-2620-6133

Abstract

A matchstick graph is a planar graph drawn in the plane with straight edges of unit length. A natural generalization is the class of 1-planar unit distance graphs, where the planarity constraint is relaxed, and we allow each edge to be crossed by at most one other edge. Let $u_0(n)$ and $u_1(n)$ denote the maximum number of edges in a matchstick graph and a 1-planar unit distance graph on n vertices, respectively. Although the exact value of $u_0(n)$ has been determined, the behavior of $u_1(n)$ remains a subject of active research. Recently, it was shown that $u_1(n)$ strictly exceeds $u_0(n)$ for all $n \geq 16135$. In this paper, we significantly improve this range of validity. Specifically, we prove that $u_1(n) > u_0(n)$ holds for every integer $n \geq 92$.

Related Version Eliška Červenková and Jan Kratochvíl. 1-Planar Unit Distance Graphs with More Edges Than Matchstick Graphs [6].

1 Introduction

Unit distance graphs are graphs that allow plane embedding such that all edges are straight line segments of equal (unit) length. A subclass of particular interest is matchstick graphs, which are unit distance graphs that are also planar (no edges cross). The maximum number of edges of a matchstick graph on n vertices, denoted by $u_0(n)$, was conjectured by Harborth [2], [3], [4] in 1981 and recently proved by Lavollée and Swanepoel [5]:

$$u_0(n) = \lfloor 3n - \sqrt{12n-3} \rfloor.$$

Recently, Gehér and Tóth [1] investigated a natural relaxation of this problem: 1-planar unit distance graphs, where each edge is allowed to cross at most one other edge. Let $u_1(n)$ denote the maximum number of edges in such a graph on n vertices. They asked the question whether this relaxation allows for strictly more edges than in the planar case, i.e., whether $u_1(n) > u_0(n)$.

Previous Work: The first positive answer to this question was presented during EuroCG 2025 [7]. There, the first author constructed specific examples that demonstrate that $u_1(n)$ can strictly exceed $u_0(n)$, although the edge surplus achieved in these cases was small (at most 2). This result was subsequently generalized in [6], where we introduced a different infinite family of graphs. Using that new construction, we demonstrated that

* Both authors gratefully acknowledge support by grant No. 23-04949X of the Czech Science Foundation (GAČR).

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

the surplus is, in fact, unbounded. Two main results were established: Theorem 1 of [6] stated that $u_1(n) > u_0(n)$ is valid for all $n \geq 16135$, and Theorem 2 showed that the surplus $u_1(n) - u_0(n)$ grows asymptotically as $\Omega(n^{1/4})$.

Our Contribution: In this paper, we slightly modify the construction from [6] and exhibit an example that shows that $u_1(17) \geq 37 > 36 = u_0(17)$. This is the least number of vertices for which it is currently known that 1-planar unit distance graphs may have more edges than matchstick graphs. For general n, we refine the analysis of the growth rate of the number of edges in our construction compared to $u_0(n)$, which enables us to significantly extend the continuous range of n for which we can prove that $u_1(n) > u_0(n)$. The main part of the paper is the proof of the following result.

▶ **Theorem 1.1.** *For all $n \geq 92$, $u_1(n) > u_0(n)$.*

The proof is divided into two parts. In Section 2, we follow the proof of Theorem 1 from [6] (for the slightly modified construction), while in Section 3, we present the new refined analysis.

2 Preliminaries

2.1 The construction

In this section, we briefly recall the construction presented in [6], which forms the basis for our new result. The core structure block is the triangular prism graph F, a 1-planar unit distance graph with 6 vertices and 9 edges (see Fig. 1, left).

We define a general graph family $G_{t,k,a,b}$ in a similar way as in Section 2.2 of [6], with two key modifications that allow more flexibility. We arrange copies of F with additonal horizontal edges into a grid and attach paths of triangles (Fig. 1, right) forming trapezoids (Fig. 2, left). Specifically:

- A rectangular core is formed by stacking t rows, where each row consists of $k+1$ concatenated copies of F. In contrast to [6], where half of the rows were vertically reflected (restricting t to even values), we stack only non-reflected rows. It is easy to verify that the number of vertices and edges of the core remain the same with this modification, while t can now be any positive integer.
- A trapezoidal grid of a paths of triangles is attached to the top of the core.
- A reflected trapezoidal grid of b paths of triangles is attached to the bottom. Unlike in [6], where the construction was assumed to be symmetric ($a = b$), we allow a and b to be chosen independently.

The resulting graph $G_{t,k,a,b}$ is a 1-planar unit distance graph. Its size is given by:

$$|V(G_{t,k,a,b})| = ka - \frac{a(a-1)}{2} + 3 + 2k + 3t + 2kt + \frac{b(2k-b+3)}{2}, \tag{1}$$

$$|E(G_{t,k,a,b})| = 3ak + \frac{a-3a^2}{2} + 3 + 4k + 6t + 6kt + 3bk + \frac{9b-3b^2}{2} - b. \tag{2}$$

Finally, we choose a specific dependence of k on a and t, namely, $k = 2a + t + 1$, and set $b = a + 1$. This choice maximizes the quantity $|E(G_{t,k,a,b})| - u_0\left(|V(G_{t,k,a,b})|\right)$. This can be shown by standard methods in calculus. We denote this optimized graph by $G_{t,a}$. Substituting these parameters yields the following:

$$|V(G_{t,a})| = 3a^2 + 9a + 6at + 7 + 2t^2 + 8t, \tag{3}$$

$$|E(G_{t,a})| = 9a^2 + 21a + 18at + 12 + 6t^2 + 19t. \tag{4}$$

For an example, see Fig. 2 right. Here we note that the number of vertices (edges, respectively) of such a graph is the same as the number of vertices (edges, respectively) of a graph $H_{t,a}$ in Proposition 3 in [6]. Therefore, we can use the result from Proposition 4 in [6] and claim that for every $a \geq t^2$, the number of edges in $G_{t,a}$ exceeds the bound for matchstick graphs $u_0(|V(G_{t,a})|)$ by at least t.

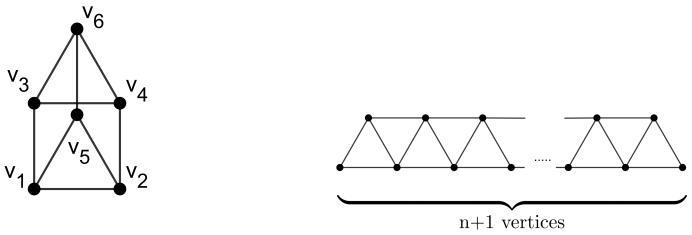


Figure 1 The triangular prism F (left) and a path T_n of triangles (right)

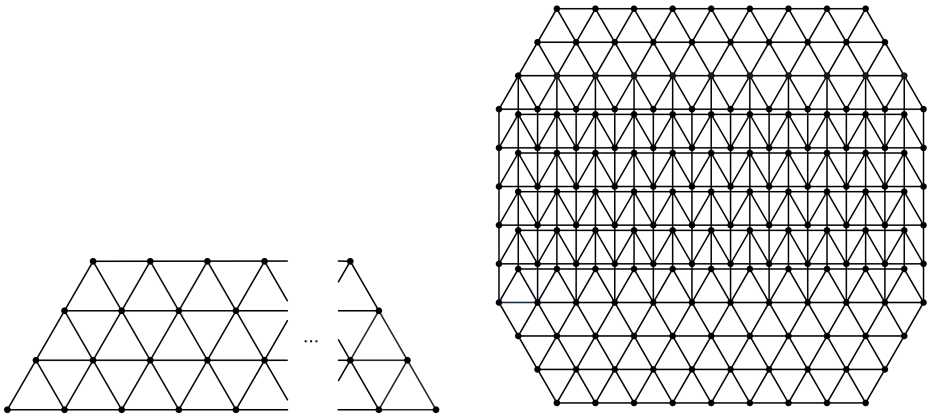


Figure 2 A trapezoid consisting of 3 paths of triangles (left) and a graph $G_{5,10,2,3}$ (right)

2.2 Filling the gaps

The optimized graphs $G_{t,a}$ provide a sequence of values $n(t,a) = |V(G_{t,a})|$ where the edge surplus is guaranteed. To prove the main result for *all* integers n, we must handle the gaps between these values. We first recall the argument from [6, Section 2.2, pp. 7–8], which establishes the surplus for large n. The strategy relies on a constructive transition between two consecutive graphs $G_{t,a}$ and $G_{t,a+1}$. The number of vertices separating the two consecutive graphs is:

$$|V(G_{t,a+1})| - |V(G_{t,a})| = 6a + 6t + 14. \tag{5}$$

Assume that an integer $n \in \{|V(G_{t,a})|, \ldots, |V(G_{t,a+1})|\}$ is given. Construct the graph $L_{t,n}$ by adding the rows attached to $G_{t,a}$ from above and below, until the number of vertices is n (the last row added may remain unfinished). The resulting graph is shown in Fig. 3.

By attaching paths of triangles T_ℓ to the horizontal sides of $G_{t,a}$ (specifically, at most 3 paths on the top and at most 4 on the bottom), we can generate intermediate graphs for all

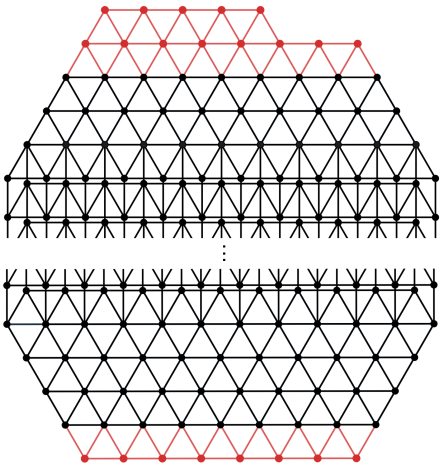


Figure 3 A construction of a graph $L_{t,n}$

n in this range. Each path T_ℓ contributes only one vertex of degree 2 (the rest are of degree 3). Using Lemma 6 from [6], adding these 7 paths of triangles decreases the edge surplus by at most 7. Since $|E(G_{t,a})| - u_0(|V(G_{t,a})|) \geq t$, the intermediate graph $L_{t,n}$ satisfies:

$$|E(L_{t,n})| - u_0(n) \geq t - 7. \tag{6}$$

For $t \geq 8$, the surplus is strictly positive. The smallest parameters that satisfy the conditions of the proof are $t = 8$ and $a = t^2 = 64$, which yields:

$$|V(G_{8,64})| = 16135. \tag{7}$$

Thus, for all $n \geq 16135$, we have $u_1(n) > u_0(n)$.

3 Lowering the threshold

In this section, we extend the validity of our result to a wider range of n. The previous analysis guarantees a surplus only for very large n. To determine the exact lower bound, we need a precise understanding of the matchstick graph upper bound $u_0(n)$. We recall the construction of the matchstick graphs on n vertices, denoted by H_n, that achieve $u_0(n)$ edges defined by Harborth [3]. The graphs H_n are subgraphs of the infinite triangular lattice $\mathcal{T}$.

3.1 Structure of Harborth graphs

The construction of H_n starts with a single central vertex and then grows in hexagonal layers around it. For any $n > 3$, the graph H_n is obtained from H_{n-1} by adding a single vertex v_n and its incident edges in a spiral order. The degree of the new vertex v_n is determined exactly: when initiating a completely new layer, new vertex of degree 2 is placed at the corner. When the spiral moves along a linear side of the inner hexagonal layer, the placement of the previous vertex creates a concave corner with the boundary of H_{n-1}. The new vertex is placed in this concave corner, and because it is connected to the previously added vertex and two vertices of the inner layer, it is of degree 3. When the spiral reaches a corner of the inner hexagonal layer, the new vertex connects to the previously added vertex and only one

vertex of the inner layer, therefore it is of degree 2. This creates the necessary geometry to resume degree 3 additions along the next side of H_{n-1}. The last two steps are repeated until the entire layer is completed, see Fig. 4. This sequence ensures that for any n, H_n achieves the maximum possible number of edges for a matchstick graph, satisfying Harborth's bound:

$$|E(H_n)| = \lfloor 3n - \sqrt{12n-3} \rfloor.$$

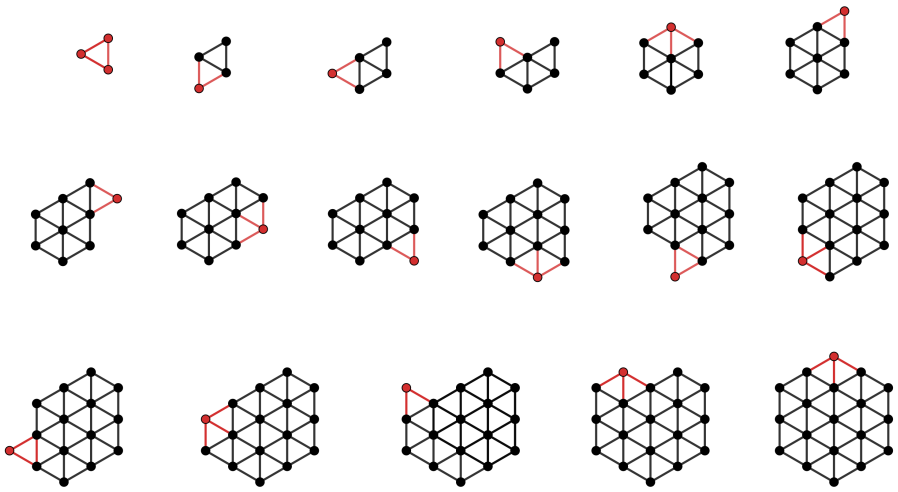


Figure 4 The beginning of Harborth's construction of series of matchstick graphs achieving $\lfloor 3n - \sqrt{12n-3} \rfloor$ edges

The vertex set of H_n can be decomposed into a maximal complete hexagonal core and a partial outer layer. Let s denote the radius (number of fully completed layers) of the core. The number of vertices C_s in a complete regular hexagon of radius s is $C_s = 3s(s+1)+1$. Given n, the radius of the maximal core is $s = \lfloor (\sqrt{12n-3}-3)/6 \rfloor$. The remaining $r = n - C_s$ vertices form the unfinished $(s+1)$-th layer. This layer consists of q fully completed sides and w vertices extending into the next side, where

$$q = \left\lfloor \frac{r}{s+1} \right\rfloor \quad \text{and} \quad w = r \pmod{(s+1)}. \tag{8}$$

3.2 Reduction process

The recursive structure of the Harborth graph H_n allows us to interpret the entire construction as a sequence of degrees $\mathcal{D}_H = (d_4, d_5, \ldots, d_n)$, where $d_i \in \{2, 3\}$ corresponds to the degree of the i-th added vertex.

A similar analysis based on degrees can be applied to our construction of $G_{t,a}$. Let us consider an instance where $|V(G_{8,64})| = 16135$. From Proposition 4 in [6] we know that $|E(G_{8,64})| - |E(H_{16135})| = 8$. To demonstrate that this edge surplus persists for a range of integers smaller than n, we perform a vertex removal process on both graphs:

- For the graph H_n: we remove vertices from H_n in the exact reverse order of their construction. This means that at each step k (from n down to 3), we lose exactly d_k edges.

- For the graph G: we iteratively remove the added paths of triangles. We use a greedy removal strategy: within the structure of these paths, we prioritize the removal of vertices with the lowest possible degree (this means the last vertices of each path).

By carefully tracking the edge counts in the reduction process, which will be described in a forthcoming full version of the paper, we observe that the procedure maintains a surplus of at least 1 down to $t = 5$. Substituting the smallest valid parameters $t = 4$ and $a = 16$ into the vertex formula, we obtain:

$$|V(G_{4,16})| = 3 \cdot 16^2 + 6 \cdot 16 \cdot 4 + 7 + 9 \cdot 16 + 2 \cdot 4^2 + 8 \cdot 4 = 1367. \tag{9}$$

Thus, we conclude that for all $n \geq 1367$, there exists a 1-planar unit distance graph on n vertices with strictly more edges than any matchstick graph with n vertices.

3.3 Computer assisted verification for small n

While the analytical derivation holds for $n \geq 1367$, we verified the existence of the edge surplus for smaller n computationally. We implemented an algorithm that explores the configuration space of the graphs $G_{t,k,k,k+1}$ by iterating through all feasible parameters $t \in [1, 26]$ and $k \in [0, 26]$. For each combination of values of those parameters the procedure goes as follows:

1. We take the graph $G_{t,k,k,k+1}$.
2. (Outer loop) We iteratively remove vertices from the upper paths of triangles one by one, following the degree-based strategy as before (prioritizing vertices of degree 3, then vertices of degree 2). This process continues until all upper paths are removed.
3. (Inner loop) After each vertex removal from the top, we execute a complete removal of the lower triangular section. This means iteratively removing vertices from the lower paths of triangles, one by one, until the lower section is completely removed.

At each step of the inner loop, we calculate the number of vertices $|V| = n$ and the number of edges $|E|$ of the current subgraph. If the condition $|E| > u_0(n)$ is met, we record the result for that specific n. This approach ensures that we test every possible combination of parameters. The code and the details can be found in [8].

The computation confirmed that for every integer n in the interval $[92, 1366]$, there exists a configuration with strictly more edges than the corresponding matchstick graph. With this verification, we can strengthen the previous statement to hold for all $n \geq 92$. Thus, for all $n \geq 92$, there exists a 1-planar unit distance graph on n vertices with more edges than any matchstick graph on the same number of vertices. This concludes the proof of Theorem 1.1.

As a concrete result of this search, we identified a new minimal counterexample on $n = 17$ vertices. This graph, shown in Fig. 5, contains 37 edges, whereas the maximum for a matchstick graph is $u_0(17) = 36$.

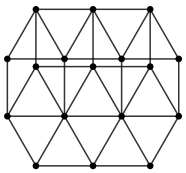


Figure 5 A 1-planar unit distance graph on 17 vertices with more edges than $u_0(17)$

4 Conclusion and open problems

In this paper, we have proved that for every integer $n \geq 92$, the maximum number of edges of a 1-planar unit distance graph strictly exceeds the maximum number of edges of a matchstick graph (i.e., $u_1(n) > u_0(n)$). This result establishes 92 as a sufficient threshold for the inequality, significantly improving the previous bound of $n \geq 16135$. While our computational verification shows that the inequality also holds for several smaller values - the smallest being $n = 17$ - our construction does not yield a surplus for all integers below the threshold. In particular, $n = 91$ is the largest integer for which our current construction still results in a graph with only $u_0(91)$ edges. Notably, our program confirmed that for all $n \geq 10$, our construction matches or exceeds the maximum edge count of matchstick graphs.

Problem 1. Our results identify $n = 17$ as the smallest currently known integer for which $u_1(n) > u_0(n)$. It remains an open question whether this is the absolute minimal counterexample: Does there exist a 1-planar unit distance graph with an edge surplus over matchstick graphs for some $n \leq 16$? Furthermore, it remains to be seen if a surplus can be achieved for all integers in the range $18 \leq n \leq 91$ using a different construction.

References

1 Panna Gehér and Géza Tóth. 1-Planar Unit Distance Graphs. In *32nd International Symposium on Graph Drawing and Network Visualization (GD 2024)*, volume 320 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 6:1–6:9. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024. doi:10.4230/LIPIcs.GD.2024.6.

2 Heiko Harborth. Lösung zu Problem 664A. *Elem. Math.*, 29:14–15, 1974.

3 Heiko Harborth. Point sets with equal numbers of unit-distance neighbors (abstract). In *Discrete Geometry*, Oberwolfach, Tagungsbericht 31/1981, pages 11–12, 1981.

4 Heiko Harborth. Match sticks in the plane. In *Proceedings of the Eugène Strens Memorial Conference on Recreational Mathematics and its History*, The Lighter Side of Mathematics, pages 281–288, Calgary, 1994.

5 Jérémy Lavollée and Konrad Swanepoel. A tight bound for the number of edges of matchstick graphs. *Discrete Computational Geometry*, 72:1530–1544, 2023. doi:10.1007/s00454-023-00530-z.

6 Eliška Červenková and Jan Kratochvíl. 1-Planar Unit Distance Graphs with More Edges Than Matchstick Graphs. In *33rd International Symposium on Graph Drawing and Network Visualization (GD 2025)*, volume 357 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 26:1–26:10. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2025. doi:10.4230/LIPIcs.GD.2025.26.

7 Eliška Červenková. Construction of 1-planar unit distance graphs with more edges than matchstick graphs. In *Abstracts of the 41st European Workshop on Computational Geometry (EuroCG 2025)*, pages 41:1 – 41:8, Liblice, Czech Republic, 2025. Extended Abstract. URL: https://kam.mff.cuni.cz/conferences/eurocg2025/booklet2025.pdf.

8 Eliška Červenková. One-Planar-UD-Graphs-Surplus-Search. https://github.com/cervenel/One-Planar-UD-Graphs-Surplus-Search, 2025. GitHub repository.

Finding Patient Zero via Low-Dimensional Geometric Embeddings

Stefan Huber[1] **and Dominik Kaaser**[2]

1 **Salzburg University of Applied Sciences**
`stefan.huber@fh-salzburg.ac.at`
2 **TU Hamburg**
`dominik.kaaser@tuhh.de`

Abstract

We study the patient zero problem in epidemic spreading processes in the independent cascade model and propose a geometric approach for source reconstruction. Using Johnson-Lindenstrauss projections, we embed the contact network into a low-dimensional Euclidean space and estimate the infection source as the node closest to the center of gravity of infected nodes. Simulations on Erdős-Rényi graphs demonstrate that our estimator achieves meaningful reconstruction accuracy despite operating on compressed observations.

1 Introduction

Epidemic spreading processes are omnipresent in our modern world, and the COVID-19 pandemic has led to a surge in research to model and analyze their fundamental properties. Interestingly, the underlying dynamics of epidemic spreading processes are analyzed (often independently) in seemingly unrelated research areas: In epidemiology, they model the spread of diseases. In sociology, they model opinion formation in social networks. In computer science, they are used, e.g., to update distributed databases [16], model the spread of cyber attacks in large networks [53], and build routing tables in the internet [40]. Surprisingly, less is known about the reverse direction: given the state of an epidemic spreading process, can we find its source "*patient zero*"?

Contribution. In this paper we take a geometric perspective on the patient zero problem. We consider a simple yet expressive model of epidemic spreading based on the prominent SIR model [31, 39, 48] and simulate a simple reconstruction algorithm. In our algorithm we embed the contact network into a geometric space. Our main idea is to estimate patient zero by selecting the node whose embedded position is closest to the center of gravity of all infected nodes. To make this approach scalable, we apply the *Johnson-Lindenstrauss* (JL) lemma [33] to project the network onto a substantially lower-dimensional space while approximately preserving pairwise distances. Our empirical results show that despite significantly reduced dimensions this heavily compressed representation retains sufficient geometric information so that our center-of-gravity estimator achieves accurate reconstruction.

2 Related Work

Global parameters of epidemic spreading processes are well understood, with known results modeling the spread of diseases in epidemiology [8, 36, 39] and the spread of rumors in social networks [5, 17, 18, 19, 26, 28, 37, 38, 41, 44, 45, 49, 54]. They model blog post propagation [42] and fake news [47, 50], marketing strategies [4, 38], attacker lateral movement

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

in corporate networks [53], updates in replicated databases and information spreading [10, 11, 16, 25, 34], or attacks on large enterprise networks and their countermeasures by hardening certain components at an additional cost [3, 23, 27]. Early work on opinion spreading dates back to Downs [21], and there is a vast body of research that models opinion spreading via autonomous agents [2, 7, 15, 22, 32]. For more details we refer to the surveys by Lorenz [43] and Pastor-Satorras et al. [46] and the work by Castellano et al. [9].

Regarding the reverse direction, only a small number of works analyze the patient zero problem in a rigorous way [14, 35, 51, 52, 55, 56]. Shah and Zaman [51, 52] consider infinite acyclic networks and analyze a special case of the SIR model with parameters $\mathrm{SIR}(p, 0)$. They use a fundamental connection between generalized Pólya urns and infection spreading on acyclic networks to prove that approximate inference of patient zero is possible, provided the networks satisfy certain expansion properties. In addition, they bound the distance between the reported patient zero and the true source of the epidemic spreading process. Dawkins et al. [14] introduce a statistical framework to analyze the patient zero problem that covers multiple epidemic spreading models. Zhu et al. [55] and Zhu and Ying [56] consider the more general SIR model and study an inference algorithm for d-regular trees based on Jordan centrality. Berenbrink et al. [6] deepen some of the results from [55, 56] w.r.t. the independent cascade model. They first prove the Bayes optimality of the model and then establish a phase transition for d-regular trees and Galton-Watson trees in terms of the spreading probability β. In addition, the authors bound the probability that the reported patient zero is far away from the true source of the epidemic spreading process. Hahn-Klimroth and Kaaser [29] incorporate a monitoring strategy for the epidemic spreading process. They place sensors on certain edges in the network which raise an alarm when the infection spreads across that edge. The authors prove a lower bound on the required number of sensors, and they simulate heuristics on real-world networks.

In all of these results, complete knowledge of all infected individuals is necessary. This poses a major limitation: in practice, monitoring strategies only deliver incomplete information. Without complete information, multiple works aim to infer an underlying ground truth from a compressed signal. However, to the best of our knowledge, none of these works can be applied directly to the patient zero problem. Prominent examples in this respect are compressed sensing [20] in information theory, the Stochastic Block Model [1] in network science, and Group Testing [12, 13, 24, 30] in information theory and distributed computing.

3 Model

The *independent cascade model* [38] is a simple model of epidemic spreading. It is a special case of the classical SIR model [31, 39, 48], where agents are either *susceptible*, *infected*, or *removed*. Agents are modeled as nodes in a social interaction graph G. Time progresses in discrete rounds. Initially, a single agent—the *patient zero* ω—is *infected*, while all other agents are *susceptible*. In each round t, every infected agent u spreads the infection to each susceptible neighbor v independently with probability β. In case of a success, agent v will be infected at the beginning of round $t + 1$. At the end of each round, every infected agent is removed. Removed agents cannot ever be infected again. The process concludes when either all nodes are removed, or no further infections are possible.

Patient Zero Problem. We are given a graph $G = (V, E)$ and a node $\omega \in V$ chosen uniformly at random from V. Starting from patient zero ω, we run the infection spreading process according to the independent cascade model with infection probability $\beta \in (0, 1)$

Algorithm 1: Patient Zero Scoring via JL-Embedded Distance Signatures

Input: graph $G = (V, E)$, infected set $I \subseteq V$, JL dimension k
Output: score array D

foreach $i \in I$ **do**
  $F[\cdot, i] \leftarrow \mathrm{BFS}(G, i)$ ▷ compute distance signatures $F \in \mathbb{R}^{|I| \times |V|}$ for infected nodes
sample $R \in \mathbb{R}^{k \times |I|}$ with $R_{ij} \sim \mathcal{N}(0, 1/k)$ ▷ sample matrix for JL projection
$X \leftarrow RF$ ▷ compute embedded coordinates
$c \leftarrow \frac{1}{|I|} \sum_{i \in I} X[i, \cdot]$ ▷ compute epidemic center of gravity c
foreach $u \in V$ **do**
  $D(u) \leftarrow \|X[\cdot, u] - c\|_2$ ▷ compute distance of u to c
return D ▷ patient-zero likelihood scores

as a *forward process*. Given a snapshot consisting of the set of infected nodes after an unknown number of t rounds, the task is to infer the identity of patient zero ω. Formally, the forward process on G with source ω defines a distribution over observable outcomes. Let $\mathcal{O}$ denote the observed final infection state. The patient zero problem is to design an estimator $\widehat{\omega} = \widehat{\omega}(G, \beta, \mathcal{O})$ that maximizes the posterior probability $\Pr[\widehat{\omega} = \omega \mid \mathcal{O}]$ or, more generally, minimizes the distance $\mathrm{dist}(\widehat{\omega}, \omega)$ in G.

4 Reconstruction via Geometric Embeddings

In our algorithm we first embed the contact network into a Euclidean space in a way that preserves the relevant structural properties of the graph. To achieve this, we rely on the JL lemma [33] which guarantees low-distortion dimensionality reduction. It states that any set of n points in a high-dimensional Euclidean space can be linearly embedded into a Euclidean space with $\mathrm{O}(\ln n/\varepsilon^2)$ dimensions such that all pairwise distances are preserved up to a factor of $1 \pm \varepsilon$. In practice, this mapping is typically realized using random projections via sparse random matrices $A \in \mathbb{R}^{k \times d}$. The lemma is formalized as follows.

▶ **Lemma 4.1** (Johnson-Lindenstrauss lemma [33]). *Let $X = x_1, \ldots, x_n \in \mathbb{E}^d$ be a set of n points in d-dimensional Euclidean space. There exists a linear function $f \colon \mathbb{R}^d \to \mathbb{R}^k$ with $k \geq 8 \ln(n)/\varepsilon^2$ such that for any pair of points $u, v \in X$*

$$(1 - \varepsilon) \cdot \|u - v\|_2^2 \leq \|f(u) - f(v)\|_2^2 \leq (1 + \varepsilon) \cdot \|u - v\|_2^2 .$$

The spreading process induces a spatial structure on the set of infected nodes. We estimate patient zero by selecting the node whose embedded position is closest to the center of gravity of all infected nodes. To this end, our reconstruction algorithm operates in three stages: extraction of distance signatures, low-distortion dimensionality reduction, and geometric scoring. Our algorithm is formalized in Algorithm 1.

Distance Signatures. For every infected node $i \in I$, the algorithm computes a *distance signature* $F[i, \cdot]$, defined as the array of shortest-path distances from i to all nodes in V. These signatures contain information about the spatial structure of the spread. The first step produces a matrix $F \in \mathbb{R}^{|I| \times |V|}$, where the ith row contains the graph distances from infected node i to all nodes in V.

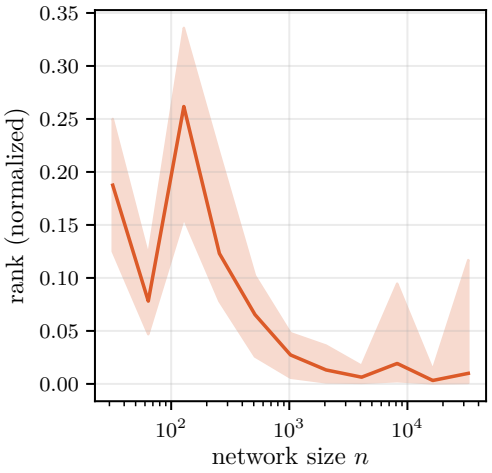


Figure 1 Rank of true patient zero over n.

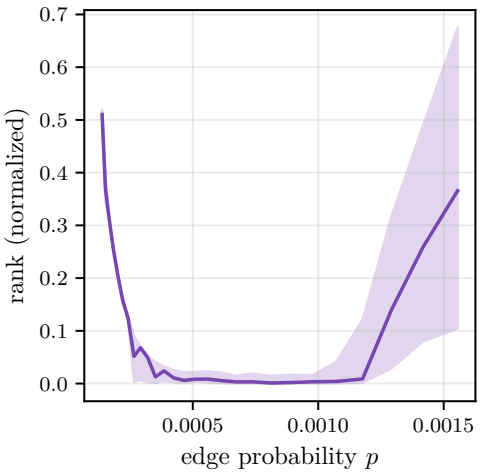


Figure 2 Rank of true patient zero over p.

Geometric Embedding. To obtain a compact geometric representation, the algorithm projects the high-dimensional matrix F onto a k-dimensional Euclidean space using a random Gaussian JL matrix $R \in \mathbb{R}^{k \times |I|}$ with $R_{ij} \sim \mathcal{N}(0, 1/k)$. The embedded coordinates are computed as $X = RF$.

Epidemic Center Estimation. Given the embedded distance signatures $\{X[\cdot, i] \colon i \in I\}$ of all infected nodes, the algorithm computes the center of gravity $c \in \mathbb{R}^k$ by taking their mean.

Patient-Zero Scoring. Finally, the algorithm computes for each node $u \in V$ a score $D(u)$ as the Euclidean distance between the embedded position of node u and the epidemic center of gravity c, that is, $D(u) = \left(\sum_{\ell=1}^{k} \left(X[\ell, u] - c_\ell\right)^2\right)^{1/2}$. Sorting the nodes by D provides a *ranking* of nodes, and the node with rank one is estimated to be patient zero.

5 Evaluation

To evaluate our reconstruction algorithm we implement a custom simulation framework for epidemic spreading on large synthetic networks. The core of our framework is written in Python, with separate modules for network generation, infection propagation, reconstruction analysis, and visualization.

While our results apply to a large number of models of social interactions, we focus in this extended abstract only on Erdős-Rényi random graphs $G(n, p)$. Unless noted otherwise, we use for our evaluation a $G(n, p)$ with $n = 10^4$ nodes and expected node degree $n \cdot p = 10$. We initialize a single known patient zero as infected and then run the epidemic spreading process with transmission probability $\beta = 0.25$. After $t = 4$ rounds we record the infection states of all nodes and apply our reconstruction algorithm to evaluate how well our geometric embedding locates the original source.

First, we show in Figures 1 and 2 the impact of the underlying graph $G(n, p)$ on our reconstruction algorithm. Each data point is based on 100 independent simulation runs. We report the rank of the true patient zero normalized over the graph size n, so values lie in $[0, 1]$ with lower being better. Our data in Figure 1 indicate that the rank improves with

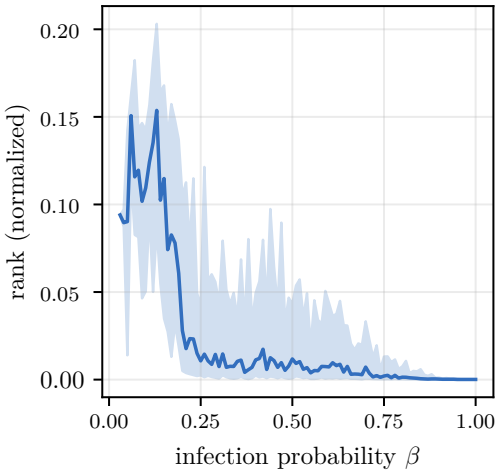


Figure 3 Rank of true patient zero over β.

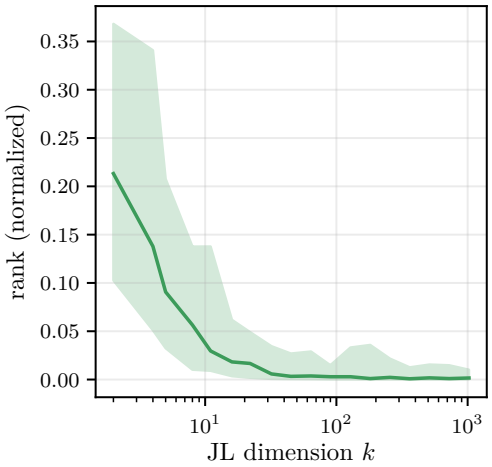


Figure 4 Rank of true patient zero over k.

n, with some variance across multiple runs. In Figure 2 we show the rank over the graph density, spanning subcritical to supercritical connectivity regimes of the $G(n, p)$. For small values of p, outbreaks rarely spread far, yielding little structural information and thus high ranks. Near the critical threshold, a giant component emerges and infection trees retain topology. This enables strong inference, and the normalized rank reaches its minimum. As p continues to increase, the graph becomes dense and diffusion homogenizes. In this regime, the patient-zero signatures blur, and the reconstruction performance worsens again.

Next, we show in Figure 3 the impact of the spreading process on our reconstruction algorithm. Our data confirm that the reconstruction quality depends on the infectiousness of the epidemic: low infection probabilities lead to sparse or quickly extinguished outbreaks, making accurate source inference difficult. Starting with $\beta \approx 0.25$, the epidemic spreading generates enough infected nodes to allow reliable localization of patient zero.

Finally, we show in Figure 4 the impact of the dimension k of the JL embedding. For small values of k, the geometric distortion becomes large, so the center-of-gravity estimator is weak and patient zero ranks poorly. As soon as k approaches $\log n$, the embedding preserves distance signatures faithfully, and the median rank drops. The curve decreases and then plateaus close to zero, indicating that additional dimensions provide diminishing returns.

6 Conclusion

Our geometric approach to the patient zero problem is based on low-dimensional embeddings of epidemic distance signatures. We propose a simple center-of-gravity estimator applied to observations that are heavily compressed via JL projections. Our empirical data indicate that much of the information required for source reconstruction is preserved under strong dimensionality reduction. This suggests that patient zero inference is feasible even when only limited or compressed data is available, a situation that is commonly encountered in real-world epidemic spreading scenarios. In the full version of this paper we will extend our approach to more realistic network models, real-world contact networks, and partially observed infection states.

References

1 E. Abbe. "Community Detection and Stochastic Block Models: Recent Developments." in: *Journal of Machine Learning Research* 18 (2017), 177:1–177:86. URL: https://jmlr.org/papers/v18/16-480.html.

2 R. Abelson and A. Bernstein. "A computer simulation model of community referendum controversies." in: *Public Opinion Quarterly* 27.1 (1963), pp. 93–122. DOI: 10.1086/267152.

3 J. Aspnes, K. L. Chang, and A. Yampolskiy. "Inoculation strategies for victims of viruses and the sum-of-squares partition problem." in: *Journal of Computer and System Sciences* 72.6 (2006), pp. 1077–1093. DOI: 10.1016/j.jcss.2006.02.003.

4 R. Becker, F. Corò, G. D'Angelo, and H. Gilbert. "Balancing Spreads of Influence in a Social Network." in: *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI*. 2020, pp. 3–10. DOI: 10.1609/aaai.v34i01.5327.

5 P. Berenbrink, R. Elsässer, and T. Sauerwald. "Communication Complexity of Quasirandom Rumor Spreading." in: *Algorithmica* 72.2 (2015), pp. 467–492. DOI: 10.1007/s00453-013-9861-5.

6 P. Berenbrink, M. Hahn-Klimroth, D. Kaaser, L. Krieg, and M. Rau. "Inference of a Rumor's Source in the Independent Cascade Model." in: *Thirty-Ninth Conference on Uncertainty in Artificial Intelligence, UAI*. 2023, pp. 152–162. URL: https://proceedings.mlr.press/v216/berenbrink23a.html.

7 P. Berenbrink, M. Hoefer, D. Kaaser, P. Lenzner, M. Rau, and D. Schmand. "Asynchronous Opinion Dynamics in Social Networks." in: *21st International Conference on Autonomous Agents and Multiagent Systems, AAMAS*. 2022, pp. 109–117. URL: https://www.ifaamas.org/Proceedings/aamas2022/pdfs/p109.pdf.

8 F. Brauer. "Mathematical epidemiology: Past, present, and future." in: *Infectious Disease Modelling* 2.2 (2017), pp. 113–127. DOI: 10.1016/j.idm.2017.02.001.

9 C. Castellano, S. Fortunato, and V. Loreto. "Statistical physics of social dynamics." in: *Reviews of Modern Physics* 81.2 (2009), pp. 591–646.

10 F. Chierichetti, G. Giakkoupis, S. Lattanzi, and A. Panconesi. "Rumor Spreading and Conductance." in: *Journal of the ACM* 65.4 (2018), 17:1–17:21. DOI: 10.1145/3173043.

11 F. Chierichetti, S. Lattanzi, and A. Panconesi. "Rumour Spreading and Graph Conductance." in: *Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms, SODA*. 2010, pp. 1657–1663. DOI: 10.1137/1.9781611973075.135.

12 A. Coja-Oghlan, O. Gebhard, M. Hahn-Klimroth, and P. Loick. "Optimal group testing." in: *Combinatorics, Probability and Computing* 30.6 (2021), pp. 811–848. DOI: 10.1017/S096354832100002X.

13 A. Coja-Oghlan, M. Hahn-Klimroth, L. Hintze, D. Kaaser, L. Krieg, M. Rolvien, and O. Scheftelowitsch. "Noisy group testing via spatial coupling." in: *Comb. Probab. Comput.* 34.2 (2025), pp. 210–258. DOI: 10.1017/S0963548324000336.

14 Q. Dawkins, T. Li, and H. Xu. "Diffusion Source Identification on Networks with Statistical Confidence." in: *38th International Conference on Machine Learning, ICML*. 2021, pp. 2500–2509. URL: http://proceedings.mlr.press/v139/dawkins21a.html.

15 M. H. DeGroot. "Reaching a consensus." in: *Journal of the American Statistical Association* 69.345 (1974), pp. 118–121. DOI: 10.1080/01621459.1974.10480137.

16 A. J. Demers, D. H. Greene, C. Hauser, W. Irish, J. Larson, S. Shenker, H. E. Sturgis, D. C. Swinehart, and D. B. Terry. "Epidemic Algorithms for Replicated Database Maintenance." in: *ACM SIGOPS Operating Systems Review* 22.1 (1988), pp. 8–32. DOI: 10.1145/43921.43922.

17 B. Doerr, M. Fouz, and T. Friedrich. "Social Networks Spread Rumors in Sublogarithmic Time." in: *Electronic Notes in Discrete Mathematics* 38 (2011), pp. 303–308. DOI: 10.1016/j.endm.2011.09.050.

18 B. Doerr, M. Fouz, and T. Friedrich. "Why rumors spread so quickly in social networks." in: *Communications of the ACM* 55.6 (2012), pp. 70–75. DOI: 10.1145/2184319.2184338.

19 B. Doerr, T. Friedrich, and T. Sauerwald. "Quasirandom Rumor Spreading." in: *ACM Transactions on Algorithms* 11.2 (2014), 9:1–9:35. DOI: 10.1145/2650185.

20 D. L. Donoho, A. Javanmard, and A. Montanari. "Information-Theoretically Optimal Compressed Sensing via Spatial Coupling and Approximate Message Passing." in: *IEEE Transactions on Information Theory* 59.11 (2013), pp. 7434–7464. DOI: 10.1109/TIT.2013.2274513.

21 A. Downs. *An economic theory of democracy*. Harper and Row, 1957. ISBN: 978-0-060-41750-5.

22 N. E. Friedkin and E. C. Johnsen. "Social influence and opinions." in: *Journal of Mathematical Sociology* 15.3–4 (1990), pp. 193–206. DOI: 10.1080/0022250X.1990.9990069.

23 T. Friedrich, S. Ihde, C. Keßler, P. Lenzner, S. Neubert, and D. Schumann. "Efficient Best Response Computation for Strategic Network Formation Under Attack." in: *Algorithmic Game Theory - 10th International Symposium, SAGT*. 2017, pp. 199–211. DOI: 10.1007/978-3-319-66700-3_16.

24 O. Gebhard, M. Hahn-Klimroth, D. Kaaser, and P. Loick. "Information-theoretic and algorithmic aspects of parallel and distributed reconstruction from pooled data." in: *Journal of Parallel and Distributed Computing* 180 (2023), p. 104718. DOI: 10.1016/j.jpdc.2023.104718.

25 G. Giakkoupis, F. Mallmann-Trenn, and H. Saribekyan. "How to Spread a Rumor: Call Your Neighbors or Take a Walk?" In: *2019 ACM Symposium on Principles of Distributed Computing, PODC*. 2019, pp. 24–33. DOI: 10.1145/3293611.3331622.

26 W. Goffman and V. A. Newill. "Generalization of Epidemic Theory: An Application to the Transmission of Ideas." in: *nature* 204.4955 (1964), pp. 225–228. DOI: 10.1038/204225a0.

27 S. Goyal, S. Jabbari, M. J. Kearns, S. Khanna, and J. Morgenstern. "Strategic Network Formation with Attack and Immunization." in: *Web and Internet Economics – 12th International Conference, WINE*. 2016, pp. 429–443. DOI: 10.1007/978-3-662-54110-4_30.

28 D. Gruhl, D. Liben-Nowell, R. V. Guha, and A. Tomkins. "Information diffusion through blogspace." in: *SIGKDD Explorations* 6.2 (2004), pp. 43–52. DOI: 10.1145/1046456.1046462.

29 M. Hahn-Klimroth and D. Kaaser. "On Reconstructing the Patient Zero from Sensor Measurements." in: *43rd IEEE International Conference on Distributed Computing Systems, ICDCS*. 2023, pp. 671–681. DOI: 10.1109/ICDCS57875.2023.00065.

30 M. Hahn-Klimroth, D. Kaaser, and M. Rau. "Distributed Pooled Data Intrusion Detection: Lessons Learned from Quantitative Group Testing." in: *44th IEEE International Conference on Distributed Computing Systems, ICDCS*. 2024, pp. 198–208. DOI: 10.1109/ICDCS60910.2024.00027.

31 T. Harko, F. S. N. Lobo, and M. K. Mak. "Exact analytical solutions of the Susceptible-Infected-Recovered (SIR) epidemic model and of the SIR model with equal death and birth rates." in: *Applied Mathematics and Computation* 236 (2014), pp. 184–194. DOI: 10.1016/j.amc.2014.03.030.

32 R. Hegselmann and U. Krause. "Opinion dynamics and bounded confidence models, analysis, and simulation." in: *Journal of Artificial Societies and Social Simulation* 5.3 (2002). URL: https://jasss.soc.surrey.ac.uk/5/3/2.html.

33 W. B. Johnson and J. Lindenstrauss. "Extensions of Lipschitz mappings into a Hilbert space." in: *Contemporary Mathematics* 26 (1984), pp. 189–206. DOI: 10.1090/conm/026/737400.

34 R. M. Karp, C. Schindelhauer, S. Shenker, and B. Vöcking. "Randomized Rumor Spreading." in: *41st Annual Symposium on Foundations of Computer Science, FOCS*. 2000, pp. 565–574. DOI: 10.1109/SFCS.2000.892324.

35 S. J. Kazemitabar and A. A. Amini. "Approximate Identification of the Optimal Epidemic Source in Complex Networks." in: *NetSci-X 2020: Sixth International Winter School and Conference on Network Science*. 2020, pp. 107–125. DOI: 10.1007/978-3-030-38965-9_8.

36 M. J. Keeling and P. Rohani. *Modeling Infectious Diseases in Humans and Animals*. Princeton University Press, 2007. ISBN: 978-1-400-84103-5.

37 D. Kempe, J. M. Kleinberg, and É. Tardos. "Influential Nodes in a Diffusion Model for Social Networks." in: *Automata, Languages and Programming, 32nd International Colloquium, ICALP*. 2005, pp. 1127–1138. DOI: 10.1007/11523468_91.

38 D. Kempe, J. M. Kleinberg, and É. Tardos. "Maximizing the Spread of Influence through a Social Network." in: *Theory of Computing* 11 (2015), pp. 105–147. DOI: 10.4086/toc.2015.v011a004.

39 W. O. Kermack and A. G. McKendrick. "A contribution to the mathematical theory of epidemics." in: *Proceedings of the Royal Society of London. Series A* 115.772 (1927), pp. 700–721. DOI: 10.1098/rspa.1927.0118.

40 C. Lenzen and B. Patt-Shamir. "Fast routing table construction using small messages: extended abstract." in: *Symposium on Theory of Computing Conference, STOC*. 2013, pp. 381–390. DOI: 10.1145/2488608.2488656.

41 K. Lerman and R. Ghosh. "Information Contagion: An Empirical Study of the Spread of News on Digg and Twitter Social Networks." in: *Fourth International Conference on Weblogs and Social Media, ICWSM*. 2010, pp. 90–97. DOI: 10.1609/icwsm.v4i1.14021.

42 J. Leskovec, M. McGlohon, C. Faloutsos, N. S. Glance, and M. Hurst. "Patterns of Cascading Behavior in Large Blog Graphs." in: *Seventh SIAM International Conference on Data Mining, SDM*. 2007, pp. 551–556. DOI: 10.1137/1.9781611972771.60.

43 J. Lorenz. "Continuous opinion dynamics under bounded confidence: A survey." in: *International Journal of Modern Physics C* 18.12 (2007), pp. 1819–1838. DOI: 10.1142/S0129183107011789.

44 T. Murayama, S. Wakamiya, E. Aramaki, and R. Kobayashi. "Modeling the spread of fake news on Twitter." in: *PLOS ONE* 16.4 (2021). DOI: 10.1371/journal.pone.0250419.

45 M. Nekovee, Y. Moreno, G. Bianconi, and M. Marsili. "Theory of rumour spreading in complex social networks." in: *Physica A: Statistical Mechanics and its Applications* 374.1 (2007), pp. 457–470. DOI: 10.1016/j.physa.2006.07.017.

46 R. Pastor-Satorras, C. Castellano, P. Van Mieghem, and A. Vespignani. "Epidemic processes in complex networks." in: *Reviews of Modern Physics* 87.3 (2015), pp. 925–979. DOI: 10.1103/RevModPhys.87.925.

47 S. Raponi, Z. Khalifa, G. Oligeri, and R. D. Pietro. "Fake News Propagation: A Review of Epidemic Models, Datasets, and Insights." in: *ACM Transactions on the Web* 16.3 (2022), 12:1–12:34. DOI: 10.1145/3522756.

48 R. Ross. "An application of the theory of probabilities to the study of a priori pathometry.–Part I." in: *Proceedings of the Royal Society of London. Series A* 92.638 (1916), pp. 204–230. DOI: 10.1098/rspa.1916.0007.

49 A. Sadilek, H. A. Kautz, and V. Silenzio. "Modeling Spread of Disease from Social Interactions." in: *Sixth International Conference on Weblogs and Social Media, ICWSM*. 2012, pp. 322–329. DOI: 10.1609/icwsm.v6i1.14235.

50 A. Saxena, P. Saxena, and H. Reddy. "Fake News Propagation and Mitigation Techniques: A Survey." in: *Principles of Social Networking: The New Horizon and Emerging Challenges*. 2022, pp. 355–386. DOI: 10.1007/978-981-16-3398-0_16.

51 D. Shah and T. Zaman. "Detecting sources of computer viruses in networks: theory and experiment." in: *2010 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems, SIGMETRICS*. 2010, pp. 203–214. DOI: 10.1145/1811039.1811063.

52 D. Shah and T. Zaman. "Rumor centrality: a universal source detector." in: *ACM SIGMETRICS/PERFORMANCE Joint International Conference on Measurement and Modeling of Computer Systems, SIGMETRICS*. 2012, pp. 199–210. DOI: 10.1145/2254756.2254782.

53 F. Wilkens, S. Haas, D. Kaaser, P. Kling, and M. Fischer. "Towards Efficient Reconstruction of Attacker Lateral Movement." in: *14th International Conference on Availability, Reliability and Security, ARES*. 2019, 14:1–14:9. DOI: 10.1145/3339252.3339254.

54 J. Zhao, J. Wu, X. Feng, H. Xiong, and K. Xu. "Information propagation in online social networks: a tie-strength perspective." in: *Knowledge and Information Systems* 32.3 (2012), pp. 589–608. DOI: 10.1007/s10115-011-0445-x.

55 K. Zhu, Z. Chen, and L. Ying. "Catch'Em All: Locating Multiple Diffusion Sources in Networks with Partial Observations." in: *Thirty-First AAAI Conference on Artificial Intelligence*. 2017, pp. 1676–1683. DOI: 10.1609/aaai.v31i1.10746.

56 K. Zhu and L. Ying. "Information source detection in networks: Possibility and impossibility results." in: *35th Annual IEEE International Conference on Computer Communications, INFOCOM*. 2016, pp. 1–9. DOI: 10.1109/INFOCOM.2016.7524363.

On the solvability of Shortest Descending Paths*

Víctor Franco-Sánchez[1], Alex Herrero[2], and Rodrigo I. Silveira[3]

1 **Dept. de Matemàtiques, Universitat Politècnica de Catalunya (UPC), Barcelona, Spain. ORCiD: 0009-0007-4163-2476**
2 **Universitat Politècnica de Catalunya (UPC), Barcelona, Spain.**
3 **Dept. de Matemàtiques, Universitat Politècnica de Catalunya (UPC), Barcelona, Spain. ORCiD: 0000-0003-0202-4543**

Abstract

We consider the Shortest Descending Path (SDP) problem: computing a shortest path between two points on a polyhedral terrain that never increases in the z coordinate. No exact polynomial time algorithm is known for this problem. We show that SDPs are not computable within the ACM$\mathbb{Q}$ model, even for terrains with only three triangles. This confirms that the difficulty of the problem is mainly algebraic. We also identify a family of SDPs that can be computed exactly, unifying all previously known solvable terrain classes.

1 Introduction

In the *Shortest Descending Path* (SDP) problem, the goal is to compute a shortest path between two points on a polyhedral terrain such that the path never increases in the z-direction. Such paths are called *descending* paths. A polyhedral terrain is an xy-monotone triangulated surface embedded in $\mathbb{R}^3$.

Descending paths on terrains were first studied by de Berg and van Kreveld [6], who gave a data structure to decide the existence of a descending path between two points in $O(\log n)$ query time after $O(n \log n)$ preprocessing. However, the problem of computing a shortest descending path was left open. Ahmed et al. [1] later proposed approximation algorithms, while Ahmed and Lubiw [2] showed that the main difficulty of the exact problem lies in computing the shortest descending path through a fixed sequence of triangles.

Exact polynomial-time algorithms are known only for restricted terrain classes, namely convex terrains [11], pseudo-convex terrains, and pseudo-orthogonal terrains [2]. For general terrains, only approximation algorithms are available, including discretization methods using Steiner points [1] and approaches adapted from shortest paths on polyhedra [5].

In this paper, we make progress toward understanding the boundary between SDPs that can and cannot be computed exactly. On the negative side, we show that the SDP problem cannot be solved exactly within the *Algebraic Computation Model over the Rational Numbers* (ACM$\mathbb{Q}$), even for terrains with just three triangles. This model, introduced by Bajaj [3], allows computations using field operations and extraction of k-th roots. Our result parallels the known unsolvability of the *Weighted Region Problem* (WRP) in ACM$\mathbb{Q}$ [8, 7]. This is not accidental: Ahmed and Lubiw [2] already observed that the behavior of SDPs closely resembles that of shortest paths in weighted regions. Our result formalizes this connection.

On the positive side, we identify a broad family of SDPs that can be computed exactly. We characterize these paths by their structure: a (possibly empty) flat path from the source, followed by a descending geodesic, and optionally followed by another flat path to the target. We show that all previously known terrain classes admitting exact SDP algorithms (convex,

* Partially supported by grant PID2023-150725NB-I00 funded by MICIU/AEI/10.13039/501100011033.

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

pseudo-convex, and pseudo-orthogonal terrains) produce shortest descending paths of this form. This unifies existing results and clarifies what makes these cases tractable.

2 Unsolvability within ACM$\mathbb{Q}$

Our proof consists of a particular instance to the SDP problem composed of three triangles, forming a terrain, and two points on it, such that computing the SDP from one point to the other requires finding the roots of a certain polynomial that cannot be solved within ACM$\mathbb{Q}$. We defer most technical details to the Appendix. The construction is based on a terrain with five vertices: $A = (-4, 10, 6)$, $O = (0, 0, 0)$, $B = (1, 8, 1)$, $C = (4, 4, 2)$, $D = (4, 2, -4)$, and three triangles: $\Delta OAB, \Delta OBC$ and ΔOCD (Figure 1).

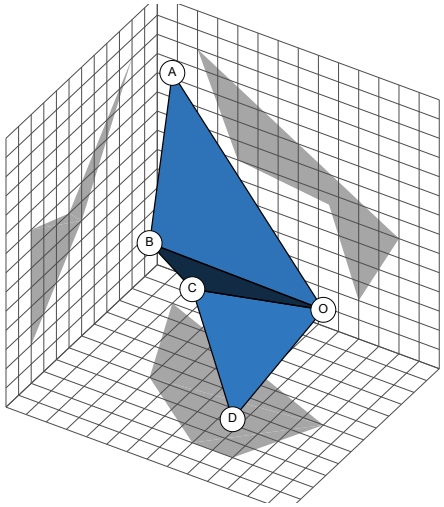


Figure 1 Visual representation of our construction (shadows are parallel projections).

Consider the Shortest Descending Path from A to D. We define two parameters $t_1, t_2 \in [0, 1]$ and two points $P_1 = Bt_1 + O(1 - t_1) = Bt_1$ and $P_2 = Ct_2 + O(1 - t_2) = Ct_2$. Any path from A to D can be expressed as a sequence $A \to P_1 \to P_2 \to D$, for two values t_1, t_2. Consequently, the distance from A to D, parametrized by t_1 and t_2 (Figure 2), is given by:

$$\begin{aligned} d(t_1, t_2) = &\sqrt{(2 - 4t_2)^2 + (4 - 4t_2)^2 + (-2t_2 - 4)^2} + \\ &\sqrt{(-8t_1 + 4t_2)^2 + (-t_1 + 2t_2)^2 + (-t_1 + 4t_2)^2} + \\ &\sqrt{(t_1 - 6)^2 + (t_1 + 4)^2 + (8t_1 - 10)^2} \end{aligned}$$

After algebraic manipulation, the optimal t_1 is the smallest real root of the polynomial:

$$\begin{aligned} &20552697x^8 - 175216932x^7 + 738557316x^6 - \\ &1914201240x^5 + 3347241359x^4 - 3951992296x^3 + \\ &3144512520x^2 - 1805138080x + 436621424 \end{aligned}$$

Next we show that this polynomial is not solvable by radicals. To that end, we will apply the following lemma by De Carufel et al. [8], which presents a criterion to determine when a polynomial is solvable in the ACM$\mathbb{Q}$ model.

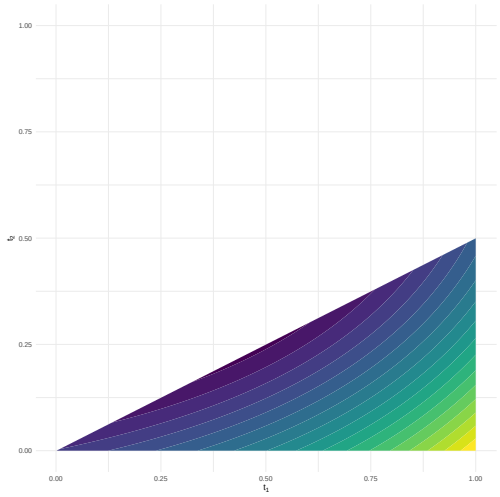


Figure 2 Contour Plot of the distance function with two parameters t_1 and t_2. Darker is lower. The empty region is the region of points that would make the path non-descending, therefore the diagonal of the boundary represents the no-ascending constraint.

▶ **Lemma 2.1** ([8]). *Let $p(x)$ be a polynomial of even degree $d \geq 6$. Suppose there are three prime numbers q_1, q_2 and q_3 that do not divide the discriminant $\Delta(p(x))$ of $p(x)$ such that*

$$\begin{aligned} p(x) &\equiv p_d(x) && \bmod q_1 \\ p(x) &\equiv p_1(x)p_{d-1}(x) && \bmod q_2 \\ p(x) &\equiv p_1'(x)p_2(x)p_{d-3}(x) && \bmod q_3 \end{aligned}$$

where $p_i(x)$ denotes an irreducible polynomial of degree i modulo q_j. Then $p(x) = 0$ is unsolvable within the ACM$\mathbb{Q}$ model.

Taking $q_1 = 67$, $q_2 = 17$, and $q_3 = 19$ satisfies Lemma 2.1, and thus the Shortest Descending Path problem is not solvable within the ACM$\mathbb{Q}$.

▶ **Theorem 2.2** (Unsolvability of SDPs). *The shortest descending path problem cannot be solved exactly within the ACM$\mathbb{Q}$.*

Figure 3 shows a numeric approximation of the SDP, obtained for $t_1 \approx 0.47$ and $t_2 \approx 0.23$.

3 Classes that can be solved exactly

Previous results have shown that there are families of terrains for which the *Shortest Descending Path* (SDP) problem can be solved exactly [2, 10]. For this reason, we explore if our former results provide any new insights on what makes these families solvable, and then we show that the optimal paths in all these cases proven solvable actually share the same property which allows them to always be solved optimally. In this section, we introduce a classification of descending paths based on regular expressions, which allows us to characterize exactly those families of paths that remain solvable within ACM$\mathbb{Q}$.

We restrict our focus to terrains in general position, meaning we exclude the degenerate case where any facet lies parallel to the ground plane.

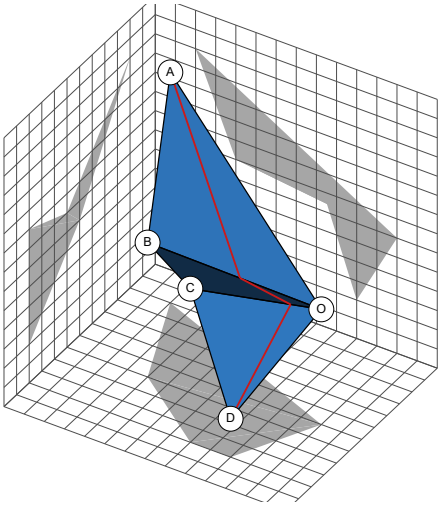


Figure 3 The red path shows the SDP from A to D. Note that the middle segment is level.

3.1 A taxonomy of paths

It has been shown [2] that the main obstacle for solving the SDP lies on finding the shortest descending path from s to t that crosses a given sequence of edges. From now on, consider an edge sequence $\sigma = (e_1, \ldots, e_m)$, where, with some abuse of terminology, $e_1 = s$ and $e_m = t$ are considered zero-length edges.

Given a shortest descending path through a sequence σ, let $\sigma_P = (s = p_1, p_2, \ldots, p_{m-1}, t = p_m)$ be the sequence of intersection points between the path and the edges in σ. Next we define a function to label each edge of a descending path depending on whether the edge is level (i.e., flat), descending, or is a (degenerate) edge consisting of a terrain vertex.

▶ **Definition 3.1.** We associate to each path σ_P a **taxonomy word** $f(\sigma_P)$ over $\{D, L, V\}$, where D denotes a descending segment, L a level segment, and V a passage through a terrain vertex (excluding s and t).

For example, consider the path in Figure 3, with the edge sequence $\sigma = (A, BO, CO, D)$ with points $\sigma_P = (p_1 = A, p_2, p_3, p_4 = D)$. In this case, we have $g(p_1, p_2) = g(p_3, p_4) = D$, $g(p_2, p_3) = L$ and $\nu(p_2) = \nu(p_3) = \varepsilon$. So this path has $f(\sigma_P) = D\varepsilon L\varepsilon D = DLD$.

Here g is a function that classifies pairs of points into descending (D) or level (L), ν returns if the point hits a vertex (V) or not (ε) and ε is the empty word.

▶ **Definition 3.2.** A **taxonomy**, identified with the word $W \in \{D, L, V\}^*$, is the set of all paths with sequence σ on all possible polygonal terrains that have $f(\sigma_P) = W$.

▶ **Definition 3.3.** A taxonomy W is **solvable** if any shortest descending path of type W can be computed exactly within ACM$\mathbb{Q}$.

It will be convenient to identify subpaths of an SDP that are geodesic with a new, specific, symbol in the taxonomy function. To that end, if the sequence σ_P of a shortest descending path contains a maximal (sub)sequence $p_i, \ldots, p_j$ $(i + 1 < j)$ that coincides with a geodesic path from p_i to p_j, we will replace in the taxonomy $f(\sigma_P)$ the substring corresponding to points $p_i, \ldots, p_j$ by a symbol G (that stands for *geodesic*).

▶ Proposition 3.4. Every taxonomy of the form L^*GL^* is solvable.[1]

Proof. Suppose the optimal path is of the form L^*GL^*, starting at vertex s and ending at vertex t, where G is a geodesic path of the terrain in between two points.
The endpoints of the geodesic must lie on terrain edges at the same height as s and t, yielding finitely many candidates whose coordinates are computable in closed form. Since geodesic paths can be computed in polynomial time (see, e.g., [4]), the proposition follows. ◀

Moreover, when an SDP passes through a vertex of the polyhedral terrain, this does not add any difficulty to the solvability of the path, since that means that we can split the path into two subpaths: one from s to the vertex, and another from the vertex to t and then join both paths on such vertex. The union does not violate the descending constraint.

▶ **Corollary 3.5.** *Let $\sigma_P = (s, \dots, p_i,)$ and $\sigma_{P'} = (p_i, \dots, t)$ be two different disjoint descending paths. If $f(\sigma_P)$ is solvable and $f(\sigma_{P'})$ is solvable, then the sequence $f(\sigma_{P''}) = f((s, \dots, p_i, \dots, t))$ is solvable.*

The two previous results leads to the following corollary:

▶ **Corollary 3.6.** *Every taxonomy of the form $(L^*GL^*V)^*L^*GL^*$ is solvable.*

Note that this taxonomy does not capture every solvable terrain, but we conjecture that for every taxonomy not captured by that regular expression, the density of solvable terrains in the space of all possible terrains is zero.

3.2 Cases for which the SDP can be computed exactly

Several classes of terrains for which the shortest descending path (SDP) problem is solvable exactly in polynomial time have been studied in the literature. In this section, we show that these results can be uniformly interpreted within our taxonomy of path types.

▶ **Theorem 3.7** (Convex terrains). *In a convex terrain, any shortest descending path consists of a sequence of level paths followed by a geodesic, and therefore belongs to the class L^*G.*

Proof. Roy et al. [9] prove in their Lemma 4.1 that, in a convex terrain, an SDP is composed of a sequence of horizontal paths followed by a geodesic segment. Since horizontal paths correspond to level edges in our taxonomy, this structure is directly captured by the regular expression L^*G. ◀

The work of Ahmed et al. [2] extends the exact solvability of the SDP problem to two additional classes of terrains: pseudo-convex and pseudo-orthogonal terrains. Pseudo-orthogonal terrains are those which only have two kinds of facets, perfectly vertical and perfectly flat, and pseudo-convex terrains are those for which every non-convex edge is either perfectly vertical or horizontal (note that for these to be terrains, one has to relax the xy-monotonicity condition). We next show that both cases can also be expressed within the same taxonomy.

We should note that Ahmed et al. [2] use a different terminology to describe the structure of shortest descending paths. They call a segment *free* if both of its endpoints can be perturbed arbitrarily slightly while preserving the descending property; otherwise, they call it *constrained*. In what follows, we relate these notions to our taxonomy.

[1] The star (*) is shorthand for the Kleene Star; L^* means "zero or more repetitions of L".

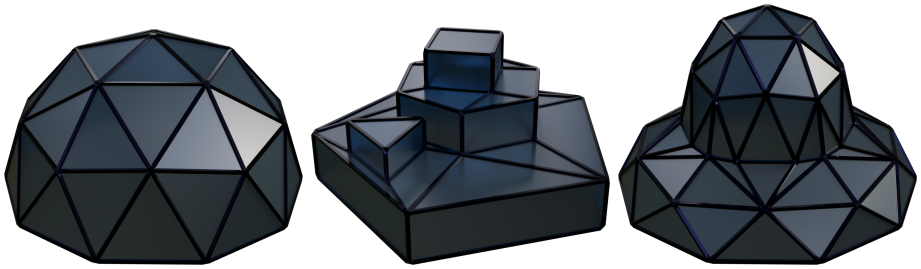


Figure 4 The different known solvable terrain types. From left to right: Convex, Pseudo-Orthogonal and Pseudo-Convex

▶ **Lemma 3.8** (Terminology correspondence). *Let P be a shortest descending path on a terrain.*

1. *Every constrained segment of P is a level (L) segment.*
2. *Any maximal sequence of free segments of P coincides with a geodesic between its endpoints.*

Proof. By definition, a constrained segment cannot be locally perturbed while remaining descending, which occurs precisely when the segment lies on a level set of the height function; hence, constrained segments are level.

Conversely, a free segment admits local perturbations while preserving descent, implying that it is locally shortest within the sequence of triangles it traverses. If a maximal sequence of free segments did not coincide with a geodesic, it could be locally shortened while maintaining the descending property, contradicting the optimality of P. Therefore, any maximal sequence of free segments forms a geodesic. ◀

▶ **Theorem 3.9** (Pseudo-convex and pseudo-orthogonal terrains). *In pseudo-convex terrains, any shortest descending path belongs to the class L^*G. In pseudo-orthogonal terrains, any shortest descending path belongs to the class G.*

Proof. Lemma 14 of [2] states that, in pseudo-convex terrains, an SDP consists of a sequence of constrained segments followed by free segments. By Lemma 3.8, constrained segments are level, and the maximal sequence of free segments coincides with a geodesic, yielding a path of type L^*G. Similarly, Lemma 15 of [2] shows that, in pseudo-orthogonal terrains, an SDP consists either entirely of constrained segments or entirely of free segments. In the first case, the path is of type L^*; in the second, it is a geodesic. Since a level-only path is a degenerate geodesic in this setting, both cases fall under the class G. ◀

4 Conclusions

We analyzed the SDP problem and showed that, even for very simple constructions with only three triangles, the SDP cannot be computed exactly within ACM$\mathbb{Q}$. While the general computational complexity of the Shortest Descending Path problem remains unknown, our results indicate that this is, at least partially, due to algebraic difficulties.

In contrast, we have identified the structure of SDPs that can be computed exactly, and shown that the three types of terrains for which exact algorithms were already known fall under this umbrella. Future work includes identifying additional solvable terrain classes and designing more efficient general algorithms to compute SDPs with this particular structure.

References

1 Mustaq Ahmed, Sandip Das, Sachin Lodha, Anna Lubiw, Anil Maheshwari, and Sasanka Roy. Approximation algorithms for shortest descending paths in terrains. *Journal of Discrete Algorithms*, 8(2):214–230, 2010.

2 Mustaq Ahmed and Anna Lubiw. Shortest descending paths: Towards an exact algorithm. *International Journal of Computational Geometry & Applications*, 21(04):431–466, 2011.

3 Chanderjit Bajaj. The algebraic degree of geometric optimization problems. *Discrete & Computational Geometry*, 3:177–191, 1988.

4 Jindong Chen and Yijie Han. Shortest paths on a polyhedron. In *Proceedings of the sixth annual symposium on Computational geometry*, pages 360–369, 1990.

5 Siu-Wing Cheng and Jiongxin Jin. Approximate shortest descending paths. *SIAM Journal on Computing*, 43(2):410–428, 2014.

6 Mark De Berg and Marc van Kreveld. Trekking in the alps without freezing or getting tired. *Algorithmica*, 18(3):306–323, 1997.

7 Sarita de Berg, Guillermo Esteban, Rodrigo I. Silveira, and Frank Staals. Exact solutions to the weighted region problem. 2025. URL: `https://arxiv.org/abs/2402.12028`, `arXiv:2402.12028`.

8 Jean-Lou De Carufel, Carsten Grimm, Anil Maheshwari, Megan Owen, and Michiel Smid. A note on the unsolvability of the weighted region shortest path problem. *Computational Geometry*, 47(7):724–727, 2014.

9 Sasanka Roy. Near optimal algorithm for the shortest descending path on the surface of a convex terrain. *Journal of Discrete Algorithms*, 15:63–70, 2012. URL: `https://www.sciencedirect.com/science/article/pii/S1570866712000366`, `doi:10.1016/j.jda.2012.03.001`.

10 Sasanka Roy, Sandip Das, and Subhas C Nandy. Shortest monotone descent path problem in polyhedral terrain. *Computational Geometry*, 37(2):115–133, 2007.

11 Xiangzhi Wei and Ajay Joneja. An improved algorithm for the shortest descending path on a convex terrain. *Journal of Discrete Algorithms*, 19:52–56, 2013.

An FPT Algorithm for Maximum $k \times k$ Square Packing Parameterized by Remaining Space

Maarten Dankers, Thomas C. van Dijk, and Kevin Verbeek

Eindhoven University of Technology
{m.dankers|t.c.v.dijk|k.a.b.verbeek}@tue.nl

Abstract

We study the complexity of Maximum $k \times k$ Square Packing, the problem of packing a maximum number of axis aligned $k \times k$ squares into a rectilinear polygon with n vertices on integer coordinates. We present a branching algorithm that runs in $O({c_k}^h \cdot \text{poly}(n))$ time, where h is the amount of remaining space (unit squares not covered) in an optimal solution, and c_k is a constant with $c_2 = 2$ converging to 1 as k goes to infinity. This runtime is fixed parameter tractable (FPT) in h and, as we show, in the length of the boundary of the polygon. Furthermore, we analyze a straightforward integer linear programming (ILP) formulation of the problem and show that our branching rule can be implemented in an ILP solver as a natural variable selection policy, and the reduction rules can be forced to hold in the fractional relaxation. This enables an ILP solver-based solution that runs in FPT time by bounding the maximum branching depth.

Figure 1 An example of a maximum 2-by-2 square packing on an image of iconic Dutch houses. The input polygon[1] has 1450 vertices, boundary length 13,320 and area 50,349. This optimal packing covers 46,736 area using 11,684 squares; 3613 cells remain uncovered.

1 Introduction

Given a polygon P, the Maximum $k \times k$ Square Packing problem asks for the maximum number of axis-aligned, disjoint squares of size k by k that are all contained in P. For $k \geq 2$, this problem is known to be NP-hard [5], even if P is a simple, rectilinear and orthogonally

[1] Design by Freepik

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

convex polygon with integer coordinates [1]. Therefore, the existing literature on this problem focuses on either approximation algorithms or exact algorithms for specific cases. Hochbaum and Maass [6] present a PTAS with $O(k^2 A^{-\varepsilon^2}/\varepsilon^2)$ running time and approximation ratio $(1+\varepsilon)^2$, for any $\varepsilon > 0$, where A is the area of P. Chan [3] considers the independent set problem on a set of candidate rectangles and presents a PTAS with approximation ratio $(1+\varepsilon)$ that runs in $O(\bar{A}\log(\bar{A}) + \bar{A}\Delta^{\frac{1}{\epsilon}-1})$ time, where $\bar{A}$ denotes the number of rectangles and Δ is the maximum number of rectangles that overlap in a single interior point (in our setting $\Delta = k^2$). Exact polynomial time algorithms for the Maximum $k \times k$ Square Packing problem are known for staircase, pyramid, and Manhattan skyline polygons [4], and for $k = 2$ there exist exact algorithms for larger composite polygon classes [7].

El-Khechen [4] presents an integer linear programming (ILP) formulation for this problem, which uses a number of variables proportional to the area of the polygon. However, the area of the input polygon does not seem to capture the complexity of a problem instance well. Consider packing a large axis-aligned rectangle. Optimally packing this is trivial, yet the ILP formulation uses a large number of variables. Thus, such ILP formulations do not directly offer much insight in how efficiently an instance can be solved. Instead of the area, we believe that the number of vertices of P, or the boundary length of P, should be much more indicative of the complexity of an instance.

In this paper, we restrict ourselves to input polygons P that are rectilinear and where each vertex has integer coordinates. Such polygons are also known as *polyominoes*. For this setting, we present an FPT algorithm for Maximum $k \times k$ Square Packing that runs in $O({c_k}^h \cdot \text{poly}(n))$ time, where n is the number of vertices, h is the number of unit squares in P that are not covered in an optimal packing, and c_k is a constant. Here, $c_2 = 2$ and c_k converges to 1 as k goes to infinity. We show that h is linearly proportional to the perimeter of P (for fixed k). Furthermore, we show that an ILP solver can be adapted to use the same branching structure used by our FPT algorithm, leading to the same worst case running time as this algorithm.

2 Preliminaries

As stated above, our input consists of a polyomino P. We say that a polyomino is *simple* if it has no holes. We denote the boundary of P as ∂P; this boundary consists of a set of cycles of axis-aligned line segments, where there is exactly one cycle if P is simple. Let $A(P)$ be the area of P and $L(P)$ be the length of ∂P. If P is clear from context, we simply write A and L for the area and perimeter, respectively. We further denote the length of a single line segment ℓ as $L(\ell)$. We say that v is a *convex* vertex if the two incident line segments bound an interior angle of 90 degrees, and a *concave* vertex if they bound 270 degrees.

Let $S_k(a,b)$ be the $k \times k$ square defined by the region $[a, a+k] \times [b, b+k]$, where k is a positive integer. If k is fixed, then we can simply represent a square $S_k(a,b)$ by its lower left corner (a,b). Hence, a set of $k \times k$ squares $\mathcal{S}_k$ can be represented as a subset of $\mathbb{R}^2$. We say that a set of $k \times k$ squares $\mathcal{S}_k$ is a packing of a polyomino P if the following conditions hold:

1. For every $(a,b) \in \mathcal{S}_k$ we have that $S_k(a,b) \subseteq P$;
2. For every two distinct $(a,b),(c,d) \in \mathcal{S}_k$, $S_k(a,b)$ and $S_k(c,d)$ are interior disjoint.

Due to the following lemma, we can restrict ourselves to squares on integer coordinates.

▶ **Lemma 1** (Baur and Fekete [2]). *For any polyomino P there always exists a maximum $k \times k$ square packing $\mathcal{S}_k$ such that $\mathcal{S}_k \subseteq \mathbb{Z}^2$.*

We call a unit square $S_1(x,y)$ a *cell* of P if $(x,y) \in \mathbb{Z}^2$ and $S_1(x,y) \subseteq P$. A cell is *covered* by a packing $\mathcal{S}_k$ if there exists $(a,b) \in \mathcal{S}_k$ such that $S_k(a,b)$ contains the cell; otherwise

it is called *uncovered*. Note that maximizing the cardinality of a $k \times k$ square packing is equivalent to minimizing the number of uncovered cells. For any polyomino P we will use $h_k(P)$ to denote the number of uncovered cells in a maximum $k \times k$ square packing. Our FPT algorithm is parameterized by h, the number of uncovered cells allowed in the solution; for use as an exact algorithm, initially set $h \geq h_k(P)$, for which we provide a useful bound in Lemma 2.

The following two reduction rules are commonly used in the literature on square packing problems [4, 7]. The first is similar to the corner placement used by El-Khechen for their staircase polygon packing [4].

▶ **Reduction Rule 1.** *Let v be a convex vertex in a polyomino P whose incident line segments ℓ_1, ℓ_2 both have length at least k. Place the unique square $S_k(a, b)$ that touches v.*

Correctness proof. Assume we have some maximum $k \times k$ square packing $\mathcal{S}_k$ on P and for the unique square $S_k(a, b)$ in P that touches v, assume that $(a, b) \notin \mathcal{S}_k$. Then either no square in the packing overlaps $S_k(a, b)$ at all, which contradicts maximality of $\mathcal{S}_k$, or there is exactly one square $S_k(c, d)$ that overlaps it (since both $L(\ell_1)$ and $L(\ell_2)$ are at least k). In the latter case, we can replace $S_k(c, d)$ with $S_k(a, b)$. ◀

▶ **Reduction Rule 2.** *Let $S_1(x, y)$ be a cell in a polyomino P. If there is no $k \times k$ square $S_k(a, b) \subseteq P$ that can cover $S_1(x, y)$, then remove $S_1(x, y)$ from P; this reduces $h_k(P)$ by 1.*

Reduction rules 1 and 2 can be exhaustively applied to reduce a polyomino to a state where each convex vertex is incident to at least one line segment whose length is less than k and where every cell in P can be covered by a $k \times k$ square contained in P. We call such a polyomino *reduced*. Note that the reduction rules may disconnect a polyomino, but the resulting components can be solved separately; a simple polyomino remains simple.

3 FPT Algorithm

We now give an FPT algorithm for the Maximum $k \times k$ Square Packing problem parameterized by h, the number of uncovered cells allowed in the solution. First we show that this parameter can in fact be bounded by the perimeter of P for fixed k.

▶ **Lemma 2.** *For any polyomino P, $h_k(P) \leq k^2 L$. If P is simple, then $h_k(P) \leq kL$.*

Proof sketch. Consider a tiling of the plane using $k \times k$ squares and place those that lie fully within P. Then all uncovered cells are within (Chebyshev) distance k of the boundary as illustrated in the example given by Figure 2. If P is simple, then this results in a bound of kL for $h_k(P)$, as the set of cells inside P at a fixed distance x from the boundary is at most of size L, for any distance x between 1 and k. If P is not simple then each hole in the polyomino may add $O(k^2)$ extra uncovered cells, for a bound of $k^2 L$ in the worst case. ◀

For this reason, a worst case running time depending on h is interesting, as Lemma 2 shows us that picking $h = kL$ gives us $h_k(P) \leq h$ and $h = O(L)$ in the worst case for fixed k, which can be much lower than A as $A = \Theta(L^2)$ in the worst case. Thus, an FPT algorithm with parameter h can offer much better worst case guarantees than an algorithm with $\Theta(A)$ decision variables as the ILP formulation of El-Khechen [4].

Our FPT algorithm uses reduction rules 1 and 2 to reach a state where a good branch is always possible, based on a particular configuration that is always present in reduced instances (see Figure 3).

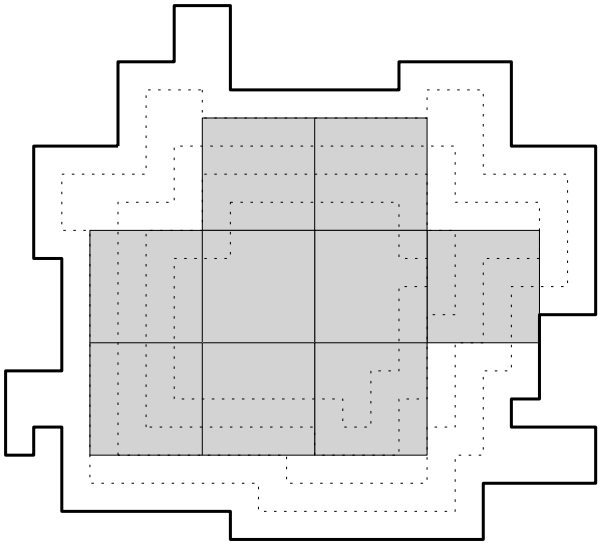


Figure 2 A simple polyomino P (solid), $k = 4$ layers of cells touching the previous boundary (dotted), and squares from an arbitrary regular tiling that are contained in P (gray).

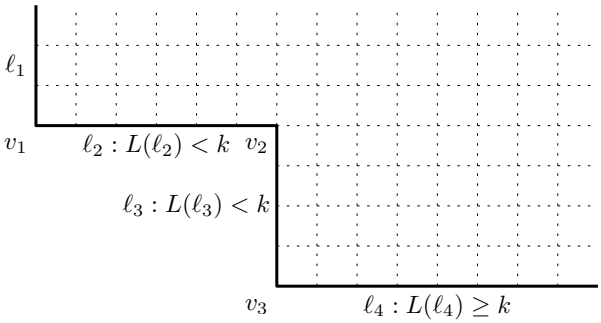


Figure 3 By Lemma 3, this configuration exists somewhere in any reduced polyomino.

▶ **Lemma 3.** *For any reduced polyomino P there exists a sequence of segments and vertices $\ell_1, v_1, \ell_2, v_2, \ell_3, v_3, \ell_4$ on ∂P such that v_1 and v_3 are convex, v_2 is concave, $L(\ell_2) < k$, $L(\ell_3) < k$ and $L(\ell_4) \geq k$.*

Proof sketch. A horizontal segment ℓ who is locally minimal (so both vertical connecting segments go up) must satisfy $L(\ell) \geq k$ by reduction rule 2. Of all such segments pick the leftmost one. As $L(\ell_3) < k$ by reduction rule 1 we either directly find the desired configuration with $\ell_4 = \ell$, or the segment ℓ_2 in this case has $L(\ell_2) \geq k$. Picking this as our next candidate for ℓ_4 we can repeat this argument until we must eventually find the desired configuration. Do note that we can never have a vertical segment on the left side that goes down as this will lead to a locally minimal segment to the left of ℓ. ◀

We can use the special configuration of Lemma 3 to define two suitable branching rules for our algorithm. We decide what branching rule to use based on the length of ℓ_2, which will guarantee us that at least $\lceil \frac{k}{2} \rceil$ uncovered cells are created when placing a square, reducing h with just as much.

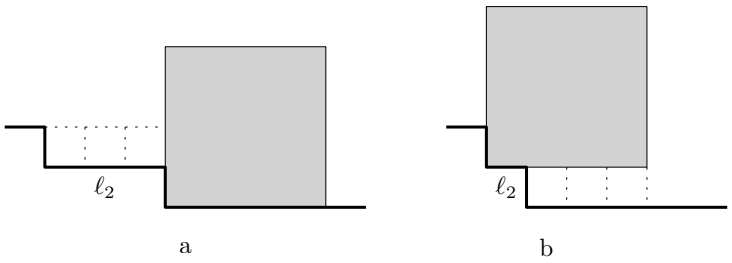


Figure 4 The square placement of option 2 for the branching rules 1, seen in (a) where $L(\ell_2) \geq \lceil \frac{k}{2} \rceil$, and 2, seen in (b) where $L(\ell_2) < \lceil \frac{k}{2} \rceil$.

▶ **Branching Rule 1.** *If $L(\ell_2) \geq \lceil \frac{k}{2} \rceil$ branch on the following two options.*

1. *Remove the unique cell touching v_3 and decrease h by 1.*
2. *Place the unique $k \times k$ square touching v_3 and remove this area from P.*

▶ **Branching Rule 2.** *If $L(\ell_2) < \lceil \frac{k}{2} \rceil$ branch on the following two options.*

1. *Remove the unique cell touching v_1 and decrease h by 1.*
2. *Place the unique $k \times k$ square touching v_1 and remove this area from P.*

For the branching rules above it is trivial to see that, in the convex vertices v_1 and v_3, all possible $k \times k$ square packings either have an uncovered cell touching the vertex or a square touching the vertex. Importantly, these branching rules create necessarily-uncovered cells, reducing the parameter in a branching algorithm (see Figure 4).

▶ **Lemma 4.** *Given a reduced polyomino P, for either of the branching rules, let h_1 and h_2 be the parameters obtained by branching on option 1 or 2, followed by exhaustively applying the reduction rules, respectively. Then $h_1 \leq h - 1$ and $h_2 \leq h - \lceil \frac{k}{2} \rceil$.*

Proof sketch. When removing a cell in the corner in option 1, we set it as uncovered, so $h_1 \leq h - 1$. If we place a square in the corner as in option 2, then either a line of uncovered cells of length $L(\ell_2)$ gets created, or one of length $k - L(\ell_2)$ by branching rules 1 and 2 respectively, for a lower bound of $\lceil \frac{k}{2} \rceil$ on the better option. Thus $h_2 \leq h - \lceil \frac{k}{2} \rceil$. ◀

By Lemma 4, we see that for any branching step the parameter h decreases using the recursive relation $T(h) = T(h-1) + T(h - \lceil \frac{k}{2} \rceil)$. This gives a branching factor as a function of k, noted as c_k, converging to 1 as k goes to infinity (see Table 1), giving us the running time of $O({c_k}^h \cdot \text{poly}(n))$ for our branching algorithm. Note that Lemma 2 does permit (and indeed suggests) that $h_k(P)$ is likely to grow as k grows: although the branches improve, we may also expect the parameter to increase.

Table 1 Branching factors as a function of k.

k	2	3-4	5-6	7-8	9-10	11-12	13-14	15-16	17-18	19-20
c_k	2.0000	1.6180	1.4656	1.3803	1.3247	1.2852	1.2554	1.2321	1.2131	1.1975

This leads to the following theorem.

▶ **Theorem 5.** *There exists an FPT algorithm that given a polyomino P and parameter h finds a maximum k by k square packing $\mathcal{S}_k$ on P in $O({c_k}^h \cdot poly(n))$ time if $h_k(P) \leq h$.*

4 Grid-based ILP Analysis

We now analyze an integer linear programming (ILP) formulation of the Maximum $k \times k$ Square Packing problem by El-Khechen [4] and show that an ILP solver can be made to achieve the fixed parameter tractable runtime from before. For each square $S_k(a, b)$ that fits within P, the ILP has a Boolean variable $x_{(a,b)}$ indicating whether to place it or not; each cell $S_1(x, y)$ of P gets one packing constraint that prevents overlapping squares there:

$$\sum_{(a,b):S_1(x,y)\subseteq S_k(a,b)} x_{(a,b)} \leq 1.$$

Maximizing the sum of the x variables then packs the maximum number of $k \times k$ squares. Unfortunately, the number of variables is $\Theta(A)$. We will now show that the fractional relaxation contains useful information and that the branching depth of a solver can be limited by selecting good variables to branch on.

▶ **Lemma 6.** *Let P' be the polyomino created by taking the union of all $k \times k$ squares $S_k(a, b)$ for which $0 < x_{(a,b)} < 1$ in an optimal solution of the LP relaxation. Then there exists an optimal solution of the LP relaxation in which P' is reduced.*

Proof sketch. Consider a convex vertex where the incident line segments both have length at least k and let $S_k(a, b)$ be the square that touches it. There is a packing constraint for a cell contained in $S_k(a, b)$, which contains all variables of the square touching this vertex and all other squares overlapping it. Therefore, all these variables sum up to at most 1 and thus we can set the square in the corner to 1 and all other overlapping squares to 0. ◀

This lemma only proves the existence of such a fractional optimum, not that the solver will produce it – but the proof is constructive, and indeed we may expect a solver to arrive at such a solution by itself based on its own heuristics and extreme-point algorithms.

Since the polyomino P' implied by the fractional optimum of Lemma 6 is reduced, we can instruct the ILP solver to branch on a variable corresponding to branching rules 1 and 2. (Such branch selection or "branch priority" is supported by many ILP solvers.) As each branching step results in more cells that are necessarily uncovered, the corresponding packing constraint for these cells equals 0 in our LP relaxation. Therefore the LP relaxation is constrained by the number of uncovered cells generated over all branching steps to that point. As the gap between the LP relaxation and the maximum $k \times k$ square packing is at most $h_k(P)$, the analysis of the branching factor of our FPT algorithm holds for the branch and bound algorithm of the solver.

5 Concluding Remarks

In practice our algorithm is likely to perform better than the bounds proven in this paper: we expect there to commonly be a convex corner that is much better than guaranteed, especially when considering the ensuing reduction rules, and the algorithm can pick the best one.

In this paper, we have reduced the exponential dependence in runtime from the area of P to the perimeter of P. We wonder if the exponential dependence might be further reduced to the number of vertices or some other structural parameter of P.

References

1 Mikkel Abrahamsen and Jack Stade. Hardness of packing, covering and partitioning simple polygons with unit squares. In *Symposium on Foundations of Computer Science (FOCS)*, pages 1355–1371. IEEE Computer Society, 2024.

2 Christoph Baur and Sándor P. Fekete. Approximation of geometric dispersion problems. *Algorithmica*, 30(3):451–470, 2001. doi:10.1007/S00453-001-0022-X.

3 Timothy M. Chan. A note on maximum independent sets in rectangle intersection graphs. *Information Processing Letters (IPL)*, 89(1):19–23, 2004. doi:10.1016/J.IPL.2003.09.019.

4 Dania El-Khechen. *Decomposing and packing polygons*. PhD thesis, Canada, 2009.

5 Robert J. Fowler, Mike Paterson, and Steven L. Tanimoto. Optimal packing and covering in the plane are NP-complete. *Information Processing Letters (IPL)*, 12(3):133–137, 1981. doi:10.1016/0020-0190(81)90111-3.

6 Dorit S. Hochbaum and Wolfgang Maass. Approximation schemes for covering and packing problems in image processing and VLSI. *Journal of the ACM*, 32(1):130–136, 1985. doi:10.1145/2455.214106.

7 André van Renssen and Bettina Speckmann. The 2x2 simple packing problem. In *Canadian Conference on Computational Geometry*, 2011. URL: http://www.cccg.ca/proceedings/2011/papers/paper14.pdf.

Exploring Mixedness of Bichromatic Point Sets

Thijs Beurskens[1], Marc van Kreveld[2], Frank Staals[2], and Jules Wulms[1]

1 **Department of Mathematics and Computer Science, TU Eindhoven**
`[t.p.j.beurskens,j.h.h.m.wulms]@tue.nl`
2 **Department of Information and Computing Sciences, Utrecht University**
`[m.j.vankreveld, f.staals]@uu.nl`

1 Introduction

We consider sets of red and blue points. The degree of mixture, or being interspersed, of such a set is the opposite of being well-separated (see Figure 1). A set of red and blue points is well-separated if there is a simple 'separator' like a line or a circle such that the red points are to the one side and the blue points to the other. A measure for well-separateness can be defined as the number of outliers or sum of distances to the separator. These measures are simple and capture the intuition well when values are small: few outliers or small sum of distances to the separator. It seems natural to use the same measures for mixedness, but for larger values these measures no longer correspond to the intuition; for the left examples in Figure 1 it is not immediately intuitive how to draw and value separating lines. This may be due to the class of separators considered: a colored point set may have many outliers when we consider single-line separators but zero if a circular or wedge separator can be used.

We therefore want different measures that address the concept of being interspersed directly, without reference to a class of separators. Such measures do exist. For instance, one can consider the number of lines needed to make every cell of the arrangement monochromatic [3]. Alternatively, we can investigate how many monochromatic disjoint convex sets need to be made to enclose all the points [4]. Closely related is the use of a discrepancy measure [1]. Each of these counts can be converted into a measure for mixedness. A drawback however, is that computing these measures requires solving an optimization problem that is NP-hard. Hence we aim for simpler measures for mixedness that can be computed in polynomial time. Other results on colored point sets are surveyed in [5].

Desirable properties. We list some properties that we think are desirable for a mixedness measure. Other properties are also possible, and depending on the application some properties might be more important than others. First and foremost, we want the measure to be *intuitive*, *simple* to understand, and *efficient* to compute. Preferably, the measure should be *unbiased*, that is, for any uncolored point set, taking a random 2-coloring should yield an expected score that is the average of the minimum and the maximum score for that point set. Other desirable properties are *range coverage* (there are point sets that receive the minimal and

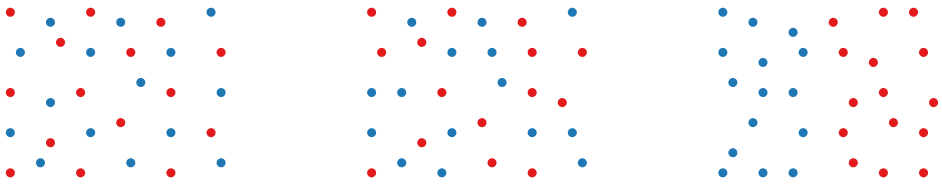


Figure 1 A very mixed, a medium mixed, and a very unmixed point set.

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

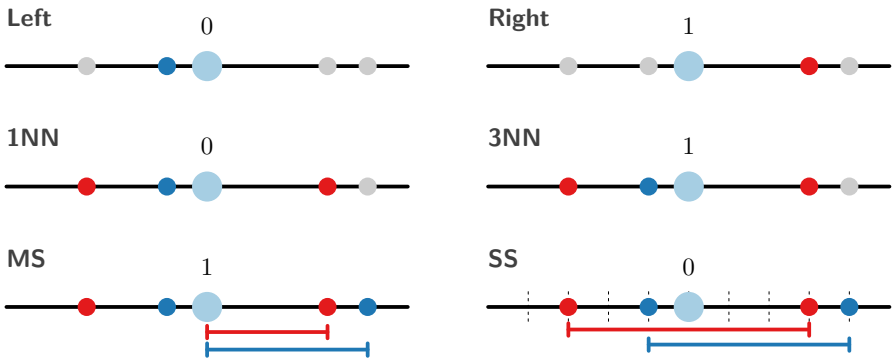


Figure 2 The different binary measures.

maximal possible scores), *invariance* (the score remains the same if we translate, reflect, or scale the entire point set), and *color symmetry* (swapping the colors yields the same score). Furthermore, we mostly consider *local* measures, that determine a score in the range $[0, 1]$ for each individual point, by considering $\mathcal{O}(1)$ other points. These scores are aggregated and, as a consequence, mixedness values lie in the range $[0, n]$, where n is the total number of points. Note that these values can be normalized to compare mixedness between different cardinality sets. However, we do not normalize in this abstract, as this conflates the ranges of individual point and aggregated scores. Lastly, we want the measures to be *color robust* (recoloring a single point changes the score by only $\mathcal{O}(1)$), and *perturbation robust*: there exists some $\varepsilon > 0$ such that perturbing each point by at most ε changes the score by only $\mathcal{O}(1)$.

Contributions. We discovered that even for points on the line, there are many natural options to define mixedness. Hence, the majority of this paper focuses on 1D point sets. In Section 2 we explore possible measures and their properties. Perhaps surprisingly, these measures can produce very different scores for the same input. In Section 3, we consider the problem of transforming a given bichromatic point set into a 'perfectly mixed' bichromatic point set. We give an algorithm to compute such a point set by deleting as few points as possible. In Section 4 we comment on extending our findings to 2D.

2 Exploring Measures for Mixedness

Let P be a sorted set of n bichromatic points on the line, that is, each point of P is assigned an x-coordinate and is colored either red or blue. We assume that no two points are positioned at the same x-coordinate. We aim for a measure of mixedness that is both 'natural', and efficient to compute. To that end, we consider measures that are based on the idea that each point of P is either completely mixed or completely unmixed (with respect to P). The *total mixedness* of P is then the number of mixed points. We call these *binary measures*.

Binary measures. We consider a number of different binary measures (see Figure 2). Perhaps the simplest measure of all is the measure **Left** (and its symmetric variant **Right**): a point is mixed if and only if its left (right) neighbor is of a different color. Note that these mixedness measures translate to 1-dimensional versions of the separability measures from [1, 3, 4]. If we care for distances, then we can use k nearest neighbors: a point p is mixed if and only if a majority of its k nearest neighbors are colored differently. In 1D, we consider only $k = 1$

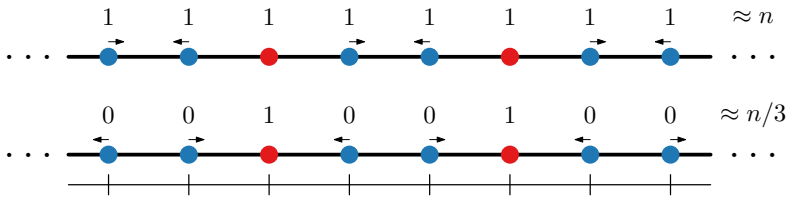


Figure 3 The score of the measure **MS** can change $\mathcal{O}(n)$ if we perturb each point by $\varepsilon \to 0$.

(**1NN**) and $k = 3$ (**3NN**); $k = 2$ could lead to ties and $k > 3$ becomes increasingly less local.

We consider two more measures, both based on the *surroundings* of points. Let p be a blue point; the definition for red points is symmetric. Let ℓ_{red} be the rightmost red point left of p; we define ℓ_{red} to be the point with x-coordinate $-\infty$ if there is no such red point. Similarly, ℓ_{blue} is the rightmost blue points left of p, and r_{red} and r_{blue} are the leftmost red and blue points right of p, respectively. For the first measure, called *shortest surround* (**SS**), p is mixed if and only if the distance between ℓ_{red} and r_{red} is strictly smaller than the distance between ℓ_{blue} and r_{blue}. For the second measure, called *minmax surround* (**MS**), p is mixed if and only if p lies strictly closer to the furthest of ℓ_{red} and r_{red} than to the furthest of ℓ_{blue} and r_{blue}. Note that the leftmost and rightmost point of the input always have a distance of ∞ to their surrounding points, so they are always assigned the score 0.

Properties. All measures can trivially be computed in linear time. It is not hard to see that all measures are unbiased, invariant, color symmetric, color robust and have range coverage. The measures **Left** and **Right** are also perturbation robust: since no two points have the same x-coordinate, perturbing the x-coordinate of each point by at most some sufficiently small $\varepsilon > 0$ does not affect the ordering of the colors, and hence does not change the scores. On the other hand, we illustrate that the measure **MS** is not perturbation robust in Figure 3. If we shift each pair of blue points inwards by any ε, then each blue point has "red" distance $2 - \varepsilon$ and "blue" distance $2 + 2\varepsilon$. So, the measure **MS** receives a score of n. However, if we shift each pair outwards by any ε, then each blue point has "red" distance $2 + \varepsilon$ and "blue" distance $2 - 2\varepsilon$; so the score becomes $n/3$. Similarly, the measures **1NN**, **3NN**, and **SS** are also not perturbation robust.

2.1 Comparing Measures

In this section we investigate whether the measures can behave very differently. In other words, given any two measures, is it possible (for some point set and coloring) that the one measure deems it very mixed and the other not so mixed?

First note that **Left** and **Right** are equivalent; both compare every consecutive pair of points. For any other pair of measures, the examples in Figure 4 establish that there is a bichromatic point set for which they produce very different results. To understand exactly how different any two measures are, one can study their *maximum discrepancy* [2, Chapter 8]. We elaborate for one pair of measures—**Left** vs. **1NN**—and comment on the other pairs.

▶ **Lemma 2.1.** *The maximum difference between the measures* **Left** *and* **1NN** *is* $n/2 \pm \mathcal{O}(1)$.

Proof. The topmost example in Figure 4 establishes the lower bound. Consider any colored point set P. By the pigeon-hole principle either for at least half the points of P their nearest neighbor is their right neighbor, or for at least half the points of P their nearest neighbor is

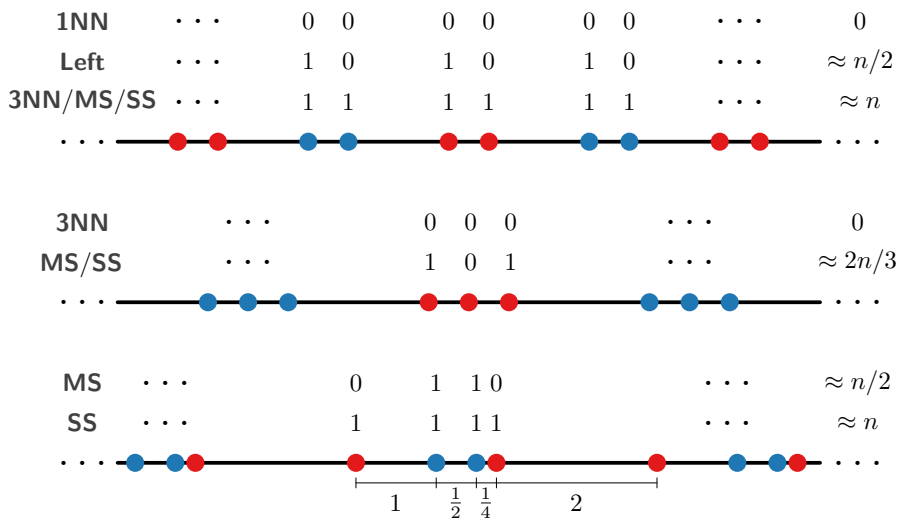


Figure 4 Any pair of measures can produce very different scores on the same input.

their left neighbor. Hence, the score of **1NN** differs from either **Left** or **Right** by at most $n/2 \pm \mathcal{O}(1)$. The measures **Left** and **Right** are equivalent, so the upper bound follows. ◀

For the remaining pairs, we can obtain similar statements. The examples in Figure 4 establish lower bounds for the maximum differences. As another example, the maximum difference between **1NN** and **3NN** is n: Figure 4 shows the lower bound, and the upper bound is trivial.

2.2 Other Measures

We can also think of other, non-binary, measures. As an example, we can think of measures that assign fractional scores to each point, such as fractional kNN measures (**FkNN**). For these measures, each point p scores a value $\frac{r}{k}$, where r is the number of points of a different color among the k nearest neighbors. Note that **F1NN** is equivalent to **1NN**. Similarly, we can define fractional versions of **SS** and **MS**, e.g. by taking the ratio between the number of points of the other color and the total number of points in the surrounding of p (disregarding p itself). Moreover, one could define real-valued measures; e.g. each point receives a score proportional to its distance to the closest point of a different color. We leave a more detailed review of different types of measures for future work.

3 Distance to Perfection

We now consider the problem of transforming a given bichromatic point set P into a point set P', such that P' scores maximally on a particular measure. The optimization criterion is to change P as little as possible. This problem can be motivated in two ways: (1) it provides us with a way of obtaining 'perfectly mixed' point sets via the chosen operation; (2) by minimally changing P, we are essentially measuring how far P is from being 'perfectly mixed'. While this *distance to perfection* could also be defined as the difference between the measure value and n, this simple definition does not always behave intuitively: for the topmost point set in Figure 4 **1NN** is 0, but moving/removing only $n/2$ points can increase **1NN** to $n/2$.

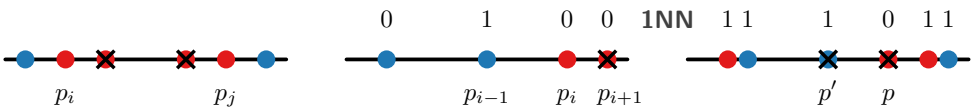


Figure 5 Delete operations in first (left), second (middle), and third (right) sweep.

To obtain a concrete instance of such an optimization problem, we need to choose an *operation* to transform P and a measure. We efficiently solve the problem for the operation of deleting points and the measure **1NN**. Specifically, we want to find a maximal subset $P' \subset P$ for which **1NN** scores $|P'|$. To find a perfectly mixed subset, one can trivially delete all but one point out of each maximal sequence of consecutive monochromatic points. However, finding a maximal subset is more involved, but can still be done in linear time.

▶ **Theorem 3.1.** *Given a sorted set P of n bichromatic points in $\mathbb{R}^1$, let $P' \subseteq P$ be a maximum subset, for which* **1NN** *is $|P'|$ on P'. We can compute such a P' in $\mathcal{O}(n)$ time.*

Proof Sketch. We first describe an algorithm to delete points from P to obtain such a $P' \subseteq P$. Our algorithm consists of three linear sweeps through the point set in order of increasing coordinates, and evaluates only the color and **1NN** score of each point during each sweep to determine which points to delete (see Figure 5). For ease of explanation, we assume that after each sweep and after each removal of a point, the remaining points are re-indexed to have consecutive indices.

Algorithm. The first sweep identifies maximal sequences $p_i, p_{i+1}, \ldots, p_j$ of consecutive monochromatic points of length at least three. For each such sequence, all points between p_i and p_j are deleted. The second sweep identifies monochromatic pairs p_i, p_{i+1} of consecutive points that score 0 on **1NN**. If deleting p_i decreases the score of p_{i-1} from 1 to 0, then delete p_{i+1}, otherwise delete p_i. The third and final sweep checks for each point p whether it scores 0 on **1NN**, if so delete p and consider each neighbor p' of p. If such a p' has a different color from p and now scores 0 on **1NN**, also delete p'. The remaining points form the set $P' \subseteq P$.

Invariants. The following invariants hold for all points left of the sweep line in the respective left-to-right sweeps:

1. For the first sweep, at most two consecutive points can have the same color.
2. For the second sweep, no two monochromatic consecutive points can score 0.
3. For the third sweep, only points scoring 1 remain.

Correctness. To prove the correctness of our algorithm, we have to show that (1) every point in P' scores 1 on **1NN** and (2) there is no $P^* \subseteq P$, with $|P^*| > |P'|$, in which every point scores 1 on **1NN**. The correctness of (1) follows directly from the correctness of Invariant 3.

To prove (2), assume for contradiction that such a $P^* \subseteq P$ exists. The proof can be sketched as follows. We first argue that out of the sequences considered in the first sweep, and the pairs considered in the second sweep, P^* contains at most as many points as P'. Then we consider the choices made by our algorithm in the first and second sweep, and argue that choosing the points in P^* instead of the points chosen by our algorithm in those sequences and pairs, will result in at least as many remaining 0 scoring points at the start of the third sweep. Finally, we consider what happens when all points that currently score 0 are deleted, and compare the choices of our algorithm to the choices that lead to P^*. This final step uses a case distinction on which sweep introduced each 0 score, and each case shows that P' lost at most as many points as P^*. As a result, $|P^*| \leq |P'|$, leading to a contradiction.

Running time. During each sweep, the points in P are visited in left-to-right order. We can store the (sorted) point set P in a (doubly) linked list, so that during each sweep the neighbors of a visited point can be queried in $\mathcal{O}(1)$ time. When a point p is visited or queried, its **1NN** value can be computed in $\mathcal{O}(1)$ time by querying the neighbors of p. As such, each of the three sweeps takes $\mathcal{O}(n)$ time, leading to $\mathcal{O}(n)$ total running time. ◀

4 Mixedness in the Plane

So far, we considered mixedness of colored points in 1D. We briefly comment on our findings for mixedness in 2D. A first observation about going from 1D to 2D is that the number of 'extremal' points can increase significantly. In 1D only the left-and rightmost points are extremal. However, in 2D all points are extremal if the point set is in convex position. Moreover, in 2D there is no clear notion of 'left' and 'right', and hence also not of 'surrounding'.

Therefore, the measures **Left** and **Right** do not directly extend to (invariant) measures on the plane. On the other hand, the measures **1NN** and **3NN** do extend directly to the plane. To extend the measures **SS** and **MS**, we need a different interpretation of the surrounding of a point. For instance, for each point p we could use an interpretation based on convex hulls. Specifically, we can maintain two convex hulls (a red hull and a blue hull) by iteratively adding the next nearest neighbor of p, until both convex hulls contain p. If we assign a score to p based on the area of the convex hulls, we recover the measure **MS**. If we assign a score to p based on the furthest point among all convex hull points, we recover the measure **SS**. In the plane, the concept of surrounding also induces other (fractional) measures, such as determining the fraction of points in the convex hull of the other color. A thorough investigation of mixedness in 2D (or even higher dimensions) is left for future work.

Acknowledgements. Research on the topic of this paper was initiated at the 9th Workshop on Applied Geometric Algorithms (AGA 2025), supported by the Dutch Research Council (NWO) under project no. VI.Veni.212.260.

References

1 Sergey Bereg, José Miguel Díaz-Báñez, Dolores Lara, Pablo Pérez-Lantero, Carlos Seara, and Jorge Urrutia. On the coarseness of bicolored point sets. *Computational Geometry*, 46(1):65–77, 2013. doi:10.1016/j.comgeo.2012.04.003.

2 Mark de Berg, Otfried Cheong, Marc van Kreveld, and Mark Overmars. *Computational Geometry: Algorithms and Applications*. Springer, 3rd edition, 2008. doi:10.1007/978-3-540-77974-2_4.

3 Panos Giannopoulos, Eduard Bonnet, and Michael Lampis. On the parameterized complexity of red-blue points separation. *Journal of Computational Geometry*, 10(1):181–206, 2019. doi:10.20382/jocg.v10i1a7.

4 Pantea Haghighatkhah, Wouter Meulemans, Bettina Speckman, Jérôme Urhausen, and Kevin Verbeek. Obstructing classification via projection. *Computing in Geometry and Topology*, 1(1):2:1–2:21, 2022. doi:10.57717/cgt.v1i1.8.

5 Mikio Kano and Jorge Urrutia. Discrete geometry on colored point sets in the plane—a survey. *Graphs and Combinatorics*, 37(1):1–53, 2021. doi:10.1007/s00373-020-02210-8.

NP-Completeness Proofs of All or Nothing, Water Walk, and Remembered Length Using the T-Metacell Framework

Pakapim Eua-anant[1], Papangkorn Apinyanon[2], Thunyatorn Jirachaisri[3], Nantapong Ruangsuksriwong[1], and Suthee Ruangwises[1]

1 **Department of Computer Engineering, Faculty of Engineering, Chulalongkorn University, Bangkok, Thailand**
pakpim.e@gmail.com, teamnantapong@gmail.com, suthee@cp.eng.chula.ac.th

2 **St. Stephen's Episcopal School, Austin, TX, USA**
apinyanonp@gmail.com

3 **New Hampton School, New Hampton, NH, USA**
tjnjtjcj@gmail.com

Abstract

All or Nothing, Water Walk, and Remembered Length are pencil puzzles that involve constructing a continuous loop on a rectangular grid subject to certain constraints. In this paper, we analyze their computational complexity using the T-metacell framework developed by Tang [15] and MIT Hardness Group [5]. We prove that all three puzzles are NP-complete by reductions from known NP-complete problems. Specifically, the first two puzzles are reduced from the problem of finding a Hamiltonian cycle in a maximum-degree-3 spanning subgraph of a rectangular grid graph, while the last is reduced from the problem of finding a Hamiltonian cycle in a required-edge directed rectangular grid graph.

1 Introduction

Pencil puzzles such as Sudoku, Nonogram, and Numberlink are logic-based puzzles designed to be solved on paper through deductive reasoning. Beyond their recreational appeal, these puzzles are also of theoretical interest in computational complexity, as they exemplify problems that are easy to state yet computationally hard to solve. Many pencil puzzles have been proven NP-complete through various reduction frameworks, including Chained Block [8], Choco Banana [9], Five Cells [7], Hashiwokakero [3], Kakuro [16], Numberlink [1], Roma [4], Slitherlink [16], Sudoku [16], Suguru [13], Sumplete [6], Tatamibari [2], and Zeiger [14].

1.1 T-Metacell

Many pencil puzzles that require constructing a single continuous loop have been shown to be computationally hard via reductions from variants of the Hamiltonian cycle problem. In 2022, Tang [15] introduced a concise framework, called the *T-metacell*, for proving NP-hardness of such puzzles using grid-graph Hamiltonicity.

A T-metacell is a gadget representing a degree-3 vertex of a grid graph. Typically constructed as a square tile, it has exits on three of its four sides, allowing the loop to pass through the gadget between any pair of these exits. The gadget can be rotated, and the loop may traverse between adjacent T-metacells only when both cells have exits on their shared edge. Furthermore, the loop is required to visit every T-metacell exactly once.

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

Tang [15] gave the following argument showing that, for the Hamiltonicity problem in maximum-degree-3 spanning subgraphs of a rectangular grid graph, all vertices in the graph, including those with degree less than three, can be represented using T-metacells.

Let G be a maximum-degree-3 spanning subgraph of a rectangular grid graph, and let H be its complementary graph; that is, two vertices are adjacent in H if and only if they are not adjacent in G. Because every vertex in G has degree at least two, each vertex in H has degree at most two. Thus H consists only of vertices of degree 0, 1, or 2. (Vertices of degree 0 in H, which correspond to corners, are not relevant to our calculation.)

We then orient the edges of H so that every vertex has indegree at most one and outdegree at most one. Using this orientation, we construct G with T-metacells by replacing each vertex with a T-metacell such that, for every degree-2 vertex, the *third exit* (i.e., the exit that does not correspond to an edge of G) faces the direction of the outward-pointing edge of the corresponding vertex in H. This direction corresponds to a non-exit side of the adjacent cell, as illustrated in Figure 1.

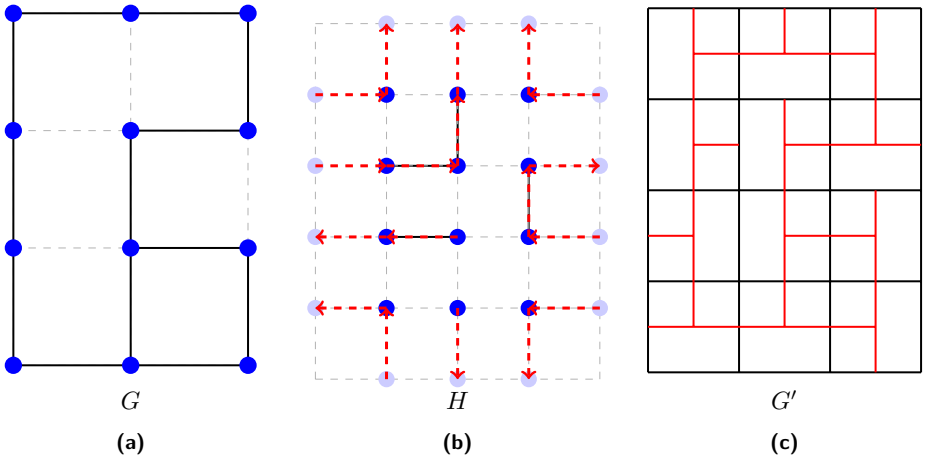


Figure 1 Graphs G, H, and the corresponding graph G' constructed from G using T-metacells

1.2 Required-edge Directed T-Metacell

MIT Hardness Group [5] extended the T-metacell framework to directed graphs. A T-metacell is *required-edge directed* if each of its three exits has a prescribed direction the cycle must follow. They showed that finding a Hamiltonian cycle on a rectangular grid of required-edge directed T-metacells is NP-complete via a reduction from the Hamiltonicity problem in a directed maximum-degree-3 spanning subgraph of a rectangular grid graph. When constructing a grid of T-metacells, the cells need not be identical; it suffices that for each required edge and each orientation, there exists a corresponding T-metacell configuration.

1.3 Our Contribution

In this paper, we prove the NP-completeness of three pencil puzzles: *All or Nothing*, *Water Walk*, and *Remembered Length* (including its weaker version in which no cells are shaded), using the T-metacell framework. Specifically, we construct reductions from the problem of finding a Hamiltonian cycle in a maximum-degree-3 spanning subgraph of a rectangular grid graph.

2 NP-Completeness Proofs

Since the problems clearly belong to NP, it remains to prove NP-hardness. In this section, we present reductions showing that the three puzzles are NP-hard using the T-metacell framework. In each reduction, vertices of the grid graph are mapped to T-metacell gadgets that simulate the constraints of the corresponding puzzle.

2.1 All or Nothing

All or Nothing is a pencil puzzle invented by Inaba Naoki [10]. In this puzzle, a rectangular grid is partitioned into polyominoes called *regions*. The goal is to draw a non-crossing loop through orthogonally adjacent cells that satisfies the following three constraints.

1. If the loop visits a region, it must pass through every cell in that region.
2. The loop may enter and exit each region at most once.
3. Two unvisited regions cannot be orthogonally adjacent. See Figure 2.

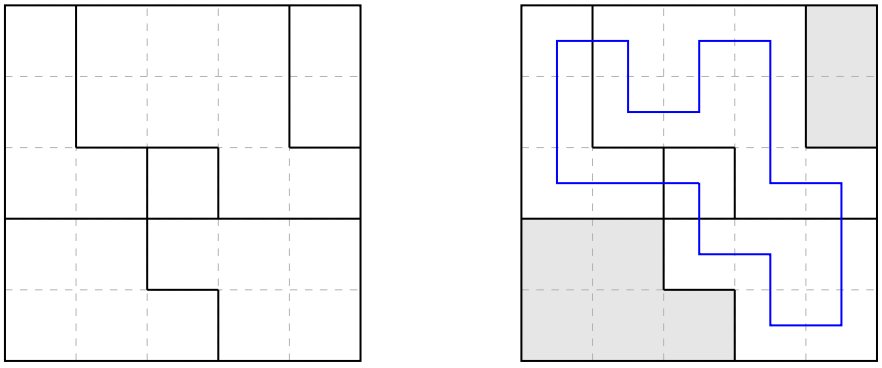


Figure 2 An All or Nothing instance (left) and one of its solutions (right)

▶ **Theorem 2.1.** *All or Nothing is NP-hard.*

Proof. We use the term *part* to denote an orthogonally connected set of cells in a T-metacell (which may not be fully enclosed by region boundaries), and *region* to denote a completely enclosed part in a full puzzle instance. A part or region is said to be *leaf-rich* if it contains at least three *leaves* (cells adjacent to exactly one other cell within the same part or region). A region is said to be *dead* if no valid loop can visit it.

We construct a T-metacell for *All or Nothing* using an 11×11 grid gadget, shown in Figure 3. Each gadget consists of a large white region (which must be visited by the loop), a one-cell region enclosed within the large region, and three partially enclosed parts, referred to as the *foundational parts*. Note that all foundational parts are leaf-rich, and the gadget itself is not enclosed by region boundaries.

When the full puzzle instance is constructed by combining the T-metacells, all parts together form enclosed regions. The outer boundary of the puzzle is also treated as a region boundary. As shown in Figure 4, there exists a path between any pair of exits of the T-metacell that traverses the entire large white region. To show that the gadget satisfies the remaining properties of a T-metacell, we prove that the three foundational parts cannot be visited by the loop.

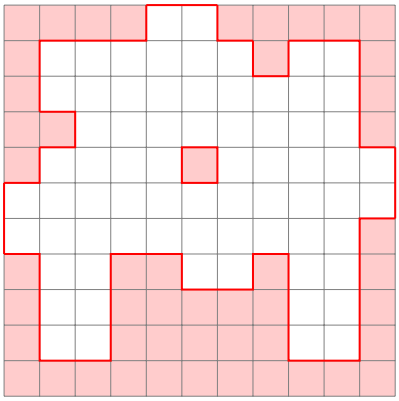


Figure 3 A T-metacell gadget for All or Nothing, with the red lines indicating region boundaries, and the shaded area indicating foundational parts and the one-cell region

First, the one-cell region is dead: entering it would force the loop to re-enter its surrounding region, violating the rule that a region can be entered and exited at most once.

Second, any leaf-rich region is dead. Indeed, visiting each leaf requires the loop to cross the boundary of the region, while a valid loop may cross a region boundary exactly twice. Therefore, a region with at least three leaves cannot be visited.

Finally, we show that in the full puzzle instance, any region containing a foundational part is leaf-rich. Consider a foundational part of a T-metacell. Within this part there is one +-type cell (which is always a leaf) and two ?-type cells located on its borders, as shown in Figure 5. Each ?-type cell is a leaf when the T-metacell is considered in isolation, but it may cease to be one when connected to an adjacent T-metacell, as illustrated in Figure 6.

However, the ?-type cells from adjacent T-metacells do not cancel each other out. Because the foundational parts of neighboring T-metacells are vertically shifted relative to one another, a ?-type cell from one gadget extends further along the shared edge than the corresponding ?-type cell of the adjacent gadget, and thus remains a leaf.

For each ?-type cell, the boundary on its side either coincides with the outer boundary of the full puzzle instance or connects to a foundational part of another T-metacell. In the former case, the ?-type cell remains a leaf. In the latter case, the two foundational parts merge into a single region that still contains a leaf from the neighboring gadget, namely its +-type cell. Thus, each of the two ?-type cells corresponds to a distinct leaf in the resulting region. Together with the part's own +-type cell, the region contains at least three leaves and is therefore leaf-rich.

Consequently, every region containing a foundational part is leaf-rich and hence dead. Therefore, the loop cannot enter any foundational part, and the gadget satisfies all required properties of a T-metacell. ◀

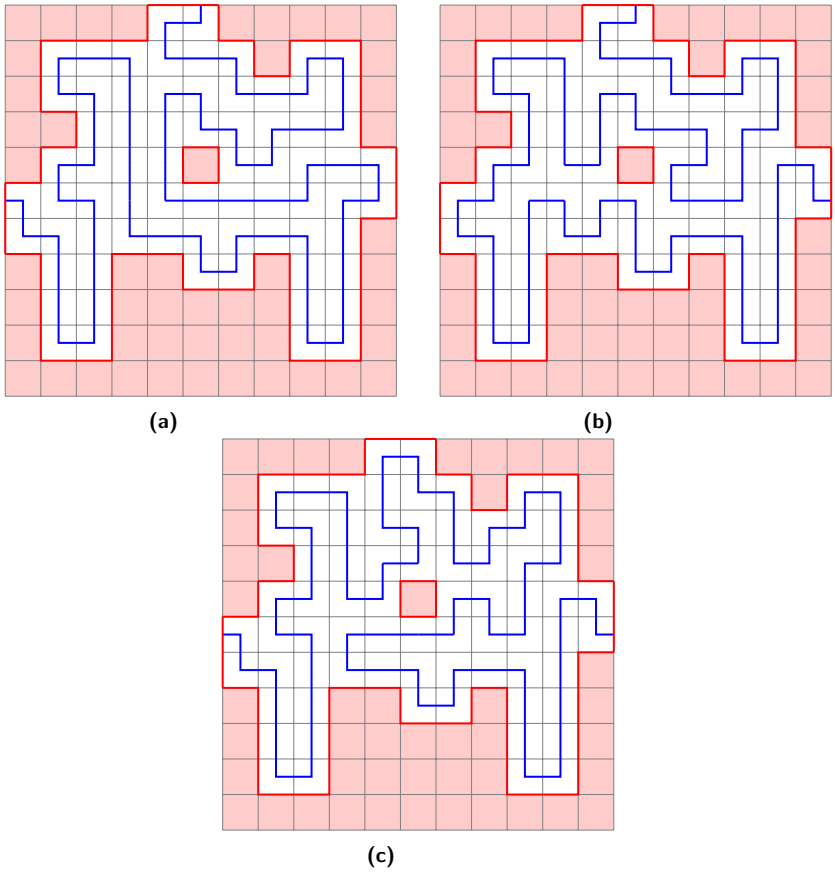


Figure 4 Path between each pair of exits shown in blue

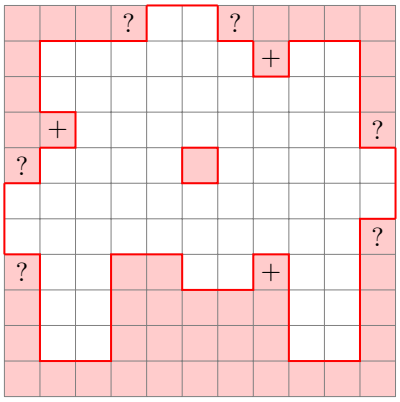


Figure 5 +-type cells and ?-type cells in the foundational parts of a T-metacell

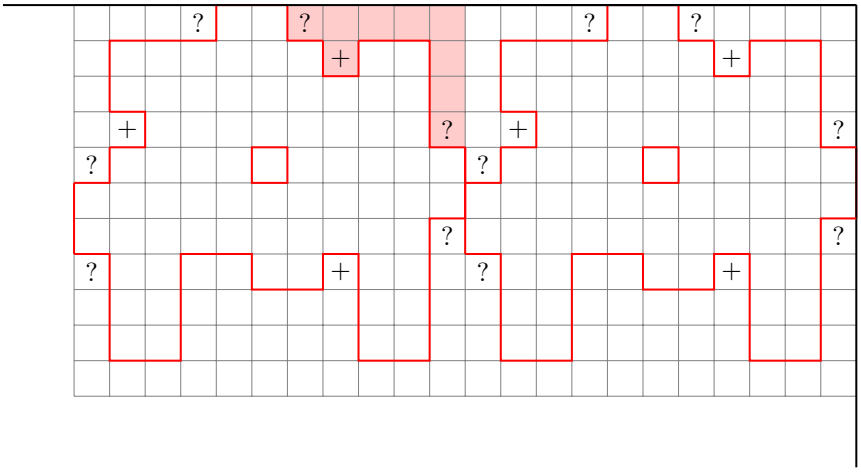


Figure 6 Two adjacent T-metacells at the top-right corner of the full puzzle instance, with the shaded area representing the foundational part of the left T-metacell. Its leftmost ?-type cell remains a leaf as it touches the outer boundary, while its rightmost ?-type cell ceases to be a leaf.

2.2 Water Walk

Water Walk is a pencil puzzle invented by Martin Ender [12]. In this puzzle, a rectangular grid consisting of *ground* and *water* cells is given, with some ground cells containing a number. The goal is to draw a non-crossing loop through orthogonally adjacent cells that satisfies the following three constraints.

1. The loop must pass through all numbered cells.
2. For each numbered cell, the number of ground cells between the nearest water cells on either side along the loop must equal the number written in the cell.
3. The loop cannot pass through three consecutive water cells. See Figure 7.

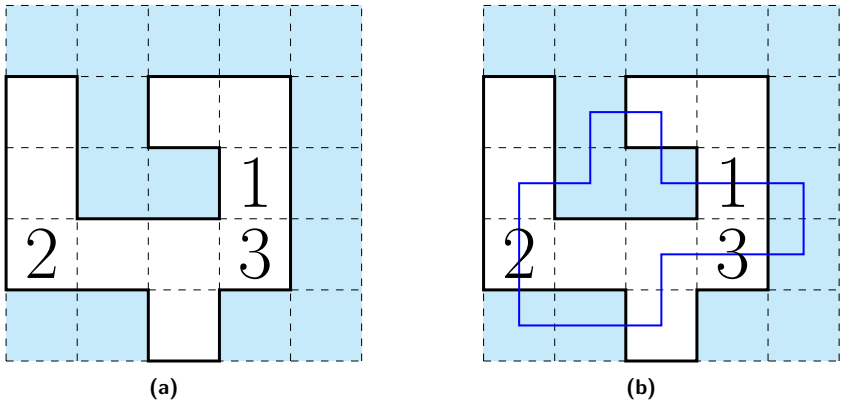


Figure 7 A Water Walk instance (left) and one of its solutions (right)

▶ **Theorem 2.2.** *Water Walk is NP-hard.*

Proof. A simple T-metacell for the puzzle is shown in Figure 8. The paths corresponding to the three possible exit choices are straightforward but not unique: there are two possible paths between exits on adjacent sides and three between exits on opposite sides. Exiting through the long side is impossible, as this would create three consecutive water cells in the loop, violating the puzzle rules. Hence the gadget correctly simulates a degree-3 vertex in the T-metacell framework, completing the reduction. ◀

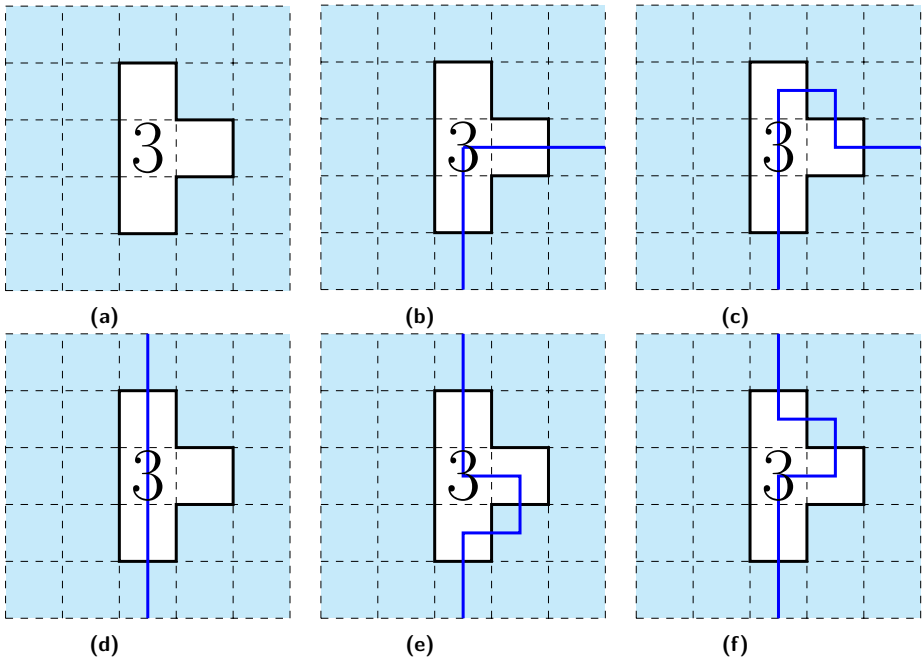


Figure 8 T-metacell for Water Walk (a), the two paths between exits on adjacent sides (b–c), and the three paths between exits on opposite sides (d–f)

2.3 Remembered Length

Remembered Length is a pencil puzzle invented by Palmer Mebane [11]. In this puzzle, a rectangular grid is partitioned into polyominoes called *regions*. Some cells are shaded, and some regions are labeled with a positive integer. The goal is to draw a non-crossing directed loop through orthogonally adjacent cells that satisfies the following constraints.

1. The loop must pass through all unshaded cells.
2. Each time the loop exits a region containing a number, the segment of the loop in the next region must consist of exactly that number of cells. See Figure 9.

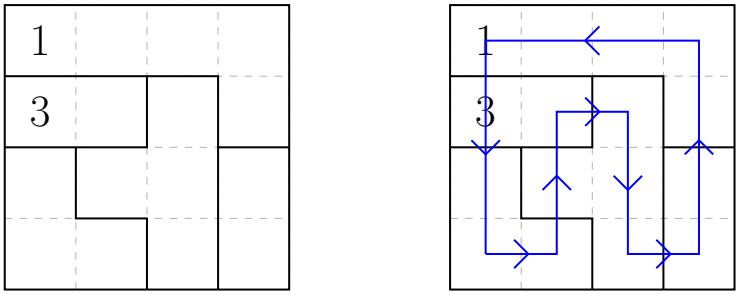


Figure 9 A Remembered Length instance (left) and one of its solutions (right)

MIT Hardness Group [5] showed that finding a Hamiltonian path in a required-edge subgraph of a rectangular grid graph is NP-complete. It follows immediately that Remembered Length is NP-complete: given an instance of this problem, we take its bounding rectangular grid graph and shade the vertices that are not present in the original graph.

We further show that the weaker variant, in which no cells are shaded, is also NP-complete. As illustrated in Figure 10, four directed T-metacells suffice; under rotation, they realize all possible degree-3 vertices of a directed grid graph.

▶ **Theorem 2.3.** *Remembered Length is NP-hard. Moreover, the weaker variant of Remembered Length in which no cells are shaded is also NP-hard.*

Proof. We use the term *ocean* to denote the largest region in the T-metacell, and *island* to denote a connected group of regions that are not part of the ocean. The island closer to the boundary of the ocean is called the *in-island*, whose minimum distance from the ocean boundary is two cells. The other island is called the *out-island*, whose minimum distance is three cells.

Each time the loop exits a T-metacell, the remembered length is reset to two. Consequently, the loop can enter the next T-metacell only on sides adjacent to an in-island and must proceed directly toward that island. On the in-island, there is a region whose number is five less than the number of cells in the ocean. Hence, after visiting this region, the loop must immediately traverse all but five cells of the ocean.

Therefore, the loop can enter a T-metacell only through the in-island (using two cells) and must exit in a straight line through one side of the out-island (using three cells), and this can occur only once. Moreover, the loop can enter and exit any island at most once, since there are not enough cells available to allow a second traversal. Because the loop always exits at the center of a side of the T-metacell, it must also enter at the center of a side.

◀

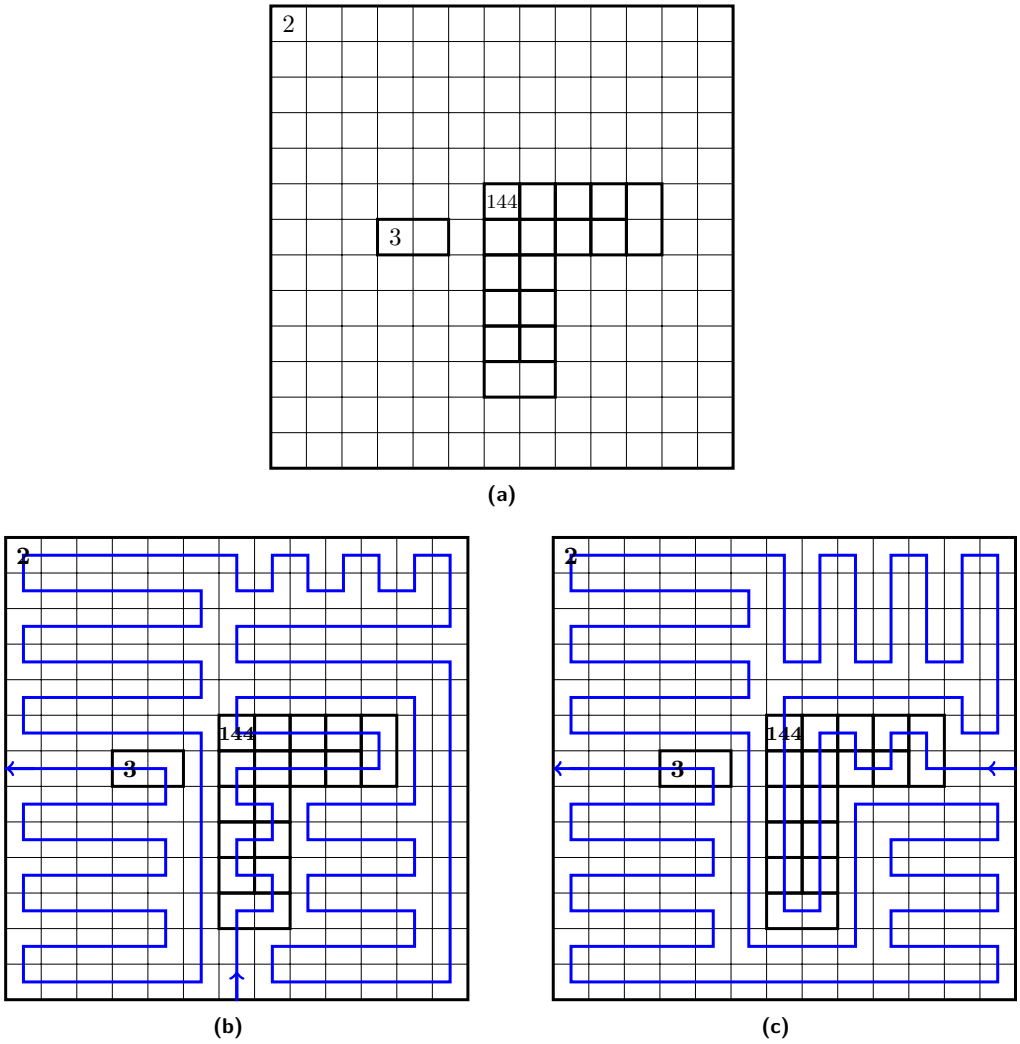
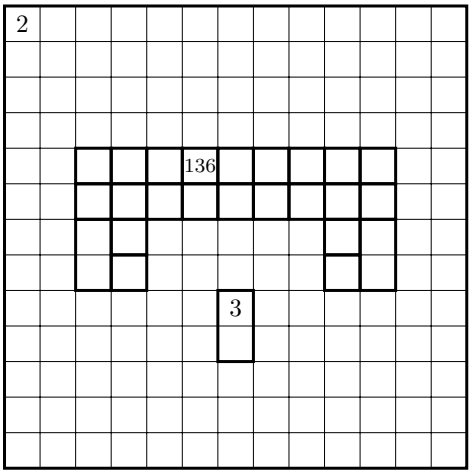
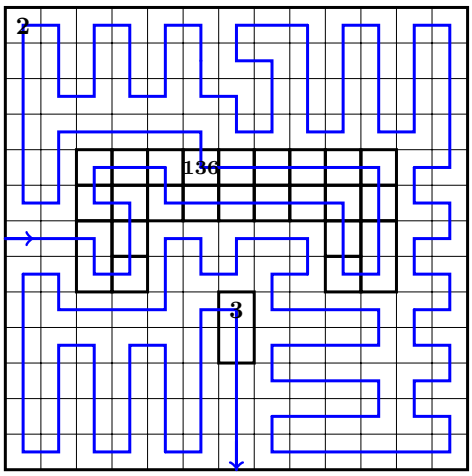


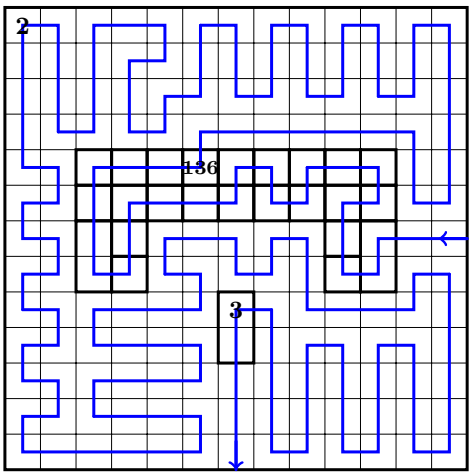
Figure 10 T-metacells for Remembered Length represent a vertex with 2 in-edges on adjacent sides (a), and the path between each chosen in-edge (b-c)



(d)

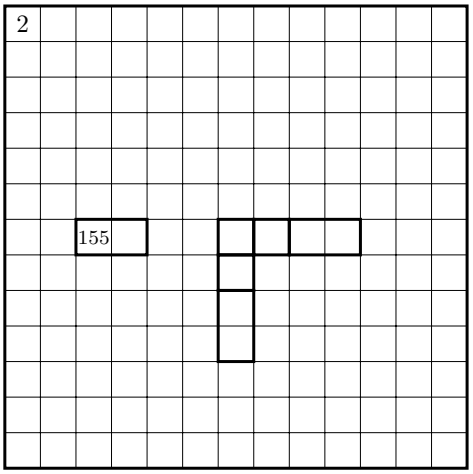


(e)

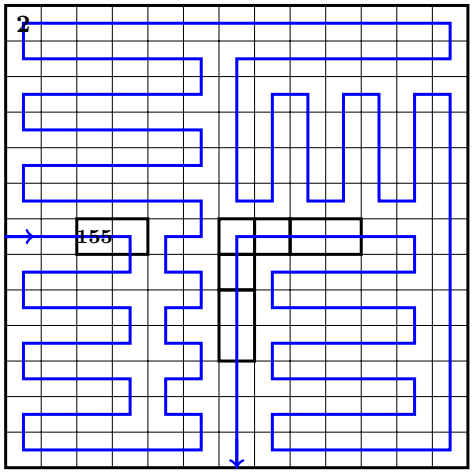


(f)

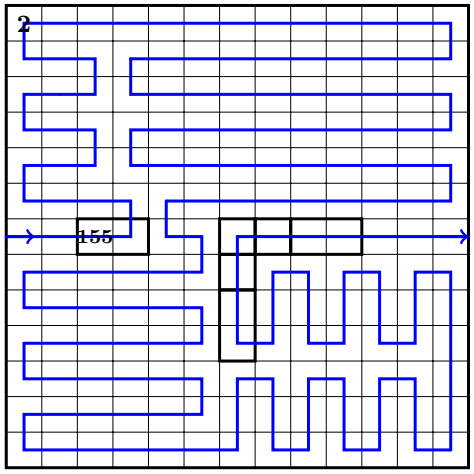
Figure 10 (continued) T-metacells for Remembered Length represent a vertex with 2 in-edges on opposite sides (d), and the path between each chosen in-edge (e-f)



(g)

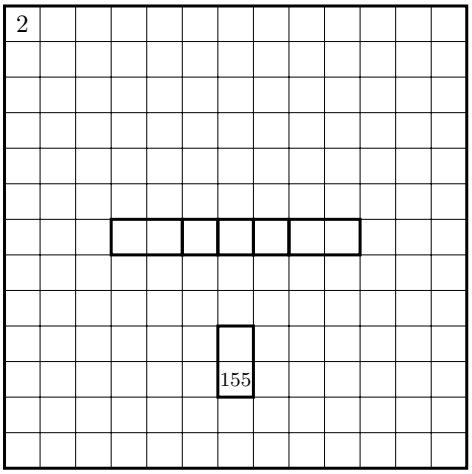


(h)

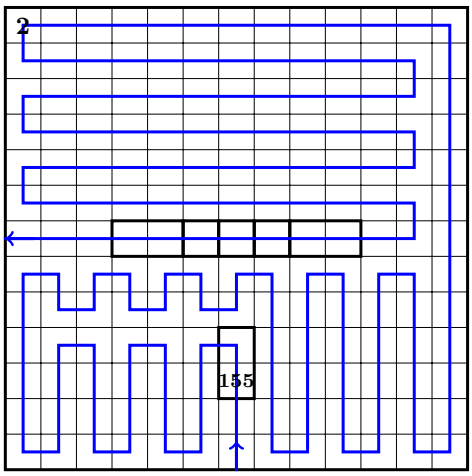


(i)

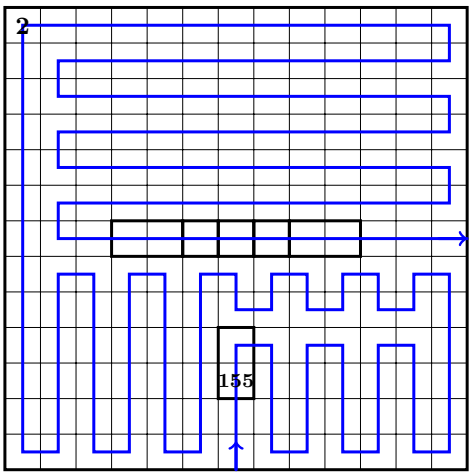
Figure 10 (continued) T-metacells for Remembered Length represent a vertex with 2 out-edges on adjacent sides (g), and the path between each chosen out-edge (h-i)



(j)



(k)



(l)

Figure 10 (continued) T-metacells for Remembered Length represent a vertex with 2 out-edges on opposite sides (j), and the path between each chosen out-edge (k-l)

3 Conclusion and Future Work

We proved that three puzzles: All or Nothing, Water Walk, and Remembered Length (including its weaker variant in which no cells are shaded) are NP-complete via T-metacell reductions from the grid graph Hamiltonicity problem.

Acknowledgement

This work was supported by grants for development of new faculty staff, Ratchadaphiseksomphot Fund, Chulalongkorn University.

References

1 Aaron B. Adcock, Erik D. Demaine, Martin L. Demaine, Michael P. O'Brien, Felix Reidl, Fernando Sánchez Villaamil, and Blair D. Sullivan. Zig-Zag Numberlink is NP-Complete. *J. Inf. Process.*, 23(3):239–245, 2015. doi:10.2197/IPSJJIP.23.239.

2 Aviv Adler, Jeffrey Bosboom, Erik D. Demaine, Martin L. Demaine, Quanquan C. Liu, and Jayson Lynch. Tatamibari Is NP-Complete. In Martin Farach-Colton, Giuseppe Prencipe, and Ryuhei Uehara, editors, *10th International Conference on Fun with Algorithms, FUN 2021, May 30 to June 1, 2021, Favignana Island, Sicily, Italy*, volume 157 of *LIPIcs*, pages 1:1–1:24. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPICS.FUN.2021.1.

3 Daniel Andersson. Hashiwokakero is NP-Complete. *Inf. Process. Lett.*, 109(19):1145–1146, 2009. doi:10.1016/J.IPL.2009.07.017.

4 Kevin Goergen, Henning Fernau, Esther Oest, and Petra Wolf. All Paths Lead to Rome. In Jin Akiyama, Hiro Ito, and Toshinori Sakai, editors, *Discrete and Computational Geometry, Graphs, and Games - 24th Japanese Conference, JCDCGGG 2022, Virtual Event, September 9-11, 2022, Revised Selected Papers*, volume 14364 of *Lecture Notes in Computer Science*, pages 228–256. Springer, 2022. doi:10.1007/978-3-032-00281-5_15.

5 MIT Hardness Group, Josh Brunner, Lily Chung, Erik D. Demaine, Della H. Hendrickson, and Andy Tockman. ASP-Completeness of Hamiltonicity in Grid Graphs, with Applications to Loop Puzzles. In Andrei Z. Broder and Tami Tamir, editors, *12th International Conference on Fun with Algorithms, FUN 2024, June 4-8, 2024, Island of La Maddalena, Sardinia, Italy*, volume 291 of *LIPIcs*, pages 23:1–23:20. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2024. doi:10.4230/LIPICS.FUN.2024.23.

6 Kyosuke Hatsugai, Suthee Ruangwises, Kyoichi Asano, and Yoshiki Abe. NP-Completeness and Physical Zero-Knowledge Proofs for Sumplete, a Puzzle Generated by ChatGPT. *New Gener. Comput.*, 42(3):429–448, 2024. doi:10.1007/S00354-024-00267-0.

7 Chuzo Iwamoto and Tatsuya Ide. Five Cells and Tilepaint are NP-Complete. *IEICE Trans. Inf. Syst.*, 105-D(3):508–516, 2022. doi:10.1587/TRANSINF.2021FCP0001.

8 Chuzo Iwamoto and Tatsuya Ide. Chained Block is NP-Complete. *IEICE Trans. Inf. Syst.*, 107(3):320–324, 2024. doi:10.1587/TRANSINF.2023FCL0001.

9 Chuzo Iwamoto and Takeru Tokunaga. Choco Banana is NP-Complete. *IEICE Trans. Fundam. Electron. Commun. Comput. Sci.*, 107(9):1488–1491, 2024. doi:10.1587/TRANSFUN.2023DML0001.

10 puzz.link. All or Nothing. https://puzz.link/rules.html?nothing. Accessed: 2025-10-24.

11 puzz.link. Remembered Length. https://puzz.link/rules.html?remlen. Accessed: 2026-01-04.

12 puzz.link. Water Walk. https://puzz.link/rules.html?waterwalk. Accessed: 2025-10-24.

13 Léo Robert, Daiki Miyahara, Pascal Lafourcade, Luc Libralesso, and Takaaki Mizuki. Physical Zero-Knowledge Proof and NP-Completeness Proof of Suguru Puzzle. *Inf. Comput.*, 285(B):104858, 2022. doi:10.1016/J.IC.2021.104858.

14 Suthee Ruangwises. NP-Completeness and Physical Zero-Knowledge Proofs for Zeiger. In Shin-ichi Nakano and Mingyu Xiao, editors, *WALCOM: Algorithms and Computation - 19th International Conference and Workshops on Algorithms and Computation, WALCOM 2025, Chengdu, China, February 28 - March 2, 2025, Proceedings*, volume 15411 of *Lecture Notes in Computer Science*, pages 312–325. Springer, 2025. doi:10.1007/978-981-96-2845-2_20.

15 Hadyn Tang. A Framework for Loop and Path Puzzle Satisfiability NP-Hardness Results. *CoRR*, abs/2202.02046, 2022. URL: https://arxiv.org/abs/2202.02046, arXiv:2202.02046.

16 Takayuki Yato and Takahiro Seta. Complexity and Completeness of Finding Another Solution and Its Application to Puzzles. *IEICE Trans. Fundam. Electron. Commun. Comput. Sci.*, 86-A(5):1052–1060, 2003.

Incremental k-lowest planes and planar k-nearest neighbor with optimal query time

John Iacono[*1], **Yakov Nekrich**[†2], **and Martin P. Seybold**[*1]

1 **Université libre de Bruxelles, Belgium**
2 **Michigan Technological University, Houghton, MI, USA**

Abstract

In a set of three-dimensional planes, the *k-lowest planes query* asks for the k lowest planes pierced by a vertical query line q.

In this paper we describe a semi-dynamic insertion-only data structure that answers k-lowest planes queries in optimal $O(\log n + k)$ time and supports insertions in poly-logarithmic time.

1 Preliminaries: The k-Level of Planes and Planar Triangulations

We rely on the standard, general position assumption: Any point $p \in \mathbb{R}^3$ is contained in $|\{h \in H : p \in h\}| \leq 3$ planes, and no plane in H contains a vertical line. I.e. any vertical query line q intersects all $|H|$ planes. The at most k-level of a set of planes H, denoted by $L(k, H)$, is the set of points from $\mathbb{R}^3$ that is the closure of the region of points that have $< k$ planes below it. The k-level $L(k, H)$ is a polyhedral surface with $O(|H|k^{3/2})$ vertices [2, Sec. 3][13], and the fastest known algorithm computes it in deterministic $O(|H| \log |H| + |H|k^{3/2})$ time [5]. Recall that the k-level of planes, tangent to the (downward opened) lifting paraboloid [7, Sec. 11.5], is in one-to-one relation to the planar order-k Voronoi diagram.

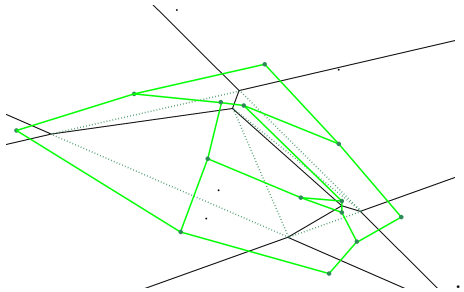


Figure 1 Order-2 Voronoi diagram of fives sites (black), its triangulation (dotted green lines), and triangle-adjacency graph (green).

We will frequently project (sets of) points in 3D with an orthogonal projection down into the xy-plane at $z = -\infty$, and work with 2D objects in that plane. This will simply be referred to as *projection* (into the plane) throughout the entire paper.

In our insertion algorithm, we will frequently compute the $\bar{k}$-level of certain subsets $H' \subseteq H$ where $\bar{k}$ is a fixed integer, project it into the plane, and triangulate the faces. The number of faces in the triangulation is near-linear $O(|H'|\bar{k}^{3/2})$ in $|H'|$ for small $\bar{k}$. Any face of the triangulation shares a common boundary with two or three other faces, and any

* Work supported by the Fonds de la Recherche Scientifique – FNRS.
† Supported by the National Science Foundation under NSF grant 2203278.

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

unbounded face is bounded by two rays and possibly one segment. We call the unbounded faces degenerate triangles, as they may contain only one or two corner points. (See Figure 1.) For a vertical query line that intersects an edge of the k-level, we allow either set of k-lowest planes from the faces as query result. That is, we allow that ties are broken arbitrarily in reporting queries, and thus assume that queries intersect triangles in their *interior*.

We will also use algorithms for computing separators in a planar graph with n nodes [10, 11, 12], specifically in the triangle-adjacency graph described above. Using [12, Sec. 2.5], for given integer $r < n$ one can compute in $O(n)$ time a decomposition in $O(n/r)$ connected components, where each connected component contains $O(r)$ nodes, from which $O(\sqrt{r})$ are in the separator. Such a separator has $O(n/\sqrt{r})$ nodes.

Let us remark that points on the k-level with equal sets of planes below are connected; see Lemma 1.1. We denote the k-lowest planes of H in vertical line q by k-lowest(q, H).

▶ **Lemma 1.1** (Convexity). *Let q_1 and q_2 denote the vertical lines through some points q_1^* and q_2^* respectively. Let q^* be an arbitrary point on the segment $\overline{q_1^* q_2^*}$, spanned by q_1^* and q_2^* and let q denote the vertical line through the point q^*.*

If k-lowest$(q_1, H) = k$-lowest$(q_2, H) = H'$, then k-lowest$(q, H) = H'$.

Due to space restrictions, we limit our technical discussion to the query correctness. See the full version of this paper for the omitted proofs and algorithms.

2 Optimal k-lowest Queries with Fast Plane Insertion

Our insertion-only data structure is based on the Bentley-Saxe (BS) partial-rebuilding approach for decomposable search problems [3]. There, the goal is to support queries over a limited time period of insertions by successively rebuilding static search structures for those elements that were inserted most recently. For this, a list of subsets that *partition* the semi-dynamic set $H = H_f \cup H_{f-1} \cup \ldots$ is maintained. Let integer $N = |H_f|$ denote the number of planes at time of the last *complete* rebuild of one static structure ($H_i = \emptyset$ for $i < f$). Let $d := \lceil \log_2 N \rceil$, and maintaining set sizes $|H_i| = O(d^i)$ leads to $f = O(\frac{\log n}{\log \log n})$ sets in the partition, and amortized insertion cost $O(fg)$, where g bounds the ratios $P(d^{i+1})/d^i$ of the construction time P, of static structures for H_{i+1}, after d^i insertions.

Static data structures with $O(\log n + k)$ query time can be obtained from the known deterministic $O(n \log n)$ time construction of (a hierarchy of) shallow cuttings in [6]. The $O(n \log n)$ space bound can be reduced to $O(n)$ by on-the-fly recomputation of conflict-lists with a known optimal half-space range-reporting structure [1, 6]. (This idea is credited to Afshani after Corollary 10 in [4].) Running the standard linear time k-Selection algorithm [9, Sec. 1.8] then yields the k planes that are lowest in vertical line q in $O(\log n + k)$ time. Using this optimal static structure, combined with the Heap-Selection method of [8, Thm. 1], immediately yields $O(\log^2 n + k)$ query time for k-lowest planes in the insertion-only setting. This solution is optimal for queries with *large* k.

Thus, the remainder of this section is concerned with the difficult cases of $k = o(\log^2 n)$. Let $\bar{k} := \lceil \log_2^2 N \rceil$ throughout the entire paper. Our data structure contains:

- $H_0, \ldots H_f$: A partition of the semi-dynamic set H. Set H_i has size $O(d^i)$.
- $\hat{H}_0, \ldots \hat{H}_f$: Set $\hat{H}_i$ contains all planes of H_i as well as some from the sets H_j, $j > i$. The added planes serve a similar purpose as samples in classical fractional cascading.
- D_i: A triangulation of the projection of the $\bar{k}$-level of $\hat{H}_i$.
- For any $\Delta \in D_i$, $H(\Delta)$ is the set of $\bar{k}$-lowest planes from $\hat{H}_i$ for any q that intersects Δ.
- $\mathsf{SMPL}_i(\ell)$ is a subset of the triangles of D_i. The planes $\cup_{\Delta \in \mathsf{SMPL}_i(\ell)} H(\Delta)$ are in $\hat{H}_{i-\ell}$.

Description here omits additional search structures attached to each set $\hat{H}_i$, and each triangle $\Delta \in D_i$. The full details are described in Lemma 2.4, which shows that they can be computed in near-linear time. Our query method is a greedy trial-and-error approach, that relies on two functions FIX and SKIP, that we formalized in Lemma 2.1 and Lemma 2.3.

Intuitively, any greedy-step assumes that a "tentative result" for $[0, i]$ is "correct" for even larger prefixes $[0, i + \ell]$ of BS sets. For certain extension lengths ℓ, this assumption is checked "quickly" with an approximate decider[1] SKIP. On rejection, the FIX step then "quickly" computes a new tentative result, and index $i' > i$, such that the result is correct for the prefix $[0, i']$. Transition from i to i' is called a greedy-step, and it involves several SKIP and one FIX calls. This completes the abstract high-level description of our query method, which we formalize in detail below.

Next, we specify what "quickly" means for a greedy-step in the context of BS sets, after which we will specify what "tentative result" and "correct for a prefix" means in the context of our search problem, $\bar{k}$-lowest planes in q.

Doing a greedy-step "quickly": As it turns out for the $\bar{k}$-lowest planes problem, the structures obtained from algorithm CONSTRUCT in Lemma 2.4 will allow performing one greedy-step, from i to i', in $(i' - i)O(\log d)$ time (cf. Lemma 2.5 and Lemma 2.6). That is, the total query time then telescopes to optimal

$$\forall\ i_0 < i_1 < \ldots < i_m\ :\ \sum_{j<m} (i_{j+1} - i_j)O(\log d) = (i_m - i_0)O(\log d) \tag{1}$$

$$= O(f \log d) = O\left(\frac{\log N}{\log d} \log d\right) \tag{2}$$

time, regardless of the number m of greedy-steps. With respect to the $\bar{k}$-lowest planes problem, the proposed greedy trial-and-error approach for queries can be described concisely with the pseudo-code in Figure 2.

```
1  QueryLowest(k, q):
2     Let Δ ∈ D_i be intersected by q, i minimal
3     seq := (Δ)
4     WHILE i < f {
5        j:=0; WHILE SKIP(Δ, q, 2^j) { j++ }
6        (i', Δ') := FIX(Δ, q, 2^j)
7        i := i'; Δ := Δ'; append Δ to seq
8     }
9     append Δ ∈ D_f that intersect q to seq
10    RETURN PRUNE(k, q, seq)
```

Figure 2 Greedy algorithm for k-lowest queries in pseudo-code form. See below for SKIP and FIX.

A naïve implementation for the post-processing PRUNE$(k, \ldots)$ runs in $O(\bar{k} \log_d N)$ time, since each of $O(\log_d N)$ triangles in the sequence has $\bar{k}$ elements in $H(\Delta)$.

In the following lemma, we define the semantics of SKIP(Δ, q, ℓ) and what an evaluation to True implies for $\bar{k}$-lowest$(q, \hat{H}_i \cup \ldots \cup \hat{H}_{i+\ell})$. After this, Lemma 2.3 will then discuss the

[1] Suitable approximate deciders have a one-sided error, i.e. have reliable True answers but may return False even if a tentative result is valid for the prefix.

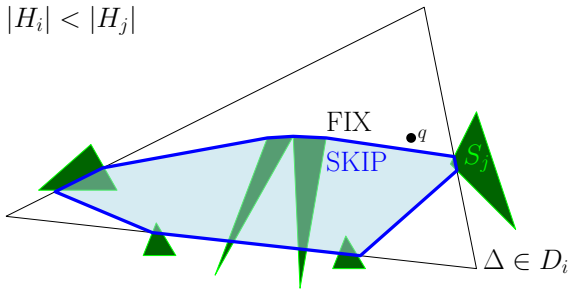


Figure 3 $\Delta \in D_i$ intersecting q (black), $S_j \subseteq \mathbb{R}^2$ of some $j > i$ (green), and $\mathsf{hull}(\Delta \cap S_j)$ (blue).

semantic of FIX and what "tentative result" and "correct for a prefix" means. Discussion of how quickly SKIP and FIX can be computed with suitable data structures is carried out in Lemmas 2.4, 2.5, and 2.6.

▶ **Lemma 2.1** (SKIP-Correctness). *Let indices $i < \ell$, triangle $\Delta \in D_i$ intersect q in point $q^* \in \mathbb{R}^2$, planar region $S_j = \bigcup_{\Delta' \in \mathsf{SMPL}_j(j-i)} \Delta'$ for any $j > i$, and $H(\Delta') \subseteq \hat{H}_i \setminus H_i$ for all triangles $\Delta' \in \mathsf{SMPL}_j(j-i)$.*

Define $\mathsf{SKIP}(\Delta, q, \ell) \iff q^* \in \bigcap_{j=i+1}^{i+\ell} \mathsf{hull}(\Delta \cap S_j)$.

If $\mathsf{SKIP}(\Delta, q, \ell) = \mathsf{True}$, *then* $\bar{k}$*-lowest*$(q, \hat{H}_i) = \bar{k}$*-lowest*$(q, \hat{H}_i \cup \ldots \cup \hat{H}_{i+\ell})$.

Proof. We call degenerate triangles just triangles in the following, as the arguments will hold regardless of corner count. It suffices to show for arbitrary $j > i$, that

$$q^* \in \mathsf{hull}(\Delta \cap S_j) \implies \bar{k}\text{-lowest}(q, \hat{H}_i) = \bar{k}\text{-lowest}(q, \hat{H}_i \cup \hat{H}_j) \quad . \tag{3}$$

Since $\Delta \in D_i$ is the triangle that intersects vertical line q, we have $\bar{k}$-lowest$(q, \hat{H}_i) = H(\Delta)$. See Figure 3.

Consider the set of points on the polyhedral surface $A \subseteq L(\bar{k}, \hat{H}_i \cup \hat{H}_j)$ where the $\bar{k}$-lowest planes coincide with $H(\Delta)$. Let $A^* \subseteq \mathbb{R}^2$ be the orthogonal projection of A. (In other words, the region of queries that may be skipped as their set of $\bar{k}$-lowest planes is equal to $H(\Delta)$.) The implication (3) follows, if we can show that $\mathsf{hull}(\Delta \cap S_j) \subseteq A^*$ is true.

If $\Delta \cap S_j = \emptyset$, then $\mathsf{hull}(\Delta \cap S_j) = \emptyset$ and the implication is true[2]. So let $q' \in \Delta \cap S_j$ be arbitrary. Since $q' \in \Delta'$ of some $\Delta' \in \mathsf{SMPL}_j(j-i)$, we have $\bar{k}$-lowest$(q', \hat{H}_j) = H(\Delta')$ are in the augmented set, i.e. $H(\Delta') \subseteq \hat{H}_i \setminus H_i$. From $q' \in \Delta$, we have that $\bar{k}$-lowest$(q', \hat{H}_i) = H(\Delta)$. Thus, $q' \in A^*$ and $A^* \neq \emptyset \neq A$.

The lemma now follows from A^* being convex, due to A being convex (Lemma 1.1). ◀

In the following, we define the necessary semantic of FIX, "tentative result", and "correct for a prefix", which allows us to establish the correctness of our greedy trial-and-error approach for the $\bar{k}$-lowest planes problem. Establishing runtime efficiency, with suitable data structures, is carried out in Lemmas 2.4, 2.5, and 2.6.

▶ **Definition 2.2** (Tentative Result, Prefix-Validity). A sequence of triangles $(\Delta_{i_0}, \Delta_{i_1}, \ldots, \Delta_{i_m})$ is called a tentative result for prefix $[0, i]$, if $\Delta_{i_j} \in D_{i_j}$ and $0 \leq i_0 < i_1 < \ldots < i_m = i$. Let $\bar{H} = \bigcup_{j=0}^{m} H(\Delta_{i_j})$ denote the planes appearing in the tentative result.

A tentative result is valid for prefix $[0, i]$, if $\bar{k}$-lowest$(q, \bar{H}) = \bar{k}$-lowest$(q, \hat{H}_0 \cup \ldots \cup \hat{H}_i)$.

[2] In other words, SKIP returns False in this case.

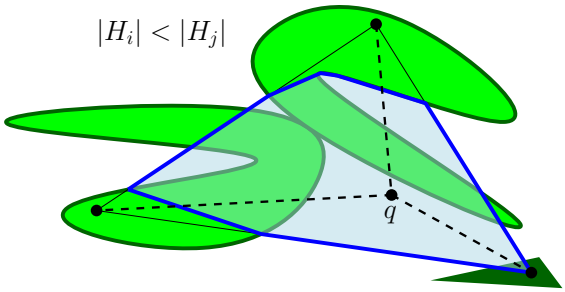


Figure 4 For $\Delta \in D_i$ and $j > i$, any connected component of $D_j \setminus \mathsf{SMPL}_j(j-i)$ containing a corner has $O(d^{16+2(j-i)})$ triangles, and is bounded by $O(d^{8+(j-i)})$ triangles, from $\mathsf{SMPL}_j(j-i)$.

▶ **Lemma 2.3** (FIX-Correctness). *Let $\Delta \in D_i$ be the triangle intersecting q, and $j \geq 0$ the smallest integer with $\mathsf{SKIP}(\Delta, q, 2^j) = \mathsf{False}$.*

Define $\mathsf{FIX}(\Delta, q, 2^j)$ to return pair (i', Δ'), such that triangle $\Delta' \in D_{i'}$ intersects q and index $i' \in [i + 2^j, i + 2^{j+1})$ is the smallest index with $\mathsf{SKIP}(\Delta, q, i') = \mathsf{False}$.

If $(\Delta_{i_j})_{j \leq m}$ is valid for prefix $[0, i]$ and $(i', \Delta') = \mathsf{FIX}(\Delta_{i_m}, q, 2^j)$, then $(\Delta_{i_0}, \ldots, \Delta_{i_m}, \Delta')$ is valid for prefix $[0, i']$.

In particular, defined semantics for SKIP *and* FIX *maintain the prefix-validity throughout any greedy-step of the query algorithm. That is, the final triangle sequence is valid for $[0, f]$.*

Proof. The proof is by induction. The prefix invariant holds for $[0, i]$ for some $i \geq 0$, since the query algorithm always determines the intersected triangle in D_i with minimal index i in the BS partition as the first triangle in a tentative result. Let $\Delta \in D_i$ be the last triangle in the sequence that is valid for $[0, i]$. Apply Lemma 2.1, and use that i' is minimal, and $\Delta' \in D_{i'}$ with $q \cap \Delta' \neq \emptyset$, so true for $[0, i']$. ◀

Next, we discuss the algorithm CONSTRUCT for computing $\hat{H}_i$ and its associated triangulation D_i, including how the triangles $\mathsf{SMPL}_i(j) \subseteq D_i$ are obtained for all possible integers $j \in [1, i]$, and the three kinds of planar point-location structures associated to D_i that we will need to establish that SKIP and FIX can be computed sufficiently fast during queries.

▶ **Lemma 2.4** (Construction). *There is an algorithm $\mathsf{CONSTRUCT}(\{H_0, \ldots H_i\}, \{D_{i+1}, \ldots D_f\})$ that computes a planar triangulation D_i of projected $L(\bar{k}, \hat{H}_i)$, where $|\hat{H}_i| = O(|H_i|)$.*

The triangulation has size $|D_i| = O(|H_i|\bar{k}^{3/2})$, and is augmented with the structures:

1. *Global-Search: A point-location structure for determining a triangle $\Delta \in D_i$.*
2. *Samples: For each integer $\ell \in [1, i]$ a subset $\mathsf{SMPL}_i(\ell) \subseteq D_i$ of triangles, such that every connected component of $D_i \setminus \mathsf{SMPL}_i(\ell)$ has $O(d^{16+2\ell})$ triangles, is adjacent to $O(d^{8+\ell})$ triangles from $\mathsf{SMPL}_i(\ell)$, and the separator contains $|\mathsf{SMPL}_i(\ell)| = O(|D_i|/d^{8+\ell})$ triangles.*
3. *Fix-Search: For each integer $\ell \in [1, i]$ and each connected component of $D_i \setminus \mathsf{SMPL}_i(\ell)$, a point-location structure for the triangles in that component.*
4. *Fix-Pointer: For every $\Delta \in D_i$ and integer $\ell \in (i, f]$, $O(1)$ pointers identifying the connected component of $D_\ell \setminus \mathsf{SMPL}_\ell(\ell - i)$ containing the corners of Δ.*
5. *Skip-Decider: For each $\Delta \in D_i$ and $2^j \in [1, f - i]$ one point-location structure, for the overlay of $\{\mathsf{hull}(\Delta \cap S_{i'}) : i' \in (i, i + 2^j]\}$, with worst-case query time $O(2^j \log d)$.*

Construction of these data structures takes $O(|D_i| \log^3 N)$ time, they occupy $O(f|D_i|)$ space.

It remains to show that the computed structures of D_i allow us to compute SKIP and FIX sufficiently fast for the outlined telescoping argument above.

▶ **Lemma 2.5** (SKIP-Runtime). *Let $j \geq 0$ be the index where $\mathsf{SKIP}(\Delta, q, 2^j)$ returns false. The structures stored with Δ allow to answer the $j+1$ calls of* SKIP *in $O(2^j \log d)$ total time.*

Proof. Recall the construction of D_i computed for each $\Delta \in D_i$ and integer $2^\ell = O(f)$ query structures, that allow to answer if $q^* \in \mathsf{hull}(\Delta \cap S_{i+1} \cap \ldots \cap S_{i+2^\ell})$, and return the smallest integer i' of a $S_{i'}$ where it is not contained. Answering a planar point-location query in the overlay for skip-length 2^ℓ takes $O(2^\ell \log d)$ time. Consequently, answering $j+1$ such queries takes $(2^0 + 2^1 + \ldots + 2^j)O(\log d)$ time using the skip-decider structures. ◀

▶ **Lemma 2.6** (FIX-Runtime). *Finding the smallest integer $i' \in [i + 2^j, i + 2^{j+1})$ where* SKIP *is false on prefix $[0, i']$ can be implemented in $(i' - i)O(\log d)$ time. Moreover, obtaining the triangle $\Delta' \in D_{i'}$ that intersects q takes $(i' - i)O(\log d)$ time.*

Proof. Recall that Lemma 2.4 precomputed a point-location structure for the planar subdivision that is the overlay of the subdivisions in $\{\mathsf{hull}(\Delta \cap S_{i'}) \ : \ i' \in [i + 2^j, i + 2^{j+1})\}$. As each region of that subdivision was labeled with the largest integer i'' where its points are contained in all $\mathsf{hull}(\Delta \cap S_{i''})$, we obtain the smallest i' in time $O(2^j \log d) = (i' - i)O(\log d)$, since $2^j = O(i' - i)$. Recall that corner-component membership information was precomputed in construction of D_i for all $i' > i$. For any given $i' > i$ and connected component of $D_{i'} \setminus \mathsf{SMPL}_{i'}(i' - i)$, a point-location query takes $O(\log d^{16+2(i'-i)}) = (i' - i)O(\log d)$ time, and there are $O(1)$ connected components to query. ◀

Note that both time bounds, of Lemmas 2.5 and 2.6, are $(i' - i)O(\log d)$, as desired.

▶ **Theorem 2.7.** *Maintaining a query structure for a semi-dynamic set H of planes under insertions such that for given vertical line q we can report a sequence of at most $f = O(\frac{\log |H|}{\log \log |H|})$ triangles such that $\bar{k}$-lowest$(q, H) = \bar{k}$-lowest$(q, \bar{H})$ is possible with $O(fd \log^6 |H|) = O(\log^8 |H|)$ amortized insertion time. The structure has size $O(|H| \frac{\log^4 |H|}{\log \log |H|})$.*

The worst-case query time to obtain the triangle sequence is $O(\log |H|)$.

References

1 Peyman Afshani and Timothy M. Chan. Optimal halfspace range reporting in three dimensions. In *20th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, page 180–186, 2009. doi:10.1137/1.9781611973068.21.

2 Pankaj K. Agarwal, Boris Aronov, Timothy M. Chan, and Micha Sharir. On Levels in Arrangements of Lines, Segments, Planes, and Triangles. *Discret. Comput. Geom.*, 19(3):315–331, 1998. doi:10.1007/PL00009348.

3 Jon Louis Bentley and James B. Saxe. Decomposable searching problems I: static-to-dynamic transformation. *Journal of Algorithms*, 1(4):301–358, 1980. doi:10.1016/0196-6774(80)90015-2.

4 Timothy M. Chan. Dynamic geometric data structures via shallow cuttings. In *35th International Symposium on Computational Geometry (SoCG)*, pages 24:1–24:13, 2019. URL: https://doi.org/10.4230/LIPIcs.SoCG.2019.24, doi:10.4230/LIPICS.SOCG.2019.24.

5 Timothy M. Chan, Pingan Cheng, and Da Wei Zheng. An Optimal Algorithm for Higher-Order Voronoi Diagrams in the Plane: The Usefulness of Nondeterminism. In *35th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, page 4451–4463, 2024. doi:10.1137/1.9781611977912.156.

6 Timothy M. Chan and Konstantinos Tsakalidis. Optimal Deterministic Algorithms for 2-d and 3-d Shallow Cuttings. *Discret. Comput. Geom.*, 56(4):866–881, 2016. URL: https://doi.org/10.1007/s00454-016-9784-4, doi:10.1007/S00454-016-9784-4.

7 Mark de Berg, Otfried Cheong, Marc J. van Kreveld, and Mark H. Overmars. *Computational geometry: algorithms and applications, 3rd Edition.* 2008. doi:10.1007/978-3-540-77974-2.

8 Sarita de Berg and Frank Staals. Dynamic data structures for k-nearest neighbor queries. *Comput. Geom.*, 111:101976, 2023. URL: https://doi.org/10.1016/j.comgeo.2022.101976, doi:10.1016/J.COMGEO.2022.101976.

9 Jeff Erickson. *Algorithms.* 2019. URL: http://archive.org/details/Algorithms-Jeff-Erickson.

10 Greg N. Frederickson. An Optimal Algorithm for Selection in a Min-Heap. *Inf. Comput.*, 104(2):197–214, 1993. URL: https://doi.org/10.1006/inco.1993.1030, doi:10.1006/INCO.1993.1030.

11 Michael T. Goodrich. Planar Separators and Parallel Polygon Triangulation. *J. Comput. Syst. Sci.*, 51(3):374–389, 1995. URL: https://doi.org/10.1006/jcss.1995.1076, doi:10.1006/JCSS.1995.1076.

12 Philip N. Klein, Shay Mozes, and Christian Sommer. Structured recursive separator decompositions for planar graphs in linear time. In *Symposium on Theory of Computing Conference (STOC)*, page 505–514, 2013. doi:10.1145/2488608.2488672.

13 Micha Sharir, Shakhar Smorodinsky, and Gábor Tardos. An improved bound for k-sets in three dimensions. In *16th Symposium on Computational Geometry (SoCG)*, page 43–49, 2000. doi:10.1145/336154.336173.

On rectilinear drawings of the hypercube*

Todor Antić[1], Niloufar Fuladi[2], Anna Margarethe Limbach[2], and Pavel Valtr[1]

1 **Department of Applied Mathematics, Charles University, Prague, Czechia**
{todor,valtr}@kam.mff.cuni.cz
2 **Computer Science Institute of Charles University, Prague, Czechia**
{nfuladi,limbach}@iuuk.mff.cuni.cz

Abstract

We study the existence of plane substructures in rectilinear drawings of the hypercube graph Q_d. We construct drawings of Q_d which contain no plane subgraph with more than $2d-2$ edges, no plane path with more than $2d-3$ edges, and no plane matching of size more than $2d-4$. On the other hand, we prove that every rectilinear drawing of Q_d with vertices in convex position contains a plane path of length d (if d is odd) or $d-1$ (if d is even). We also prove that if a plane graph G is contained in every drawing of Q_d for a sufficiently large d, then G is necessarily a forest of caterpillars. Lastly, we give a short proof of a generalization of a result by Alpert et al. on the maximum rectilinear crossing number of Q_d.

1 Introduction

Among the first graphs introduced in any introductory graph theory course are complete graphs, complete bipartite graphs and d-dimensional hypercube graphs (which we refer to simply as d-cubes). It is therefore not surprising that drawings of complete [2, 22, 20, 27, 40, 6, 7, 8] and complete bipartite [11, 14, 1, 29] graphs are a frequent topic of research in graph drawing, with many results and interesting problems. However, what one might find surprising is, that drawings of d-cubes have not gained such interest. As far as we are aware, except for various versions of crossing number problems [4, 23, 39, 15, 36, 16, 24, 21], drawings of d-cubes were not studied otherwise.

The d-cube Q_d is a graph with 2^d vertices labeled by binary strings of length d in which two vertices are connected by an edge if their labels differ exactly in one digit. A *rectilinear drawing* is a drawing of a graph with vertices represented by points in the plane and edges represented by line segments between the vertices. We say that a drawing is *plane* if no two edges cross. We call a drawing *convex-geometric* if it is rectilinear and its vertices are in convex position. We call a drawing *simple* if edges are represented by simple curves with no self intersections, and every two curves intersect in at most one point which is either a common vertex or a proper crossing in the interior of both edges (no tangential "touches").

We investigate the problem of finding plane subgraphs in (rectilinear) drawings of Q_d. This problem has been considered for complete and complete bipartite graphs extensively. If we are trying to find a single plane Hamiltonian path (or cycle), the problem is almost trivial for rectilinear drawings of K_n. However, if we relax the setting slightly and consider simple drawings, the problem becomes much harder. In fact, the existence of a plane

* The research by Todor Antić and Pavel Valtr was supported by the Czech Science Foundation under the grant agreement no. 23-04949X. The research by Niloufar Fuladi was supported by the Czech Science Foundation under the grant agreement no. 25-16627S and the research by Anna Margareth Limbach was supported by ERC-CZ LL2328.

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

Hamiltonian path in every simple drawing of K_n was conjectured in 1988 by Rafla [34], and even after almost 40 years of effort by multiple authors [2, 9, 33, 38, 17, 18, 19, 32, 25], the best known result is that every simple drawing of K_n contains a plane path of length $O(\log(n)/\log\log(n))$ [2]. We mention that finding large plane matchings, cycles, and trees has also been considered in the literature [32, 18, 2]. In the case of rectilinear drawings, since finding one plane Hamiltonian path is easy, finding multiple such paths has been considered [26, 5].

Finding plane spanning paths in rectilinear drawings of complete bipartite graphs has proven to be an interesting problem (without considering more relaxed drawings). It is known that there are drawings with no plane path of length more than $(4-2\sqrt{2})n+o(n)$ [13], where n is the total number of vertices. On the other hand, it is known that a plane path of length $\frac{n}{2}$ always exists [37]. Since plane Hamiltonian paths do not exist in every drawing of the complete bipartite graph, and while several authors gave different sufficient and necessary conditions for a drawing to contain a plane Hamiltonian path [12, 31, 13, 35, 28, 37], but a complete characterizations still remains unknown.

Our contribution. We construct drawings of Q_d that contain no plane subgraph with more than $2d-2$ edges for all sufficiently large d, no plane path of length more than $2d-3$ and no plane matching of size more than $2d-4$. On the other hand, we prove that every rectilinear drawing of Q_d with vertices in convex position contains a plane path of length $d-1$ (if d is odd) or d (if d is even). Additionally, we prove that if a graph G is a plane subgraph of every drawing of a d-cube for a sufficiently large d, then G is necessarily a forest of caterpillars.

We also use a variation of our construction to revisit the problem of computing the *maximum rectilinear crossing number* of a d-cube. The maximum rectilinear crossing number of a graph G (denoted by $\mathrm{CR}_{\max}(G)$) is the maximum number of crossings, taken over all rectilinear drawings of G. This parameter was studied for many graph classes, such as cycles [10], complete graphs [22] and k-regular graphs [3], among others. Alpert et al. [4] investigated this question for Q_d. They proved that $\mathrm{CR}_{\max}(Q_d) \geq 2^{d-2}(2^{d-1}(d^2 - 2d+3) - d^2 - 1)$ and conjectured that this is tight. We give a simple proof of a more general result. In particular, we give a formula for computing the crossing numbers for a class of drawings that includes the drawing constructed by Alpert et al. Interestingly, even though our construction is different from theirs, the resulting drawings are *weakly isomorphic*, i.e. the same pairs of edges cross in both drawings.

1.1 Preliminaries

Let $\mathcal{G}$ be a convex-geometric drawing of a graph G. The *combinatorial length*, or *length* for short, of an edge $e = uv$ in $\mathcal{G}$ is the graph distance between u and v in the cycle bounding the convex hull of the vertices of $\mathcal{G}$ (the edges of this cycle are not necessarily edges of the drawing); see Figure 1. Assume that $G = Q_d$ for some d. The possible lengths of edges are $1, 2, \ldots, 2^{d-1}$. To a vertex $v \in \mathcal{G}$ we associate a vector $\hat{v} = (\ell_1, \ell_2, \ldots, \ell_d)$ where $\ell_1 \geq \ell_2 \geq \cdots \geq \ell_d$ are the lengths of edges adjacent to v. If $\hat{v} = \hat{u}$ for every two vertices $u, v \in \mathcal{G}$ we say that $\mathcal{G}$ is *length-regular* and we call $\hat{v}$ the *length profile* of $\mathcal{G}$.

For a convex geometric drawing $\mathcal{G}$ and two vertices $v, w \in \mathcal{G}$ we write $[v, w]$ for the set of vertices of $\mathcal{G}$ which are seen when traversing the convex hull of the vertices of $\mathcal{G}$ clockwise from v to w, we call this an *interval*. We write (v, w) for $[v, w] \setminus \{v, w\}$. Throughout the paper, we slightly abuse notation and identify vertices of a graph G with their representation in a drawing of G. As mentioned before, the d-cube Q_d is a graph whose vertices are binary strings of length d. We denote the vertices of Q_d by $\overline{x}$, where x is a string in $\{0, 1\}^d$.

All proofs are moved to the appendix, and marked with clickable (*).

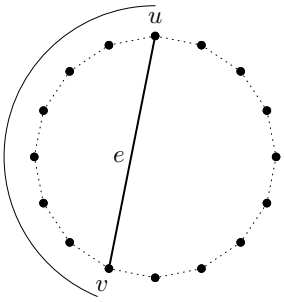


Figure 1 The length of e is 7: the number of segments covered by the depicted arc.

2 Plane subgraphs in convex-geometric drawings of the d-cube

We start by giving a lower bound on the length of a plane path which is present in any convex geometric drawing of the d-cube. The following result is an easy consequence of a theorem by Perles, mentioned in [30, p. 292], which we state and prove in Appendix **??**.

▶ **Theorem 2.1** (⋆). *For all $d \geq 2$, any convex-geometric drawing of Q_d contains a plane path of length at least d if d is odd, and $d-1$ if d is even.*

We now define a particular convex-geometric drawing of Q_d which we use to find upper bounds for maximum sizes of various plane subgraphs in convex-geometric drawings of Q_d. In our construction, the vertices of Q_d are placed on a circle C obtained from the interval $[0,1] \subset \mathbb{R}$ by identifying the endpoints. With slight abuse of notation, we identify the points of C with points in $[0,1)$. Now, we define a drawing $\mathcal{H}_2$ of Q_2 by mapping the vertices $\overline{00} \to 0$, $\overline{01} \to \frac{1}{4}$, $\overline{11} \to \frac{1}{2}$, $\overline{10} \to \frac{3}{4}$. For $d \geq 3$, we construct the drawing $\mathcal{H}_d$ by taking two copies $\mathcal{H}', \mathcal{H}''$, of $\mathcal{H}_{d-1}$ and rotating $\mathcal{H}''$ so that vertex $\overline{00\ldots0} \in \mathcal{H}''$ is placed on the point $\frac{1}{2} - \frac{1}{2^d} \in C$, in other words, we rotate the drawing by slightly less than 180°. We then connect every vertex in $\mathcal{H}'$ to its copy in $\mathcal{H}''$. Now, we map vertices of Q_d to $\mathcal{H}_d$ by mapping a vertex $\overline{0x}$ to the copy of x in $\mathcal{H}'$ and $\overline{1x}$ to the copy of x in $\mathcal{H}''$. See Figure 2 for the construction of $\mathcal{H}_d$.

Observe that, by definition, $\mathcal{H}_d$ is a length-regular drawing with length profile $(2^{d-1}-1, 2^{d-1}-2, 2^{d-2}-4, \ldots, 2^{d-1}-2^{d-2}, 2^{d-1}-2^{d-2})$. For a vertex $v \in \mathcal{H}_d$, let e_1^v, e_2^v, …, e_{d-2}^v be the edges adjacent to v with length strictly larger than $2^{d-1}-2^{d-2}$, sorted by length (with e_1^v being the longest) and let p_v be the line through v and the center of C. We associate with each such edge e_i^v a *direction* $d(e_i^v) = \pm 1$ where $d(e_i^v) = +1(-1)$ means that the other endpoint of e_i^v lies to the left (right) of p_v when viewed from v. We call the tuple $(d(e_1^v), d(e_2^v), \ldots, d(e_{d-2}^v))$ the *length-rotation* of v. Note that there are 2^{d-2} length-rotations that can appear in $\mathcal{H}_d$, we start by observing that all of them indeed appear in $\mathcal{H}_d$.

▶ **Lemma 2.2** (⋆). *For $d \geq 2$ and $\overline{v} \in \{+1,-1\}^{d-2}$, there are exactly two pairs of antipodal vertices in $\mathcal{H}_d$ with length-rotation $\overline{v}$. Thus, every possible length-rotation exists 4 times.*

Lastly, we observe that the length-rotations of adjacent vertices in $\mathcal{H}_d$ agree up to the coordinate corresponding to the edge between them.

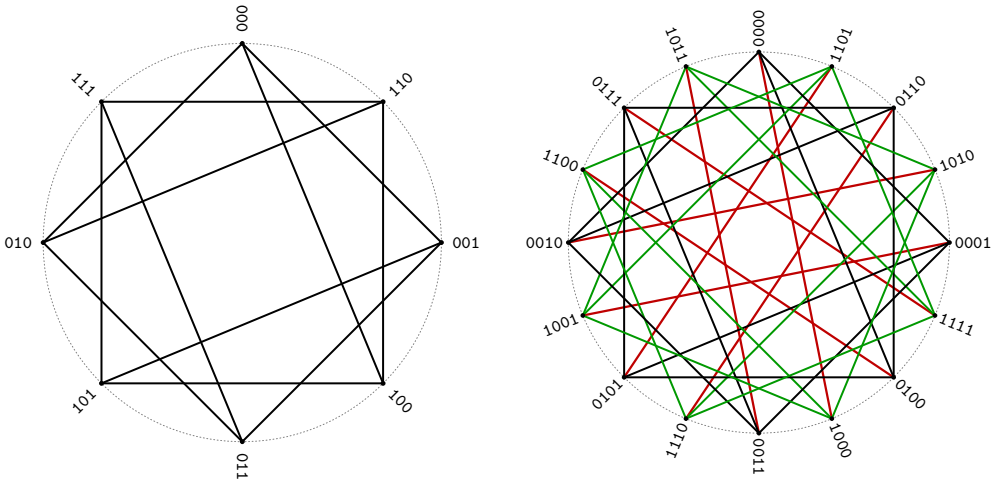


Figure 2 The construction of $\mathcal{H}_4$ (right) from $\mathcal{H}_3$ (left). In the right picture, the edges in $\mathcal{H}'$ and $\mathcal{H}''$ are depicted in black and green, respectively, and the red edges depict the ones between these two copies of $\mathcal{H}_3$.

▶ **Lemma 2.3** ($\star$). *Let $d \geq 2$ and let $e = uv$ be an edge of length $2^{d-1} - 2^m$ in $\mathcal{H}_d$ for some $m \in \{0, 1, \ldots, d-2\}$. Then if $\overline{u}$ and $\overline{v}$ are length-rotations of u and v, respectively, $\overline{u}_i = \overline{v}_i$ for all $i \in \{m+2, m+3, \ldots, d-2\}$.*

From now until the end of this section, we present several results about plane subgraphs of $\mathcal{H}_d$. We start by discussing the length of plane subpaths of $\mathcal{H}_d$.

▶ **Theorem 2.4** ($\star$). *Let $d \geq 4$, then $\mathcal{H}_d$ contains a plane path of length $2d-3$.*

Figure 3, depicts a plane path of length 7 in $\mathcal{H}_5$ constructed in the proof of Theorem 2.4.

Let us now consider an alternative construction of $\mathcal{H}_d$ which will be useful for proving our next result. Let $\mathcal{H}'_2 = \mathcal{H}_2$. For $d \geq 3$, we obtain $\mathcal{H}'_d$ from $\mathcal{H}'_{d-1}$ as follows. Let v_1, v_2, $\ldots$, $v_{2^{d-1}}$ be vertices of $\mathcal{H}'_{d-1}$ ordered clockwise around C. Now for each vertex v_i add another vertex u_i between v_i and v_{i+1}. Now for each $i \neq j$ add an edge u_iu_j if and only if edge v_iv_j was present in $\mathcal{H}'_{d-1}$, note that if present, the two edges u_iu_j and v_iv_j necessarily cross. Lastly, for each i, we add two parallel edges $v_iu_{i+2^{d-2}}$ and $v_{i+2^{d-2}}u_i$ (note that we consider indices modulo 2^{d-1}), see Figure 4. We call the parallel edges added in the last step of construction *parallel pairs*. It is clear from definition that if an edge e belongs to a parallel pair, then it is an edge of length $2^{d-1} - 1$ and it crosses every edge belonging to a parallel pair different then itself. One can check that $\mathcal{H}'_d$ is weakly isomorphic to $\mathcal{H}_d$.

To show that Theorem 2.4 is in fact optimal, we prove the following more general result.

▶ **Theorem 2.5** ($\star$). *Let $d \geq 3$ and G be a plane subgraph of $\mathcal{H}_d$. Then $|E(G)| \leq 2d-2$.*

We get the following as a consequence of Theorem 2.5.

▶ **Corollary 2.6** ($\star$). *Let $d \geq 3$ and P be a plane path in $\mathcal{H}_d$, then $|E(P)| \leq 2d-3$.*

▶ **Theorem 2.7** ($\star$). *Let $d \geq 4$, and let M be a plane matching in $\mathcal{H}_d$, then $|E(M)| \leq 2d-4$, and this is tight.*

Lastly, we give a necessary condition for an abstract graph to be embeddable into every convex geometric drawing of the d-cube (for a sufficiently large d).

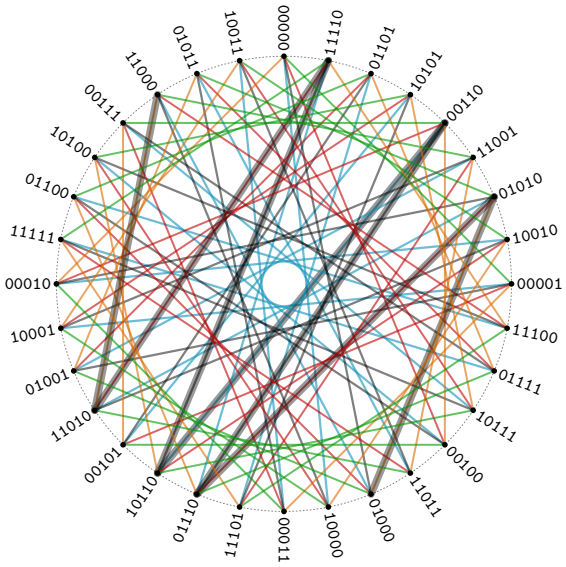


Figure 3 A plane path of length 7 in $\mathcal{H}_5$ is highlighted.

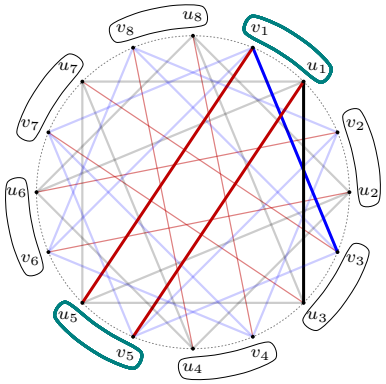


Figure 4 The alternative construction of $\mathcal{H}_d$.

▶ **Theorem 2.8** ($\star$). *If there is a $d \geq 2$ such that every rectilinear drawing of Q_d contains a fixed graph G as a plane subgraph, then G is a forest of caterpillars.*

3 Maximum Rectilinear Crossing number of the d-cube

We now define another drawing of Q_d, in a similar way as we defined $\mathcal{H}_d$. Let C as before and let $\mathcal{R}_2$ be a drawing of Q_2 defined by mapping the vertices of Q_2 as follows: $\overline{00} \to 0$, $\overline{01} \to \frac{1}{2}$, $\overline{11} \to \frac{1}{4}$, $\overline{10} \to \frac{3}{4}$. For $d \geq 3$, we obtain $\mathcal{R}_d$ from $\mathcal{R}_{d-1}$ in the same way we obtained $\mathcal{H}_d$ from $\mathcal{H}_{d-1}$; see Figure 5. As mentioned before, Alpert et al., [4] constructed a drawing weakly isomorphic to $\mathcal{R}_d$ and conjectured that this drawing achieves $CR_{\max}(Q_d)$. We prove the following more general result.

▶ **Theorem 3.1** ($\star$). *Let $d \geq 2$, then every length-regular drawing of Q_d with length profile $(\ell_1, \ell_2, \ldots, \ell_d)$ has exactly $2^{d-1} \sum_{i=1}^{d} (\ell_i - 1)\left(i - \frac{1}{2}\right)$ crossings.*

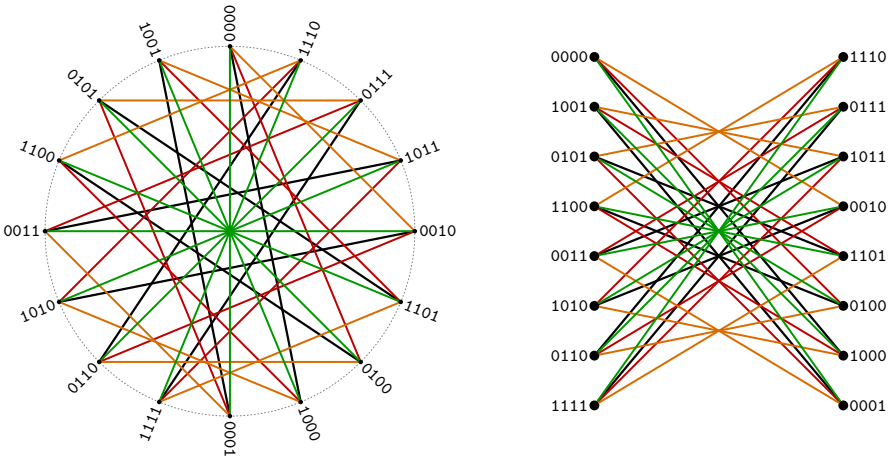


Figure 5 Left: $\mathcal{R}_4$; Right: The drawing from the construction of Alpert et al. for Q_4.

▶ **Corollary 3.2.** *As the drawing $\mathcal{R}_d$ is length-regular with lengths $\ell_1 = 2^{d-1}$ and $\ell_i = 2^{d-1} - 2^{i-2}$ for $2 \leq i \leq d$, the maximum rectilinear crossing number $CR_{\max}(Q_d)$ is at least $2^{d-2}(2^{d-1}(d^2 - 2d + 3) - d^2 - 1)$.*

4 Discussion and Open Problems

To the best of our knowledge, we were the first to investigate existence of plane substructures in drawings of Q_d. While we gave partial or full answer to some questions, many still remain.

Firstly, our study was mostly concerned with convex-geometric drawings of Q_d, and we did not investigate general rectilinear or simple drawings of Q_d. Investigating these problems in these settings may pose some interesting challenges. We attempted to find a counterpart to Theorem 2.1 for general rectilinear drawings. However, the bound that we obtain is considerably worse than that obtained from Theorem 2.1 for convex-geometric drawings.

▶ **Proposition 4.1** ($\star$). *Any rectilinear drawing of Q_d, for $d \geq 3$, contains a plane path of length at least 4.*

Note that the upper bound on the length of the longest plane path from Theorem 2.4 is still valid for general rectilinear drawings, so the gap between the lower and upper bounds is very large. It would be interesting to improve any of these bounds.

For simple drawings, we know even less. We were able to construct a drawing of Q_3 that does not contain a plane path of length more than 3. We present this drawing in Figure 6.

▶ **Proposition 4.2** ($\star$). *The drawing of Q_3 presented in Figure 6 does not contain a plane path of length 4.*

It is possible that our construction could be extended to higher dimensions, which would highlight the difference between rectilinear and simple drawings of Q_d.

Another intriguing problem would be to close the gap between the bounds of Theorems 2.1 and 2.5. Currently, we know that every convex-geometric drawing of Q_d contains a plane path of length at least $d-1$ and that some drawings ($\mathcal{H}_d$) do not contain plane paths of length larger than $2d-3$. We are not sure which (if any) of these bounds is tight.

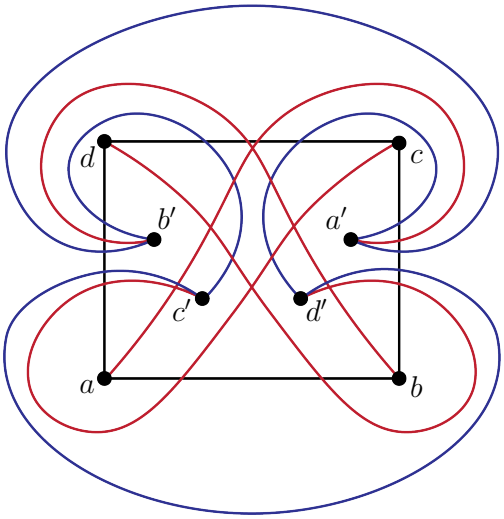


Figure 6 Simple drawing of Q_3 with no plane path of length 4.

Lastly, we mention the conjecture of Alpert et al. [4], which says that the maximum crossing number $\mathrm{CR}_{\max}$ of Q_d is obtained by the drawing $\mathcal{R}_d$. Since this conjecture seems to be hard to solve, wee think that verifying it for the more restricted classes of convex-geometric or even length-regular convex-geometric drawings of Q_d might be more approachable. A more ambitious direction might be to consider the maximum crossing number of Q_d for general simple drawings.

References

1 Oswin Aichholzer, Man-Kwun Chiu, Hung P. Hoang, Michael Hoffmann, Jan Kynčl, Yannic Maus, Birgit Vogtenhuber, and Alexandra Weinberger. Drawings of complete multipartite graphs up to triangle flips. volume 258, pages 6:1–6:16. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023. URL: https://drops.dagstuhl.de/entities/document/10.4230/LIPIcs.SoCG.2023.6, doi:10.4230/LIPICS.SOCG.2023.6.

2 Oswin Aichholzer, Alfredo García, Javier Tejel, Birgit Vogtenhuber, and Alexandra Weinberger. Twisted ways to find plane structures in simple drawings of complete graphs. *Discrete & Computational Geometry*, 71(1):40–66, January 2024. doi:10.1007/s00454-023-00610-0.

3 Matthew Alpert, Elie Feder, and Heiko Harborth. The maximum of the maximum rectilinear crossing numbers of d-regular graphs of order n. *Electron. J. Comb.*, 16, 2008. URL: https://api.semanticscholar.org/CorpusID:9548696.

4 Matthew Alpert, Elie Feder, Heiko Harborth, and Sheldon Klein. The maximum rectilinear crossing number of the n dimensional cube graph. In *Proceedings of the Fortieth Southeastern International Conference on Combinatorics, Graph Theory and Computing*, volume 195, pages 147–158, 2009.

5 Todor Antić, Aleksa Džuklevski, Jiří Fiala, Jan Kratochvíl, Giuseppe Liotta, Morteza Saghafian, Maria Saumell, and Johannes Zink. *Edge-Constrained Hamiltonian Paths on a Point Set*, page 532–546. Springer Nature Switzerland, 2026. URL: http://dx.doi.org/10.1007/978-3-032-17801-5_39, doi:10.1007/978-3-032-17801-5_39.

6 Alan Arroyo, Dan McQuillan, R. Bruce Richter, Gelasio Salazar, and Matthew Sullivan. Drawings of complete graphs in the projective plane. *Journal of Graph Theory*, 97(3):426–440, March 2021. URL: http://dx.doi.org/10.1002/jgt.22665, doi:10.1002/jgt.22665.

7 Alan Arroyo, R. Bruce Richter, and Matthew Sunohara. Extending drawings of complete graphs into arrangements of pseudocircles. *SIAM Journal on Discrete Mathematics*, 35(2):1050–1076, January 2021. URL: http://dx.doi.org/10.1137/20M1313234, doi:10.1137/20m1313234.

8 Martin Balko, Anna Brötzner, Fabian Klute, and Josef Tkadlec. Faces in rectilinear drawings of complete graphs. *European Journal of Combinatorics*, 130:104217, December 2025. URL: http://dx.doi.org/10.1016/j.ejc.2025.104217, doi:10.1016/j.ejc.2025.104217.

9 Helena Bergold, Stefan Felsner, Meghana M. Reddy, Joachim Orthaber, and Manfred Scheucher. Plane hamiltonian cycles in convex drawings. *Discrete & Computational Geometry*, July 2025. URL: http://dx.doi.org/10.1007/s00454-025-00752-3, doi:10.1007/s00454-025-00752-3.

10 Jens-P. Bode, Elie Feder, Heiko Harborth, David Horowitz, and Tamar Lichter. Extremal values of the maximum rectilinear crossing number of cycles with diagonals. *Congr. Numer.*, 220:33–48, 2014.

11 Jean Cardinal and Stefan Felsner. Topological drawings of complete bipartite graphs. *Journal of Computational Geometry*, page Vol. 9 No. 1 (2018), 2018. URL: https://jocg.org/index.php/jocg/article/view/3053, doi:10.20382/JOCG.V9I1A7.

12 Josef Cibulka, Jan Kynčl, Viola Mészáros, Rudolf Stolař, and Pavel Valtr. *Hamiltonian Alternating Paths on Bicolored Double-Chains*, page 181–192. Springer Berlin Heidelberg, 2009. URL: http://dx.doi.org/10.1007/978-3-642-00219-9_18, doi:10.1007/978-3-642-00219-9_18.

13 Endre Csóka, Zoltán L. Blázsik, Zoltán Király, and Dániel Lenger. Upper bounds for the necklace folding problems. *Journal of Combinatorial Theory, Series B*, 157:123–143,

November 2022. URL: http://dx.doi.org/10.1016/j.jctb.2022.05.012, doi:10.1016/j.jctb.2022.05.012.

14 Etienne de Klerk, Dmitrii V. Pasechnik, and Gelasio Salazar. Book drawings of complete bipartite graphs. *Discrete Applied Mathematics*, 167:80–93, April 2014. URL: http://dx.doi.org/10.1016/j.dam.2013.11.001, doi:10.1016/j.dam.2013.11.001.

15 Luerbio Faria and Celina Miraglia Herrera de Figueiredo. On Eggleton and Guy's conjectured upper bound for the crossing number of the n-cube. *Math. Slovaca*, 50(3):271–287, 2000.

16 Luerbio Faria, Celina Miraglia Herrera de Figueiredo, Ondrej Sýkora, and Imrich Vrt'o. An improved upper bound on the crossing number of the hypercube. *Journal of Graph Theory*, 59(2):145–161, June 2008. URL: http://dx.doi.org/10.1002/jgt.20330, doi:10.1002/jgt.20330.

17 Radoslav Fulek. Estimating the number of disjoint edges in simple topological graphs via cylindrical drawings. *SIAM Journal on Discrete Mathematics*, 28(1):116–121, January 2014. URL: http://dx.doi.org/10.1137/130925554, doi:10.1137/130925554.

18 Alfredo García, Javier Tejel, Birgit Vogtenhuber, and Alexandra Weinberger. *Empty Triangles in Generalized Twisted Drawings of K_n*, page 40–48. Springer International Publishing, 2023. URL: http://dx.doi.org/10.1007/978-3-031-22203-0_4, doi:10.1007/978-3-031-22203-0_4.

19 Alfredo García Olaverri, Javier Tejel Altarriba, and Alexander Pilz. On plane subgraphs of complete topological drawings. *Ars Mathematica Contemporanea*, 20(1):69–87, August 2021. URL: http://dx.doi.org/10.26493/1855-3974.2226.e93, doi:10.26493/1855-3974.2226.e93.

20 Emeric Gioan. Complete graph drawings up to triangle mutations. *Discrete & Computational Geometry*, 67(4):985–1022, March 2022. URL: http://dx.doi.org/10.1007/s00454-021-00339-8, doi:10.1007/s00454-021-00339-8.

21 Richard H. Hammack and Paul C. Kainen. A new view of hypercube genus. *The American Mathematical Monthly*, 128(4):pp. 352–359, 2021. URL: https://www.jstor.org/stable/48662634.

22 Heiko Harborth and Christian Thürmann. Maximum rectilinear crossing number in drawings of the complete graph with a given convex hull. *Congr. Numer.*, 221:121–127, 2014.

23 Paul C Kainen. A lower bound for crossing numbers of graphs with applications to K_n, $K_{p,q}$, and $Q(d)$. *Journal of Combinatorial Theory, Series B*, 12(3):287–298, June 1972. URL: http://dx.doi.org/10.1016/0095-8956(72)90042-1, doi:10.1016/0095-8956(72)90042-1.

24 Paul C. Kainen. Skewness, crossing number and Euler's bound for graphs on surfaces, 2025. URL: https://arxiv.org/abs/2501.02400, doi:10.48550/ARXIV.2501.02400.

25 Balázs Keszegh, Andrew Suk, Gábor Tardos, and Ji Zeng. Unavoidable patterns and plane paths in dense topological graphs, 2025. URL: https://arxiv.org/abs/2512.04795, doi:10.48550/ARXIV.2512.04795.

26 P. Kindermann, J. Kratochvíl, G. Liotta, and P. Valtr. *Three Edge-Disjoint Plane Spanning Paths in a Point Set*, page 323–338. Springer Nature Switzerland, 2023. URL: http://dx.doi.org/10.1007/978-3-031-49272-3_22, doi:10.1007/978-3-031-49272-3_22.

27 Jan Kynčl. Enumeration of simple complete topological graphs. *European Journal of Combinatorics*, 30(7):1676–1685, October 2009. URL: http://dx.doi.org/10.1016/j.ejc.2009.03.005, doi:10.1016/j.ejc.2009.03.005.

28 Jan Kynčl, János Pach, and Géza Tóth. Long alternating paths in bicolored point sets. *Discrete Mathematics*, 308(19):4315–4321, October 2008. URL: http://dx.doi.org/10.1016/j.disc.2007.08.013, doi:10.1016/j.disc.2007.08.013.

29 William J. Lenhart and Giuseppe Liotta. *Mutual Witness Gabriel Drawings of Complete Bipartite Graphs*, page 25–39. Springer International Publishing, 2023. URL: http://dx.doi.org/10.1007/978-3-031-22203-0_3, doi:10.1007/978-3-031-22203-0_3.

30 W. Moser and J. Pach. Geometric graphs. In *New Trends in Discrete and Computational Geometry (J. Pach, ed.)*. Springer, New York, 1993.

31 Wolfgang Mulzer and Pavel Valtr. Long alternating paths exist. volume 164, pages 57:1–57:16. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020. URL: https://drops.dagstuhl.de/entities/document/10.4230/LIPIcs.SoCG.2020.57, doi:10.4230/LIPICS.SOCG.2020.57.

32 János Pach, Jozsef Sólymosi, and Géza Tóth. Unavoidable configurations in complete topological graphs. *Discrete and Computational Geometry*, 30(2):311–320, August 2003. URL: http://dx.doi.org/10.1007/s00454-003-0012-9, doi:10.1007/s00454-003-0012-9.

33 János Pach and Géza Tóth. *Disjoint Edges in Topological Graphs*, page 133–140. Springer Berlin Heidelberg, 2005. URL: http://dx.doi.org/10.1007/978-3-540-30540-8_15, doi:10.1007/978-3-540-30540-8_15.

34 Nabil H. Rafla. *The good drawings D_n of the complete graph K_n*. PhD thesis, McGill University, 1988.

35 Marco Ricci, Jonathan Rollin, André Schulz, and Alexandra Weinberger. On plane cycles in geometric multipartite graphs, 2025. URL: https://arxiv.org/abs/2506.20421, doi:10.48550/ARXIV.2506.20421.

36 Farhad Shahrokhi, Ondrej Sykora, László A. Székely, and Imrich Vrt'o. *Bipartite crossing numbers of meshes and hypercubes*, page 37–46. Springer Berlin Heidelberg, 1997. URL: http://dx.doi.org/10.1007/3-540-63938-1_48, doi:10.1007/3-540-63938-1_48.

37 Jan Soukup. Bicolored point sets admitting non-crossing alternating hamiltonian paths, 2024. URL: https://arxiv.org/abs/2404.06105, doi:10.48550/ARXIV.2404.06105.

38 Andrew Suk and Ji Zeng. *Unavoidable Patterns in Complete Simple Topological Graphs*, page 3–15. Springer International Publishing, 2023. URL: http://dx.doi.org/10.1007/978-3-031-22203-0_1, doi:10.1007/978-3-031-22203-0_1.

39 Ondrej Sýkora and Imrich Vrťo. *On the crossing number of the hypercube and the cube connected cycles*, page 214–218. Springer Berlin Heidelberg, 1992. URL: http://dx.doi.org/10.1007/3-540-55121-2_21, doi:10.1007/3-540-55121-2_21.

40 Bernardo M. Ábrego, Oswin Aichholzer, Silvia Fernández-Merchant, Pedro Ramos, and Gelasio Salazar. The 2-page crossing number of K_n. *Discrete & Computational Geometry*, 49(4):747–777, June 2013. URL: http://dx.doi.org/10.1007/s00454-013-9514-0, doi:10.1007/s00454-013-9514-0.

Shortest Paths, Convexity, and Treewidth in Regular Hyperbolic Tilings*

Sándor Kisfaludi-Bak[1], Tze-Yang Poon[2], and Geert van Wordragen[1]

1 **Department of Computer Science, Aalto University, Finland**
{sandor.kisfaludi-bak, geert.vanwordragen}@aalto.fi
2 **Agency for Science, Technology and Research (A*STAR), Singapore**
poontzeyang@gmail.com

Abstract

Hyperbolic tilings are natural infinite planar graphs where each vertex has degree q and each face has p edges for some $\frac{1}{p} + \frac{1}{q} < \frac{1}{2}$. We study the structure of shortest paths in such graphs. We show that given a set of n terminals, we can compute a so-called isometric closure (closely related to the geodesic convex hull) of the terminals in near-linear time, using a classic geometric convex hull algorithm as a black box. We show that the size of the convex hull is $\mathcal{O}(N)$ where N is the total length of the paths to the terminals from a fixed origin.

Furthermore, we prove that the geodesic convex hull of a set of n terminals has treewidth only $\max(12, \mathcal{O}(\log \frac{n}{p+q}))$, a bound independent of the distance of the points involved. As a consequence, we obtain algorithms for subset TSP and Steiner tree with running time $\mathcal{O}(N \log N) + \text{poly}(\frac{n}{p+q}) \cdot N$.

Related Version arXiv:2510.26110

1 Introduction

The regular tilings of the three geometries (Euclidean, elliptical/spherical, and hyperbolic) are fundamental discrete structures that have received a lot of well-deserved attention in algorithms and discrete mathematics. In the geometric algorithms literature, these objects are studied in their own right, and they are also the basis of countless algorithmic techniques: e.g., a square grid is very often the basis of geometric approximation algorithms, but the triangular and hexagonal grids are also often sought out for their properties. In the spherical setting, the tilings correspond to the platonic solids; these are finite structures that have been used and studied for millennia.

In the hyperbolic plane, unlike the other geometries, there is a much richer set of regular tilings: for any pair of integers p, q where $\frac{1}{p} + \frac{1}{q} < \frac{1}{2}$ there exists a decomposition of the hyperbolic plane (henceforth denoted by $\mathbb{H}^2$) into congruent copies of interior-disjoint regular p-gons (tiles) where each vertex is shared by q tiles. This decomposition is called the regular hyperbolic tiling of Schläfli symbol $\{p, q\}$. An emerging body of work studies regular hyperbolic tilings from the algorithmic perspective [11, 17, 16, 13], but many fundamental questions, such as computing shortest paths, remain mostly unexplored.

In Euclidean square grids it is easy to characterize shortest paths: given a pair of points, these are the set of x- and y-monotone paths connecting the pair. These shortest paths are covered by the minimum bounding box of the point pair. The problem is much less clear when one wants to find a shortest path between two given vertices of a hyperbolic tiling. Due to the fact that $\mathbb{H}^2$ is not a vector space (a typical pair of "translations" do not commute), it is significantly harder to find shortest paths in hyperbolic tilings.

* This work was supported by the Research Council of Finland, Grant 363444.

42nd European Workshop on Computational Geometry, Hagen, Germany, March 25–27, 2026.
This is an extended abstract of a presentation given at EuroCG'26. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

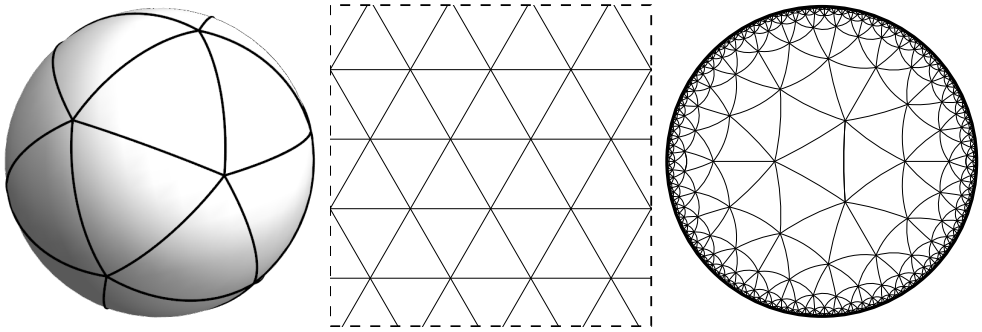


Figure 1 The regular tiling in spherical, Euclidean and hyperbolic geometry where $p = 3$ and $q = 5, 6$, and 7, respectively.

Shortest paths and intervals in graphs and tilings. In the graph setting, *geodesic convexity* has been studied intensively, see the monograph [20] for a detailed overview of the topic. In graphs, there can be several shortest paths between a pair of endpoints, so in order to generalize the notion of convexity, we need some further terminology, introduced here only for unweighted graphs. The *interval* of a vertex pair $u, v \in V(G)$ is the subgraph $I_G(u, v) \subset E(G)$ given by the union of *all* shortest paths between u and v.

An important property of shortest paths in hyperbolic tilings is that they stay together in the following sense: for any pair of points u, v in the tiling, and any pair of shortest (u, v)-paths P and P' we have that each vertex of P' is within distance $\mathcal{O}_{p,q}(1)$ from some vertex of P. This property holds much more generally, even for certain approximate shortest paths in the more general setting of Gromov-hyperbolic metric spaces. See [6] for a detailed exposition. As a result, we can already derive that an interval $I(u, v)$ in a hyperbolic tiling has constant-size vertex separators, unlike the Euclidean grid where the interval of a pair of points can be a square-shaped patch of the grid of arbitrary side length. In this article, we will get a more accurate description of intervals in hyperbolic tilings.

Convexity in graphs. In the Euclidean setting, it is natural to talk about the bounding box of a set P of grid points, and observe that all pairwise shortest paths in P are contained in the bounding box. This notion is similar to the geometric notion of *convex hulls*.

Let G be a graph. We say that a subgraph $H \subseteq E(G)$ is *convex* if for any $u, v \in V(H)$ we have $I_G(u, v) \subseteq H$. On the other hand, there is a natural weaker property: a subgraph $H \subseteq E(G)$ is *isometric* if for any $u, v \in V(H)$ we have $\text{dist}_H(u, v) = \text{dist}_G(u, v)$, where dist_X denotes the shortest-path distance in the (sub)graph X. Notice that any convex subgraph is automatically isometric, but this is not true the other way around: an isometric subgraph is guaranteed to contain *some* shortest path between any pair of vertices, but not all of them.

Using the notions of convex and isometric subgraphs, we can define corresponding closures. For a given vertex set $K \subseteq V(G)$ a subgraph H of G is a *convex hull* (respectively, a *minimal isometric closure*) of K if H is a minimal convex (resp., minimal isometric) subgraph with $K \subseteq V(H)$. Notice that the convex hull of K is in fact unique, while there can be many pairwise incomparable minimal isometric closures. See Figure 2 illustrating the difference between a minimal isometric closure and a convex hull in a Euclidean grid.

We will be interested in the following question.

▶ Question 1. Is there an efficient algorithm to compute minimal isometric closures and convex hulls in hyperbolic tilings?

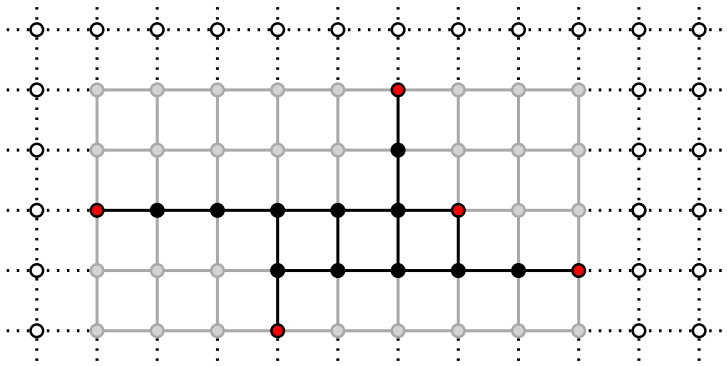


Figure 2 The convex hull (gray) and a minimal isometric closure (black) of a set of terminals (red) in the grid graph.

Optimization problems and convex hulls. Beyond the basic distance properties and convexity, the subgraphs of hyperbolic tilings hold a significant algorithmic promise as a graph class. Kisfaludi-Bak [11] observes that any n-vertex subgraph of a regular hyperbolic tiling $G_{p,q}$ has treewidth $\mathcal{O}(\log n)$ in stark contrast with the Euclidean setting where a $\sqrt{n} \times \sqrt{n}$ grid graph has treewidth $\Theta(\sqrt{n})$. This structural result combined with the literature on treewidth-based algorithms (see the book by Cygan et al. [9] for an overview) gives polynomial algorithms in this graph class for several problems that are NP-hard on planar graphs.

In the graph setting, the Subset TSP problem asks for the shortest closed walk containing a given set of terminals, and Steiner tree asks for the shortest tree containing the terminals. In our setting of an infinite host tiling, we can restrict our attention to the convex hull of the terminals. Intuitively, the number of tiling vertices inside the convex hull is similar to the geometric area of the hull; indeed, we can show that the minimal isometric closure has area that is *linear* as a function of the size of the minimum spanning tree of the terminal set, which comes from the linear isoperimetric inequality [6] that is unique to the hyperbolic setting. Using the logarithmic treewidth bound on the minimal isometric closure, one should be able to get polynomial algorithms for subset TSP and Steiner tree via a black-box usage of treewidth-based algorithms for these problems [4]. It is however unclear if it is possible to get an algorithm that is near-linear as a function of the diameter, and polynomial in the number of terminals.

▶ Question 2. Is there an algorithm to compute Subset TSP and Steiner tree in regular hyperbolic tilings that is (near)-linear in the input bit complexity and polynomial in the number of terminals?

Organization of the paper. We describe our contributions towards Question 1 in Section 2 and towards Question 2 in Section 3, together with proof sketches. Much more detailed proofs can be found in the appendix.

2 Shortest Paths and Isometric Closure Computation

We start by showing that there is a strong link between shortest paths in the tiling graph $G_{p,q}$ and lines in the hyperbolic plane; see Figure 3. While the result is somewhat expected, its proof is quite tedious, so it has been relegated to the full version.

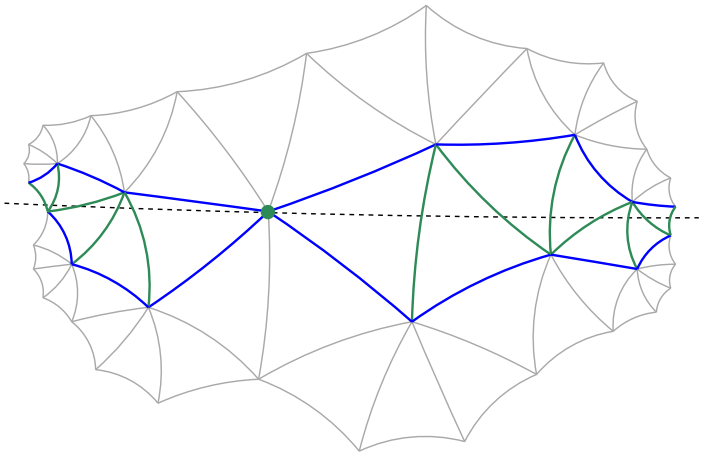


Figure 3 The subgraph (blue and green) intersected by a line ℓ (dashed) and the sequence S of vertices and edges (green) intersected by ℓ. An additional layer of nearby tiles is depicted (grey).

▶ **Lemma 2.1.** *Let S be the sequence of edges of the tiling $G_{p,q}$ intersected by some line ℓ. Then for any pair of vertices v, w that are endpoints of such edges there exists a shortest path from v to w that passes through at least one endpoint of each edge of S between the edge of v and the edge of w.*

With this lemma and some bound on the intervals $I(u, v)$ at hand, we are able to compute a shortest path from u to v (or even all shortest paths) in $\mathcal{O}(\text{dist}_{G_{p,q}}(u, v))$ time.

We can now also move on to computing isometric closures, but in order to do that, we need to fix how we receive the input points. We denote by K the set of terminals from a tiling graph $G_{p,q}$, and let $n = |K|$. We assume that each terminal is defined via a path (or walk) in $G_{p,q}$ that starts at the fixed origin and ends at the terminal (rather than with coordinates, since conversion is non-trivial). We let N be the total length of these paths.[1]

Towards Question 1, we show the following result.

▶ **Theorem 2.2.** *Given a set K of terminals in $G_{p,q}$, there is an isometric closure $\hat{G}_K$ of K that is a subgraph of the graph convex hull $\text{conv}_G(K)$. With access to real RAM, we can compute $\hat{G}_K$ in $\mathcal{O}(N \log N)$ time, where N is the total length of the paths used to define K.*

Proof sketch. The algorithm has three steps. First it computes the (geometric) convex hull of the terminals as points in the hyperbolic plane. Since in the Beltrami-Klein model hyperbolic lines appear as Euclidean lines, standard convex hull techniques such as Graham's scan [10] can be used to do this in $\mathcal{O}(n \log n)$ time. Next, it uses Lemma 2.1 to compute shortest paths between the terminals on the boundary of this convex hull, giving the boundary of $\hat{G}_K$. By the triangle inequality, the boundary of $\hat{G}_K$ has length $\mathcal{O}(N)$ and thus this can be done in $\mathcal{O}(N)$ time. Last, it fills in $\hat{G}_K$ with a depth-first search, using a balanced binary search tree to check if the vertex from the infinite graph $G_{p,q}$ with the given coordinates is already represented in $\hat{G}_K$. We can prove that $\hat{G}_K$ (and by the same arguments $\text{conv}_G(K)$) has $\mathcal{O}(N)$ vertices: a hyperbolic polygon with $\mathcal{O}(N)$ vertices has area $\mathcal{O}(N)$ and tiles of $G_{p,q}$ have area $\Omega(p)$, so $\hat{G}_K$ encloses at most $\mathcal{O}(N/p)$ tiles, each of which is incident to p vertices.

[1] For constant p, q, the number N is within a constant factor of the bit complexity if the points were given by coordinates; see the full version for a more in-depth discussion.

Thus, the depth-first search has a running time of $\mathcal{O}(N \log N)$, dominating the other running times. ◀

Note that [20] gives an algorithm that computes the graph convex hull of n terminals in any graph with m edges in $\mathcal{O}(mn)$ time, which is already optimal (under SETH) for recognizing if a given subgraph is convex [7]. In our setting we have an infinite base graph, and the straightforward restriction to a finite graph (taking all vertices within a given distance from one fixed vertex) can have $q^{\Theta(N)}$ edges.

3 Treewidth of the Convex Hull

To bound the treewidth of the convex hull $\mathrm{conv}_G(K)$, we use the following lemma.

▶ **Lemma 3.1** (Lemma 3.2 of Kisfaludi-Bak [11]). *Let $\mathcal{T}$ be a compact regular tiling of $\mathbb{H}^2$. Then the neighborhood graph of any finite tile set $S \subset \mathcal{T}$ with $|S| \geq 2$ is $(c \log |S|)$-outerplanar, where c is an absolute constant independent of our choice of $\mathcal{T}$.*

▶ **Theorem 3.2.** *For any set K of n vertices in $G_{p,q}$, the convex hull $\mathrm{conv}_G(K)$ has treewidth at most $\max\{12, \mathcal{O}(\log \frac{n}{p+q})\}$.*

Proof sketch. Since the geometric convex hull of K has area $\mathcal{O}(n)$ and faces of $G_{p,q}$ have area $\Omega(p)$, only $\mathcal{O}(n/p)$ faces can be contained inside this convex hull. Consequently, the neighborhood graph of these faces is $\mathcal{O}(\log \frac{n}{p})$-outerplanar, and instead considering the faces contained in the graph convex hull $\mathrm{conv}_G(K)$ only adds a constant to this. Because treewidth can be bounded in terms of outerplanarity [3, Theorem 83], this also gives treewidth $\mathcal{O}(\log \frac{n}{p})$ for the neighborhood graph of the faces inside $\mathrm{conv}_G(K)$. Finally, $\mathrm{conv}_G(K)$ is the dual of this and therefore still has treewidth $\mathcal{O}(\log \frac{n}{p})$ [5]. A bound of $\mathcal{O}(\log \frac{n}{q})$ follows similarly by looking at the dual tiling (which has Schläfli symbol $\{q, p\}$ instead of $\{p, q\}$) and these combine to give the theorem statement. ◀

This can be seen as a far-reaching strengthening of Lemma 3.1, which was reminiscent of bounds for other graphs that come from hyperbolic geometry [8, 1, 2]. Indeed, the treewidth depends logarithmically on the number of terminals and it is independent of the size of the convex hull, and we get stronger bounds for larger values of p and q. This is in line with the observation that hyperbolic structure becomes more tree-like and hence simpler at larger distances; see [12, 14, 2] for further examples of this phenomenon.

The above treewidth bound automatically holds also for the isometric closure $\hat{G}_K \subseteq \mathrm{conv}_G(K)$ that we compute. We prove that any minimal isometric closure of K contains a solution to Subset TSP and Steiner Tree, so we are able to compute an exact solution to these problems in $\hat{G}_K$. Using existing treewidth-based algorithms [4, 18], we finally get the following answer to Question 2.

▶ **Theorem 3.3.** *Given a set of n terminals with total description size N in a regular hyperbolic tiling graph $G_{p,q}$ with Schläfli symbol $\{p, q\}$, the Steiner tree and Subset TSP problems can be solved in $\mathcal{O}(N \log N) + \mathrm{poly}(\frac{n}{p+q}) \cdot N$ time.*

This is a significant strengthening compared to the general planar graph setting, where the problem is NP-hard and the best known algorithms are subexponential [15, 19]. Note that due to the logarithmic treewidth bound [11] one naturally expects a running time that is polynomial in the size of the underlying subgraph (here, the convex hull); the main contribution of the above theorem is that it is *near-linear* in the input size N and depends

only polynomially on the number n of terminals. Moreover, the algorithm becomes faster as p or q grows, reaching an $\mathcal{O}(N \log N)$ algorithm when $\max(p, q) = \Omega(n)$.

References

1 Thomas Bläsius, Tobias Friedrich, and Anton Krohmer. Hyperbolic random graphs: Separators and treewidth. In *24th Annual European Symposium on Algorithms, ESA 2016, August 22-24, 2016, Aarhus, Denmark*, volume 57 of *LIPIcs*, pages 15:1–15:16. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2016. doi:10.4230/LIPICS.ESA.2016.15.

2 Thomas Bläsius, Jean-Pierre von der Heydt, Sándor Kisfaludi-Bak, Marcus Wilhelm, and Geert van Wordragen. Structure and independence in hyperbolic uniform disk graphs. In *41st International Symposium on Computational Geometry, SoCG 2025*, volume 332 of *LIPIcs*, pages 21:1–21:16. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2025. doi:10.4230/LIPICS.SOCG.2025.21.

3 Hans L. Bodlaender. A partial k-arboretum of graphs with bounded treewidth. *Theor. Comput. Sci.*, 209(1-2):1–45, 1998. doi:10.1016/S0304-3975(97)00228-4.

4 Hans L. Bodlaender, Marek Cygan, Stefan Kratsch, and Jesper Nederlof. Deterministic single exponential time algorithms for connectivity problems parameterized by treewidth. *Inf. Comput.*, 243:86–111, 2015. doi:10.1016/J.IC.2014.12.008.

5 Vincent Bouchitté, Frédéric Mazoit, and Ioan Todinca. Treewidth of planar graphs: connections with duality. *Electron. Notes Discret. Math.*, 10:34–38, 2001. doi:10.1016/S1571-0653(04)00353-1.

6 Martin R. Bridson and André Haefliger. *Metric spaces of non-positive curvature*. Springer Berlin, Heidelberg, 1999. doi:10.1007/978-3-662-12494-9.

7 Sergio Cabello. Testing whether a subgraph is convex or isometric. *CoRR*, abs/2502.16193, 2025. arXiv:2502.16193.

8 Victor Chepoi, Feodor F. Dragan, Bertrand Estellon, Michel Habib, and Yann Vaxès. Diameters, centers, and approximating trees of delta-hyperbolic geodesic spaces and graphs. In *Proceedings of the 24th ACM Symposium on Computational Geometry, College Park, MD, USA, June 9-11, 2008*, pages 59–68. ACM, 2008. doi:10.1145/1377676.1377687.

9 Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 2015. doi:10.1007/978-3-319-21275-3.

10 Ronald L. Graham. An efficient algorithm for determining the convex hull of a finite planar set. *Inf. Process. Lett.*, 1(4):132–133, 1972. doi:10.1016/0020-0190(72)90045-2.

11 Sándor Kisfaludi-Bak. Hyperbolic intersection graphs and (quasi)-polynomial time. In *Proceedings of the 2020 ACM-SIAM Symposium on Discrete Algorithms, SODA 2020*, pages 1621–1638. SIAM, 2020. doi:10.1137/1.9781611975994.100.

12 Sándor Kisfaludi-Bak. A quasi-polynomial algorithm for well-spaced hyperbolic TSP. In Sergio Cabello and Danny Z. Chen, editors, *36th International Symposium on Computational Geometry, SoCG 2020*, volume 164 of *LIPIcs*, pages 55:1–55:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPICS.SOCG.2020.55.

13 Sándor Kisfaludi-Bak, Jana Masaríková, Erik Jan van Leeuwen, Bartosz Walczak, and Karol Wegrzycki. Separator theorem and algorithms for planar hyperbolic graphs. In *40th International Symposium on Computational Geometry, SoCG 2024*, volume 293 of *LIPIcs*, pages 67:1–67:17. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2024. doi:10.4230/LIPICS.SOCG.2024.67.

14 Sándor Kisfaludi-Bak and Geert van Wordragen. A quadtree, a Steiner spanner, and approximate nearest neighbours in hyperbolic space. In Wolfgang Mulzer and Jeff M. Phillips, editors, *40th International Symposium on Computational Geometry, SoCG 2024*,

volume 293 of *LIPIcs*, pages 68:1–68:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2024. doi:10.4230/LIPICS.SOCG.2024.68.

15 Philip N. Klein and Dániel Marx. A subexponential parameterized algorithm for subset TSP on planar graphs. In Chandra Chekuri, editor, *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2014*, pages 1812–1830. SIAM, 2014. doi:10.1137/1.9781611973402.131.

16 Eryk Kopczynski. Hyperbolic minesweeper is in P. In *10th International Conference on Fun with Algorithms, FUN 2021*, volume 157 of *LIPIcs*, pages 18:1–18:7. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPICS.FUN.2021.18.

17 Eryk Kopczyński and Dorota Celińska-Kopczyńska. Dynamic distances in hyperbolic graphs, 2021. arXiv:2111.01019.

18 Hung Le. A PTAS for subset TSP in minor-free graphs. *CoRR*, abs/1804.01588, 2018. arXiv:1804.01588.

19 Dániel Marx, Marcin Pilipczuk, and Michal Pilipczuk. On subexponential parameterized algorithms for Steiner tree and directed subset TSP on planar graphs. In Mikkel Thorup, editor, *59th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2018*, pages 474–484. IEEE Computer Society, 2018. doi:10.1109/FOCS.2018.00052.

20 Ignacio M. Pelayo. *Geodesic Convexity in Graphs*. SpringerBriefs in Mathematics. Springer New York, NY, 1 edition, 2013. doi:10.1007/978-1-4614-8699-2.

Host

FernUniversität in Hagen

Sponsors

Fakultät für
Mathematik und Informatik

Funded by
DFG Deutsche Forschungsgemeinschaft
German Research Foundation

GESELLSCHAFT DER
FREUNDE DER
FERNUNIVERSITÄT e.V.

TABLESCOPE

EuroCG 2026

eurocg26@fernuni-hagen.de