39th European Workshop on Computational Geometry

March 29-31, 2023, Barcelona, Spain
https://dccg.upc.edu/eurocg23/

EuroCG2023

Book of Abstracts

Preface

The 39th European Workshop on Computational Geometry (EuroCG 2023) was held on March 29-31 in Barcelona, Spain. EuroCG is an annual workshop that combines a strong scientific tradition with a friendly and informal atmosphere. The workshop is a forum where established researchers and students can meet, discuss their work, present their results, and establish scientific collaborations in order to promote research in the field of Computational Geometry.

This year we had 116 registered participants. Concerning the scientific program, we received 69 submissions, which underwent a limited refereeing process by the program committee in order to ensure some minimal standards and to check for plausibility. We selected 63 submissions for presentation at the workshop.

EuroCG does not have formally published proceedings; therefore, we expect most of the results outlined here to be also submitted to peer-reviewed conferences and/or journals. This book of abstracts, available through the EuroCG 2023 web site, should be regarded as a collection of preprints. In addition to the 63 contributed talks, this book contains abstracts of the three invited lectures.

The invited speakers were Sergio Cabello, Evanthia Papadopoulou and Alberto Márquez.

Many thanks to all authors, session chairs, speakers, and invited speakers for the participation, and to the members of the program committee and all external reviewers for their insightful comments. We also thank the organizing committee members. Finally, we are very grateful for the generous support of our sponsors: Gold sponsor Omron, Contributors: Universitat Politècnica de Catalunya, BarcelonaTech (UPC), Departament de Matemàtiques (UPC), and Societat Catalana de Matemàtiques.

EuroCG 2023 has had a Best Student Presentation Award. The prize was voted by the workshop attendees to recognize the effort of young researchers to present their work clearly and elegantly. The winner was Max van Mulken, for the presentation of paper "Density Approximation for Kinetic Groups". During the business meeting Michael Bekos presented the 40th edition of EuroCG, which will take place in Ioannina, Greece, March 13-15, 2024. A single bid was presented for 2025 by Pavel Valtr and, as a consequence, EuroCG 2025 will take place in Prague. Looking forward to seeing you all next year in Ioannina!

March 2023,

Clemens Huemer and Carlos Seara

External Reviewers

Henk Alkema
Patrizio Angelini
Helena Bergold
Sujoy Bhore
Piotr Borowiecki
Anna Brötzner
Sergio Cabello
Jean Cardinal
Anna De Mier
Bianca Dornelas
Adrian Dumitrescu
Kunal Dutta
Nicolas El Maalouly
David Eppstein
Ruy Fabila-Monroy
Stefan Felsner
Oksana Firman
Henry Förster
Jarosław Grytczuk
Rimma Hamalainen
Thekla Hamm
Tim Hegemann
Rahul Jain
Andrei Jalba
Julia Katheder
Philipp Kindermann
Sándor Kisfaludi-Bak
Linda Kleist
Felix Klesen
Fabian Klute
Kolja Knauer
Irina Kostitsyna
Miroslaw Kowaluk
Robert Lauff
Christos Levcopoulos
Ioannis Mantas
Bodo Manthey
Lucas Meijer
Till Miltzow
Giacomo Ortali
Pavel Paták
Julian Pfeifle
Vincent Pilaud
Tommaso Piselli
Chrysanthi Raftopoulou
Meghana M Reddy
Marco Ricci
Christian Rieck
Sandro Roch
Jonathan Rollin
Arpan Sadhukhan
Felix Schröder
Marie Diana Sieper
Jan Soukup
Milos Stojakovic
Sabine Storandt
Ichiro Suzuki
Martin Tancer
Alessandra Tappini
Leonidas Theocharous
Hans Raj Tiwary
Thomas C. van Dijk
Simon Weber
Alexander Wolff
Ji Zeng
Johannes Zink

Table of Contents

Three Open Problems

Sergio Cabello

University of Ljubljana & Institute of Mathematics, Physics and Mechanics, Ljubljana

Abstract

I would like to share with you three open problems in Computational Geometry:

- Expected volume of the stochastic bounding box;
- Maximum matching in unit disk graphs or unit square graphs;
- Barrier resilience.

For each of these problems, there are interesting results and I will discuss some of the ideas that have been used, but the main question remains open.

Abstract Voronoi-like Graphs and Applications

Evanthia Papadopoulou

Università della Svizzera italiana

Abstract

Differences between classical Voronoi diagrams of points, versus Voronoi diagrams of segments, disks, polygons, or clusters of points in the plane, tend sometimes to be overlooked. As a result some basic questions concerning the latter diagrams may remain open or non-optimally solved. In this talk, I will discuss Voronoi-like graphs as a tool towards resolving such differences.

A Voronoi-like graph is a relaxed Voronoi structure, defined as a graph on the arrangement of a given bisector system. In a Voronoi-like graph, a vertex v and its incident edges, within a small neighborhood around v, must appear in a Voronoi diagram of three sites. For points in the Euclidean plane, where the bisector system forms a line arrangement, a Voronoi-like graph always coincides with the Voronoi diagram of the involved sites. How about bisector systems, which are not lines (nor pseudolines), such as those related to line-segments, more generally, to abstract Voronoi diagrams? I will answer this question in this talk and give examples of simple expected linear-time algorithms to compute Voronoi-like trees (or forests). Examples include updating an abstract Voronoi diagram after deletion of one site, updating a constraint Delaunay triangulation after inserting a new segment constraint, and others.

Some parts of this talk are joint work with Kolja Junginger.

Graph theory meets geometry

Alberto Márquez

Universidad de Sevilla

Abstract

Geometric graph theory involves considering the realization of a graph in Euclidean space and studying its metric properties. This area has been extensively researched, with a vast literature available. However, there are certain functional differences between geometric graph theory and traditional graph theory, and studying these functional differences has been a focus of recent research. By considering the ways in which geometric graph theory differs from traditional graph theory, we can better understand how to modify geometric graphs in order to improve those functionals.

Joint work with several people, mainly D. Garijo and R. Silveira.

Two Equivalent Representations of Bicolored Order Types

Oswin Aichholzer[1] **and Anna Brötzner**[2]

1 **Graz University of Technology, Graz, Austria**
`oaich@ist.tugraz.at`

2 **Malmö University, Malmö, Sweden**
`anna.brotzner@mau.se`

Abstract

In their seminal work on Multidimensional Sorting, Goodman and Pollack introduced the so-called order type, which for each ordered triple of a point set in the plane gives its orientation, clockwise or counterclockwise. This information is sufficient to solve many problems from discrete geometry where properties of point sets do not depend on the exact coordinates of the points but only on their relative positions. Goodman and Pollack showed that an efficient way to store an order type in a matrix λ of quadratic size (w.r.t. the number of points) is to count for every oriented line spanned by two points of the set how many of the remaining points lie to the left of this line.

We generalize the concept of order types to bicolored point sets (every point has one of two colors). The bicolored order type contains the orientation of each bicolored triple of points, while no information is stored for monochromatic triples. Similar to the uncolored case, we store the number of blue points that are to the left of an oriented line spanned by two red points or by one red and one blue point in λ_B. Analogously the number of red points is stored in λ_R.

We show that the equivalence of the information contained in the orientation of all bicolored point triples and the two matrices λ_B and λ_R also holds in the colored case. This is remarkable, as in general the bicolored order type does not even contain sufficient information to determine all extreme points (points on the boundary of the convex hull of the point set).

1 Introduction

Many problems from discrete geometry are based on properties of point sets in the plane that do not depend on the exact coordinates of the points but only on their relative positions. In their seminal work on Multidimensional Sorting [1], Goodman and Pollack introduced the so-called order type, which for each ordered triple of a point set in the plane gives its orientation, clockwise, counterclockwise, or collinear. For a point set S the order type can be stored in a matrix Λ which is of cubic size with respect to the number of points of S. For every triple $p, q, s \in S$ we encode its orientation by $\Lambda(p, q, s) \in \{-1, 0, 1\}$ where "-1" means clockwise, "0" means collinear, and "$+1$" means counterclockwise. [1] An alternative way to code the order type is to use a matrix λ of quadratic size. For two points $p, q \in S$ the entry $\lambda(p, q)$ counts the number of points from $S \setminus \{p, q\}$ to the left of the oriented line through p and q. Goodman and Pollack [1] showed that the information contained in λ is the same as in Λ, and that the two matrices can be converted into each other in polynomial time.

Colored point sets have a long history in discrete and computational geometry – see the recent survey of Kano and Urrutia [2] for a nice collection of problems in this area.

[1] Note that Goodman and Pollack[1] define $\Lambda(p, q)$ as the set of points on the left of ℓ_{pq}. However, both definitions contain the equivalent information. For convenience, we directly use the triple orientation $\Lambda(p, q, s)$ of the point triple (p, q, s).

39th European Workshop on Computational Geometry, Barcelona, Spain, March 29–31, 2023.

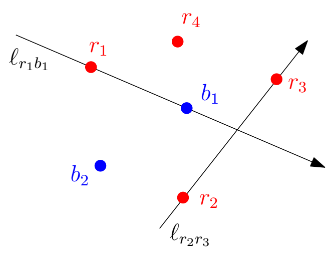


Figure 1 Counting points to the left of a line induced by the red point r_1 and the blue point b_1: $\lambda_B(r_1, b_1) = 0$ and $\lambda_R(r_1, b_1) = 2$. For the two red points r_2 and r_3, $\lambda_B(r_2, r_3) = 2$, while $\lambda_R(r_2, r_3)$ is not defined.

Consequently, we extend the concept of order types to bicolored order types. Let P be a set of at least 3 points in the plane in general position, that is, no three points lie on a common line. Using the symbol $\dot\cup$ for the disjoint union let $P = B \,\dot\cup\, R$ be partitioned into two disjoint sets B and R, $|B| = m$ and $|R| = n$, where the points $b_1, \ldots, b_m \in B$ are colored blue, and the points $r_1, \ldots, r_n \in R$ are colored red. An oriented line through two points $p, q \in P$, directed from p to q, will be denoted by ℓ_{pq}. For three points $p, q, s \in P$ the triple orientation $\Lambda(p, q, s)$ (clockwise or counterclockwise) is determined by considering the oriented line ℓ_{pq} and checking in which of the open half-planes defined by ℓ_{pq} (right or left) the point s lies. We denote the collection of the orientations of all bicolored point triples of P (triples that contain at least one blue and one red point) by $\Lambda(P)$ or simply by Λ if it is clear from the context which point set is considered. Point sets where all triple orientations are the same belong to the same equivalence class called the *bicolored order type* of P.

▶ **Definition 1.1.** Let $\mathcal{P}$ be the set of all bicolored point sets in general position in the plane consisting of m blue and n red points. The *bicolored order type* of size (m, n) is the equivalence class on $\mathcal{P}$ where two sets of $\mathcal{P}$ are equivalent if all bicolored triple orientations are the same.

Note that the orientation of monochromatic point triples is not encoded in Λ. Similar to λ for uncolored order types we define λ_B, λ_R to count the number of blue, respectively red, points to the left of an oriented line spanned by two points from P. More precisely, for any pair of red points $r_i, r_j \in R$ we count the number of blue points to the left of the oriented line $\ell_{r_ir_j}$ in $\lambda_B(r_i, r_j)$, and for any bicolored pair of points $b_i \in B$, $r_j \in R$ we count the number of blue points to the left of the oriented line $\ell_{b_ir_j}$ in $\lambda_B(b_i, r_j)$ and the number of blue points to the left of the oriented line $\ell_{r_jb_i}$ in $\lambda_B(r_j, b_i)$. In an analogous way we count the number of red points in λ_R. See Figure 1 for an example.

In Section 2 we show our central result that the information contained in λ_B and λ_R is equivalent to the information given by all bicolored triple orientations Λ. Moreover, given one of the two representations of a bicolored order type, the other can be derived in polynomial time. This result has to be seen in contrast to the fact that in general we cannot use the bicolored order type to determine all extreme points of a bicolored point set as the orientation of monochromatic point triples is not given; see Figure 2 for an example.

2 Equivalence of (λ_B,λ_R) and Λ

Given all point triples Λ, in the uncolored case it is straightforward to compute the λ-matrix. Goodman and Pollack [1] showed that the opposite direction also holds, namely that the λ-matrix uniquely determines Λ. Given λ, for an extremal point p it holds that $\lambda(p, q) = 0$ for

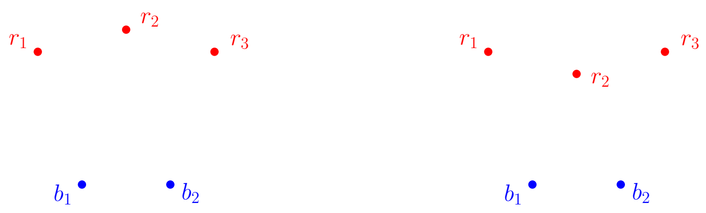


Figure 2 Two point sets with the same bicolored order type. We cannot determine whether r_2 lies on the boundary of the convex hull as the orientation of the red point triple (r_1, r_2, r_3) is unknown.

some point $q \in P \setminus \{p\}$. To compute the triple orientations from the λ-matrix, we iteratively choose an extremal point p and remove it from the set after computing all triple orientations involving p. This is essentially done by sorting all remaining points radially around p, using the information of λ to compare the order of two elements.

We show that a similar equivalence holds for bicolored order types. Again, given all bicolored triple orientations Λ, computing λ_B and λ_R is straightforward. So in the following, we will argue the inverse direction, which requires some more involved steps.

For bicolored point sets, a bicolored edge $\{b, r\}$ lies on the boundary of the convex hull if and only if either (1) $\lambda_B(b, r) = \lambda_R(b, r) = 0$ or (2) $\lambda_B(b, r) = m - 1$ and $\lambda_R(b, r) = n - 1$ holds. Then b and r are extreme points and can be found by testing all mn bicolored edges.

For monochromatic edges we can in general not determine whether they lie on the boundary of the convex hull, see Figure 2 for an example. The reason is that in λ_B and λ_R it is not encoded if points of the same color as the monochromatic edge lie on both sides of it. This also implies that we cannot always determine all extreme points.

However, if no bicolored edge on the boundary of the convex hull exists, we know that the boundary of the convex hull consists solely of points of one color. We thus can find a point of this color which is extremal to the set of the other color, that is, which lies outside of the convex hull of the points of the other color, by inspecting all $\mathcal{O}(m^2 + n^2)$ monochromatic edges. For such an edge we check whether all points of the other color lie on one side of it. As obviously an extreme point (for example from a bicolored edge on the boundary of the convex hull) is also extremal to the set of the other color, we obtain the following observation.

▶ **Observation 2.1.** *For a given bicolored point set $P = B \dot\cup R$ we can determine all points which are extreme w.r.t. the other color by using λ_B and λ_R in $\mathcal{O}(m^2 + n^2)$ steps.*

As we have seen, using just λ_B and λ_R it is not always possible to determine whether a point is an extreme point. We therefore extend the concept and make use of points that are not "dominated" by other points. We call such points *undominated*.

▶ **Definition 2.2.** A red point $r \in R$ is *undominated* if it (1) lies outside of the convex hull of B, and (2) the wedge formed by the tangents of r to the convex hull of B which is opposite of B is empty of points of R. The symmetric definition holds for a blue point $b \in B$.

▶ **Lemma 2.3.** *Given the matrices λ_B and λ_R and an undominated point $p \in B \dot\cup R$, all bicolored triple orientations involving p can be determined in constant time per triple.*

Proof. Without loss of generality let $r \in R$ be an undominated red point. By definition, r lies outside the convex hull of B and the wedge between the two tangents of r to the convex hull of B that lies opposite of B is empty. We first compute the triple orientations of r and any two blue points; see Figure 3 for an illustration of the proof. Recall that for every $b \in B$

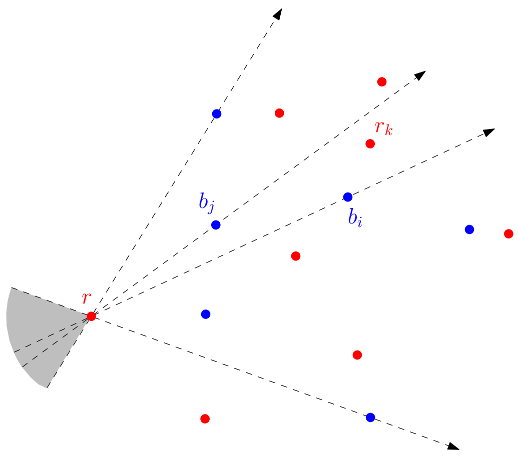


Figure 3 For an undominated point $r \in R$ the wedge marked in gray is empty of other points. All bicolored triple orientations involving r can be determined by using λ_B.

the number of blue points to the left of ℓ_{rb} is given by $\lambda_B(r,b)$. Thus, the rotational order of blue points around r can be read from λ_B, which yields the desired triple orientations.

$$\Lambda(r,b_i,b_j) = \Lambda(b_i,b_j,r) = \Lambda(b_j,r,b_i) = \begin{cases} +1, & \text{if } \lambda_B(r,b_i) > \lambda_B(r,b_j) \\ -1, & \text{otherwise.} \end{cases} \tag{1}$$

This implies the inverse triple orientations $\Lambda(b_j,b_i,r) = \Lambda(b_i,r,b_j) = \Lambda(r,b_j,b_i) = -\Lambda(r,b_i,b_j)$.

Next we consider the triple orientations involving r, another red point and one blue point. For every red point r_k, the number of blue points lying to the left of ℓ_{rr_k} is given by $\lambda_B(r,r_k)$. If a blue point b lies on the right side of ℓ_{rr_k}, then all blue points that lie to the left of ℓ_{rr_k} also lie to the left of ℓ_{rb}, as r is undominated. This is the case if and only if $\lambda_B(r,r_k) \leq \lambda_B(r,b)$. Thus we get

$$\Lambda(r,b,r_k) = \Lambda(b,r_k,r) = \Lambda(r_k,r,b) = \begin{cases} +1, & \text{if } \lambda_B(r,r_k) \leq \lambda_B(r,b) \\ -1, & \text{otherwise.} \end{cases} \tag{2}$$

Similar as before we have $\Lambda(r_k,b,r) = \Lambda(b,r,r_k) = \Lambda(r,r_k,b) = -\Lambda(r,b,r_k)$.

Observe that for every triple we only query two entries in λ_B. As claimed we can thus compute any triple orientation that involves r in constant time per triple. ◀

Next, we show that for a bicolored point set with a monochromatic, say red, convex hull we can always find an undominated red point by using only the information given in λ_B and λ_R. A red point r which is extremal with respect to B can easily be found via searching for another red point r_2 such that $\lambda_B(r,r_2) = 0$. Note that in that case both r and r_2 are extremal w.r.t. B, and that r_2 serves as a witness for r. The proof of the following Lemma describes how to additionally test whether r is undominated.

▶ **Lemma 2.4.** *Let $P = B \dot\cup R$ be a bicolored point set with a monochromatic convex hull and let p be a point which is extreme with respect to the other color class. Then, in $\mathcal{O}(m)$ steps for $p \in R$ (respectively $\mathcal{O}(n)$ steps for $p \in B$) we can determine whether p is undominated.*

Proof. Assume without loss of generality that the convex hull is red and let $r \in R$ be extremal with respect to B. From λ_B we can easily find the blue points b^1 $(\lambda_B(r,b^1) = 0)$

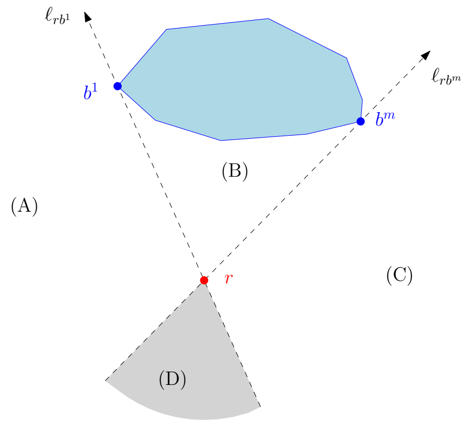


Figure 4 The tangents ℓ_{rb^1}, ℓ_{rb^m} of r define the four wedges (A), (B), (C), and (D).

and b^m ($\lambda_B(r, b^m) = m-1$) of B which define the left and the right tangent of r to the convex hull of B. If the wedge between these tangents through r that lies opposite of B – wedge (D) in Figure 4 – is empty, then r is undominated. To see this, we first compute the number of red points in wedge (A); see again Figure 4. For every red point r_j we have $\lambda_B(r, r_j) = 0$ if and only if r_j lies in wedge (A). Counting the corresponding entries in λ_B yields the number of red points in (A). Finally, $\lambda_R(r, b^1)$ gives the total number of red points in (A) and (D) together. Subtracting the number of red points in (A) from $\lambda_R(r, b^1)$ yields the number of red points in the wedge (D). If (D) is empty, then r is undominated by definition.

To find the two tangents through r and for counting the number of red points in (A) we have to inspect m entries of λ_B, all other steps need only constantly many values of λ_B and λ_R. Thus $\mathcal{O}(m)$ steps are sufficient to determine whether r is undominated. ◀

▶ **Theorem 2.5.** *For a point set $P = B \mathbin{\dot\cup} R$ the information contained in λ_B and λ_R is equivalent to the information given by all bicolored triple orientations Λ. Given one of the two representations of a bicolored order type, the other can be computed in polynomial time.*

Proof. Computing λ_B and λ_R from Λ can trivially be done in $\mathcal{O}(mn(m+n))$ steps. So assume we are given λ_B and λ_R and want to compute Λ. For any point set there always exist undominated points, as every extreme point is obviously an undominated point. Thus, by combining Observation 2.1 and Lemma 2.4 we can find an undominated point, say p, in time $\mathcal{O}(m^3 + n^3)$. The proof of Lemma 2.3 tells us how to compute all bicolored triple orientations involving p in time $\mathcal{O}(m^2 + n^2)$. After this, we can remove p from the set and update the λ-matrices as follows. If w.l.o.g. we remove a red point $p = r_k$ then, for every triple (r_k, b_i, b_j), $i < j$, with $\Lambda(r_k, b_i, b_j) = 1$, r_k lies to the left of $\ell_{b_i b_j}$, so $\lambda_R(b_i, b_j)$ has to be decremented by 1 for each such triple. Similarly, if $\Lambda(r_k, b_i, b_j) = -1$, r_k lies to the right of $\ell_{b_i b_j}$ and thus $\lambda_R(b_j, b_i)$ is decremented by 1. For each triple consisting of r_k, a blue point b and another red point r_l, $r_l \neq r_k$, we also need to update λ_R. If $\Lambda(r_k, b, r_l) = 1$, then the value of $\lambda_R(b, r_l)$ is decremented by 1. Vice versa, in case $\Lambda(r_k, b, r_l) = -1$, $\lambda_R(r_l, b)$ gets decremented by 1. Moreover, the row and the column at index k are removed from both λ_B and λ_R as r_k is removed from the set. These updates can be done in total time $\mathcal{O}(m^2 + n^2)$. We are left with λ-matrices of size $(m+n-1) \times (m+n-1)$. Doing this repeatedly $\mathcal{O}(m+n)$ times shows that all bicolored triple orientations Λ can be computed from λ_B and λ_R. The total number of steps needed is $\mathcal{O}(m^4 + n^4)$. ◀

Note that the algorithms and thus the time bounds given in the previous proof are not optimized, but just serve as witnesses to show that the transformation between the two representations of a bicolored order type can be done in polynomial time.

3 Outlook

In a forthcoming paper, we use the equivalence between Λ and λ_B, λ_R to show that several tasks on bicolored point sets can be solved in polynomial time by taking only the information contained in its bicolored order type. We show how to sort one color class around a point of the other color class, and how to determine whether the two color classes can be linearly separated. Furthermore, we elaborate on how to determine crossings between bicolored edges. We use this to find bicolored plane perfect matchings for a given bicolored order type. For given λ_B and λ_R we show how to compute the number of crossings of the complete bipartite graph drawn on the represented bicolored point set in quadratic time. Finally, we generate all realizable bicolored order types of small cardinality and give their numbers up to $m+n \leq 10$ and compare them to the number of uncolored order types.

References

1 Jacob E. Goodman and Richard Pollack. Multidimensional Sorting. *SIAM Journal on Computing*, 12(3):484–507, 1983. `doi:10.1137/0212032`.

2 Mikio Kano and Jorge Urrutia. Discrete geometry on colored point sets in the plane — a survey. *Graphs and Combinatorics*, 37(1):1–53, Jan 2021. `doi:10.1007/s00373-020-02210-8`.

Using SAT to study plane Hamiltonian substructures in simple drawings*†

Helena Bergold[1], **Stefan Felsner**[2], **Meghana M. Reddy**[3], and **Manfred Scheucher**[2]

1 **Department of Computer Science, Freie Universität Berlin, Germany,**
`firstname.lastname@fu-berlin.de`

2 **Institut für Mathematik, Technische Universität Berlin, Germany,**
`lastname@math.tu-berlin.de`

3 **Department of Computer Science, ETH Zürich, Switzerland,**
`meghana.mreddy@inf.ethz.ch`

Abstract

In a *simple drawing* of a complete graph, all edges are drawn as simple curves and every pair of edges intersects in at most one point which is either a common vertex or a proper crossing. Simple drawings generalize straight-line drawings in a similar vein as abstract order types generalize point sets. They have been studied for many years and have become a source for many open problems and conjectures.

We provide a SAT framework which allows enumerating simple drawings with specified properties or to decide that no such drawing exists. Using this framework we look at open problems on classes of simple drawings. Based on the data, we provide new strengthenings and modifications of existing conjectures. Some of these problems concern non-crossing substructures in simple drawings. The most prominent example may be the conjecture by Rafla (1988), which asserts that every simple drawing of the complete graph K_n contains a plane Hamiltonian cycle. Today, however, only the existence of plane paths of logarithmic size and plane matchings of size $\Omega(\sqrt{n})$ is known (Suk and Zeng 2022; Aichholzer et al. 2022). Here we present a proof for the existence of plane Hamiltonian subgraphs with $2n-3$ edges in *convex drawings* which are a rich subclass of simple drawings. Our proof comes with a polynomial time algorithm.

1 Introduction

In a *simple drawing* of a graph in the plane (resp. on the sphere), the vertices are mapped to distinct points, and edges are drawn as simple curves that connect the corresponding endpoints but do not contain other vertices. Moreover, every pair of edges intersects in at most one point, which is either a common vertex or a proper crossing (no touching), and no three edges cross at a common point. Figure 1 shows the obstructions to simple drawings. Throughout this article, we will only consider simple drawings of the complete graph K_n.

Problems related to simple drawings have attracted a lot of attention. One of the reasons is that simple drawings are closely related to interesting classes of drawings such as crossing-minimal drawings or straight-line drawings, a.k.a. *geometric drawings*. The focus of this article will be on the class of convex drawings, which was recently introduced by Arroyo

* Supplemental source code and data is available at [1]

† We thank Felix Schröder for fruitful discussions. H. Bergold was supported by the DFG-Research Training Group 'Facets of Complexity' (DFG-GRK 2434). S. Felsner was supported by the DFG Grant FE 340/13-1. M. M. Reddy was supported by the Swiss National Science Foundation within the collaborative DACH project *Arrangements and Drawings* as SNSF Project 200021E-171681. M. Scheucher was supported by the DFG Grant SCHE 2214/1-1.

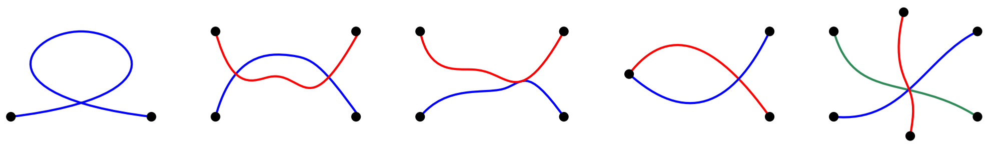


Figure 1 The five obstructions to simple drawings.

et al. [5]. Convex drawings are nested between simple drawings and geometric drawings, and a large subclass of convex drawings is in correspondence with acyclic oriented matroids of rank 3. For further aspects of the convexity hierarchy we refer the interested reader to [5, 4, 6].

The notion of convexity is based on the *triangles* of a drawing, i.e., the subdrawings induced by three vertices. Since the edges of a triangle do not cross in simple drawings, a triangle partitions the plane (resp. sphere) into exactly two connected components. The closures of these components are the two *sides* of the triangle. A side S of a triangle is *convex* if every edge that has its two vertices in S is completely drawn inside S. A simple drawing of the K_n is *convex* if every triangle has a convex side.

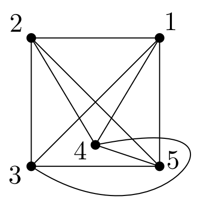
To study forced and forbidden substructures in general simple drawings and subclasses such as convex drawings, we develop a Python program which generates a Boolean formula in conjunctive normal form (CNF) that is satisfiable if and only if there exists a simple drawing of K_n (for a specified value of n) with prescribed properties. Moreover, the solutions of these instances are in one-to-one correspondence with non-isomorphic simple drawings with the prescribed properties. We then use the state of the art SAT solvers PicoSAT [7] and CaDiCaL [8] to decide whether a solution exists and to enumerate the solutions. Unsatisfiability results can be verified using the independent proof checking tool DRAT-trim [21].

In order to encode a simple drawing of the complete graph in terms of a CNF, it is important to note that combinatorial properties such as pairs of crossings are fully determined by the rotation system. For a given simple drawing D and a vertex v of D, the cyclic order π_v of incident edges in counterclockwise order around v is called the *rotation of v* in D. The collection of rotations of all vertices is called the *rotation system* of D. More specifically, the rotation system captures the combinatorial properties of a simple drawing on the sphere – the choice of the outer cell when stereographically projecting the drawing onto a plane has no effect on the rotation system.

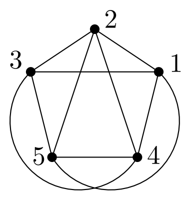
Π°_4 :	$\Pi^{\circ}_{5,1}$:	$\Pi^{\circ}_{5,2}$:
π_1 : 2 3 4	π_1 : 2 3 4 5	π_1 : 2 3 4 5
π_2 : 1 3 4	π_2 : 1 3 4 5	π_2 : 1 3 5 4
π_3 : 1 2 4	π_3 : 1 4 2 5	π_3 : 1 4 2 5
π_4 : 1 3 2	π_4 : 1 5 3 2	π_4 : 1 5 3 2
	π_5 : 1 4 2 3	π_5 : 1 2 4 3

Figure 2 The three obstructions Π°_4, $\Pi^{\circ}_{5,1}$, and $\Pi^{\circ}_{5,2}$ for rotation systems.

The SAT encoding uses Boolean variables and clauses to encode the rotations of vertices. To assert that the prescribed permutations for the vertices (which we refer to as *pre-rotation system*) can be realized by a simple drawing, we use Boolean formulas to forbid obstructions. A computational result of Ábrego et al. [2] together with a result of Kynčl [14, Theorem 1.1] characterizes drawability in terms of induced 4- and 5-tuples: A pre-rotation system on n elements is drawable if and only if it does not contain Π°_4, $\Pi^{\circ}_{5,1}$ and $\Pi^{\circ}_{5,2}$ (Figure 2) as a subconfiguration. To encode convex drawings, we use of result by Arroyo et al. [5] that characterizes convex drawings in terms of 5-tuples: A simple drawing is *convex* if and only if it does not contain $\Pi^{\text{oc}}_{5,1}$ or $\Pi^{\text{oc}}_{5,2}$ (depicted in Figure 3) as a subconfiguration.



$\Pi^{\text{oc}}_{5,1}$:
π_1 : 2 3 4 5
π_2 : 1 3 4 5
π_3 : 1 2 4 5
π_4 : 1 2 5 3
π_5 : 1 2 4 3



$\Pi^{\text{oc}}_{5,2}$:
π_1 : 2 3 4 5
π_2 : 1 3 5 4
π_3 : 1 2 4 5
π_4 : 1 2 5 3
π_5 : 1 4 2 3

Figure 3 The two obstructions $\Pi^{\text{oc}}_{5,1}$ and $\Pi^{\text{oc}}_{5,2}$ for convex drawings.

We discuss questions regarding plane substructures in drawings of the complete graph. In particular we focus on variants of Rafla's conjecture [17], which asserts that every simple drawing of the complete graph contains a plane Hamiltonian cycle.

▶ **Conjecture 1.1** ([17]). *Every simple drawing of K_n contains a plane Hamiltonian cycle.*

Rafla verified the conjecture for $n \leq 7$. Later Ábrego et al. [2] did a complete enumeration of all rotation systems for $n \leq 9$ and verified the conjecture for $n \leq 9$. Furthermore the conjecture was proven for some particular subclasses such as geometric, monotone and cylindrical drawings. With the SAT framework we are able to verify Conjecture 1.1 for all $n \leq 10$.

Recently, Suk and Zeng [20] and Aichholzer et al. [3] independently showed that simple drawings contain a plane path of length $(\log n)^{1-o(1)}$. Suk and Zeng also showed that every simple drawing of K_n contains a plane copy of every tree on $(\log n)^{1/4-o(1)}$ vertices. Aichholzer et al. moreover showed the existence of a plane matching of size $\Omega(n^{1/2})$, which improves older bounds from a series of papers [15, 16, 9, 19, 11, 10, 18].

Fulek and Ruiz–Vargas [11, Lemma 2.1] showed that every simple drawing of K_n contains a plane subdrawing with $2n-3$ edges. Note that this bound is best-possible because every plane subgraph of the straight-line drawing of K_n on a point set in convex position (see Figure 4(left)) is outerplanar and thus has at most $2n-3$ edges. In general, it is NP-complete to determine the size of the largest plane subdrawing [12]. Based on the data for small n, we conjecture that indeed every simple drawing contains a plane Hamiltonian subgraph on $2n-3$ edges. We verified this conjecture for $n \leq 8$.

▶ **Conjecture 1.2.** *Every simple drawing of K_n with $n \geq 3$ contains a plane Hamiltonian subdrawing on $2n-3$ edges.*

For the class of convex drawings, we succeeded in proving a strengthened version of the conjecture where the plane Hamiltonian subgraph contains a spanning star.

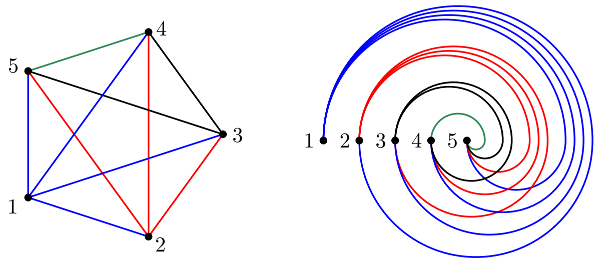


Figure 4 (left) the perfect convex C_5 and (right) the perfect twisted drawing T_5.

▶ **Theorem 1.3.** *Let D be a convex drawing of K_n with $n \geq 3$ and let $v_\star$ be a vertex of D. Then D contains a plane Hamiltonian cycle C which does not cross any edge incident to $v_\star$. This Hamiltonian cycle can be computed in $O(n^2)$ time.*

Proof sketch. We show that convex drawings have a layering structure and reduce the problem of finding a Hamiltonian cycle to finding a Hamiltonian path for each layer independently. To simplify the proof and to reduce the number of cases that have to be considered, we make use of the SAT framework.

For a convex drawing D of the complete graph K_n and a fixed vertex $v_\star$, we give an algorithm that computes a plane Hamiltonian cycle which does not cross edges incident to $v_\star$. We assume that $v_\star = n$ and that the other vertices are labeled from 1 to $n-1$ in cyclic order around $v_\star$. Moreover, by applying suitable stereographic projections, we can assume that $v_\star$ belongs to the outer face. We denote vertex $v_\star = n$ as the *star vertex* and edges incident to $v_\star$ as *star edges*. We call an edge *star-crossing* if it crosses a star edge. For the Hamiltonian cycle we only use edges which are not star-crossing. We particularly focus on the edges $e = \{v, v+1\}$ between consecutive neighbors in the cyclic order around $v_\star$, that is, $1 \leq v < n$ and $v+1$ is considered modulo $n-1$. An edge $e = \{v, v+1\}$ is called *good* if it is not star-crossing. Otherwise, if the edge $b = \{v, v+1\}$ crosses a star edge $\{w, v_\star\}$, then we say that b is a *bad edge* and w is a *witness* for b.

If there is at most one bad edge $\{v, v+1\}$, then the $n-2$ good edges together with the two star edges $\{v, v_\star\}$ and $\{v+1, v_\star\}$ form a Hamiltonian cycle. Hence it remains to deal with the case of two or more bad edges. Firstly, using the properties of a convex drawing we prove that every pair of bad edges appears in a nested structure as illustrated in Figure 5 where for every witness w of a bad edge $\{v, v+1\}$, we have $w < v$. For this we may have to relabel the vertices cyclically and choose a different outer face. Next, the nesting property implies that we can label the bad edges as $b_1, \ldots, b_m$ for some $m \geq 2$, such that if $b_i = \{v_i, v_i + 1\}$, then $1 < v_1 < v_2 < \ldots < v_m = n-2$. Moreover, let w_i^L and w_i^R be the leftmost and the rightmost witnesses of the bad edge b_i, and by $L_i = \{x \in [n-1] : w_{i+1}^R < x < w_i^L\}$ and $R_i = \{x \in [n-1] : v_i + 1 \leq x \leq v_{i+1}\}$ be the left and the right blocks of vertices between two consecutive bad edges b_i and b_{i+1}; see Figure 5.

To construct the desired plane Hamiltonian cycle of non-star-crossing edges, we begin with the edge $\{v_\star, v_1\}$ and then iteratively add a plane path from v_i to v_{i+1} that includes all previously unvisited vertices from L_{i-1}, all the vertices of R_i and some vertices of L_i. When reaching the vertex $v_m + 1 = n-1$, we close the plane Hamiltonian path by adding edge $\{v_m + 1, v_\star\}$. Figure 6 illustrates the simple case, where all L_i's are empty.

In general, however, the L_i's are not empty and the procedure is more involved. The path from v_i to v_{i+1} consists of a subpath from v_i to $v_i + 1$ and a subpath from $v_i + 1$ to v_{i+1}.

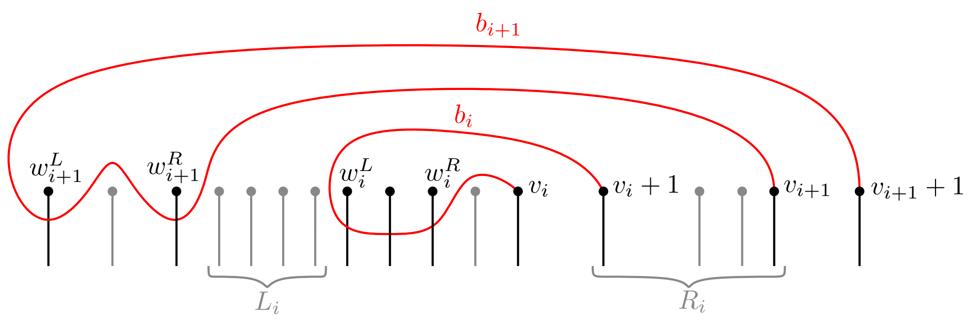


Figure 5 An illustration of nesting of two bad edges, where bad edges are depicted in red.

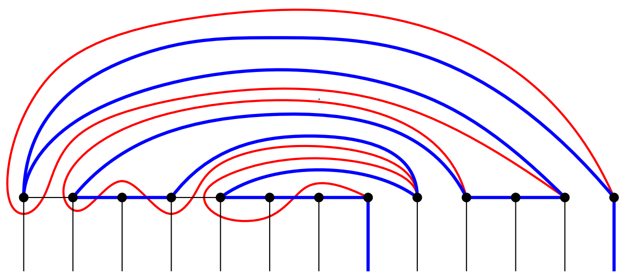


Figure 6 Example of a plane Hamiltonian cycle (blue) in the case where $L_i = \emptyset$ for all i.

For the former path (from v_i to $v_i + 1$), we go from v_i to w_i^R via the previously unvisited vertices from L_{i-1} and then one by one with decreasing labels to w_i^L. As long as there are vertices in L_i from which we can return to $v_i + 1$ via a non-star-crossing edge, we traverse these vertices and then return to $v_i + 1$. The latter path (from $v_i + 1$ to v_{i+1}) traverses all the vertices of R_i and some further vertices of L_i, and lies entirely in the region between the two bad edges b_i and b_{i+1}. ◀

2 Variations

We also studied variations of plane Hamiltonian structures which we briefly mention below.

Extending Hamiltonian cycles

Another way to interpret Theorem 1.3 is the following: given a spanning star in a convex drawing of K_n, we can extend this star to a plane Hamiltonian subdrawing of size $2n - 3$. As a variant of this formulation, we tested whether the other direction is true, i.e., whether any given plane Hamiltonian cycle can be extended to a plane subdrawing with $2n - 3$ edges. We verified that such an extension is possible for $n \leq 10$.

▶ **Conjecture 2.1.** *Let D be a convex drawing of K_n. Then every plane Hamiltonian cycle can be extended to a plane Hamiltonian subgraph on $2n - 3$ edges.*

Hamiltonian cycles avoiding a matching

Instead of a prescribed spanning star, we can also prescribe a matching and ask whether there is a Hamiltonian cycle that together with the matching builds a plane Hamiltonian substructure, i.e., the edges of the matching are not crossed by the Hamiltonian cycle. Hoffman and Tóth [13] investigated the geometric setting. They showed that for every plane perfect matching M in a geometric drawing of K_n there exists a plane Hamiltonian cycle that (possibly contains edges of M but) does not cross any edge from M. We believe that the same holds for convex drawings as well, and have verfied it for $n \leq 11$.

▶ **Conjecture 2.2.** *For every plane (not necessarily perfect) matching M in a convex drawing of K_n there exists a plane Hamiltonian cycle that (possibly contains edges of M but) does not cross any edge from M.*

Hamiltonian paths with a prescribed edge

In general, it is not possible to find a Hamiltonian cycle containing a prescribed edge in simple drawings. Even if we only ask for a Hamiltonian path containing a prescribed edge there is an easy example: consider the edge $\{1, 5\}$ in the perfect twisted T_5 depicted in Figure 4. However, the relaxation to Hamiltonian paths seems to be true for convex drawings. We have verified it for $n \leq 11$.

▶ **Conjecture 2.3.** *Let D be a convex drawing of K_n and let e be an edge of D. Then D has a plane Hamiltonian path containing the edge e.*

3 Discussion

We remark that our SAT-based investigation of substructures certainly surpasses the enumerative approach from [2]. While their approach required the enumeration and testing of 7 billion rotation systems on 9 vertices, our framework allows us to make investigations for up to $n = 20$ vertices with reasonably small resources. For example, most computations for $n = 9$ only took about 1 CPU hour and 200MB RAM. Even though exact computation times are not given in [2], the fact that the database for $n \leq 9$ covers about 1TB of disk space shows that our approach works with significantly less resources. Moreover, an enumerative approach for $n = 10$ (e.g. to test the existence of plane Hamiltonian cycles) would not be possible in reasonable time with contemporary computers because there are way too many rotation systems to be enumerated and tested.

Using our framework, we also studied uncrossed edges, crossing families, and empty triangles in simple drawings of the complete graph and believe that SAT-based investigations can be useful to make advancements in the study of simple drawings.

References

1 Supplemental source code and data. `https://page.math.tu-berlin.de/~scheuch/supplemental/simple_drawings/rotsys_supplemental1.zip`.

2 B. M. Ábrego, O. Aichholzer, S. Fernández-Merchant, T. Hackl, J. Pammer, A. Pilz, P. Ramos, G. Salazar, and B. Vogtenhuber. All Good Drawings of Small Complete Graphs. In *Proc. 31st European Workshop on Computational Geometry EuroCG '15*, pages 57–60, 2015.

3 O. Aichholzer, A. García, J. Tejel, B. Vogtenhuber, and A. Weinberger. Twisted ways to find plane structures in simple drawings of complete graphs. In *38th International Symposium on Computational Geometry (SoCG 2022)*, volume 224 of *LIPIcs*, pages 5:1–5:18. Dagstuhl, 2022.

4 A. Arroyo, D. McQuillan, R. B. Richter, and G. Salazar. Levi's Lemma, pseudolinear drawings of K_n, and empty triangles. *Journal of Graph Theory*, 87(4):443–459, 2018.

5 A. Arroyo, D. McQuillan, R. B. Richter, and G. Salazar. Convex drawings of the complete graph: topology meets geometry. *Ars Mathematica Contemporanea*, 22(3), 2022.

6 H. Bergold, S. Felsner, M. Scheucher, F. Schröder, and R. Steiner. Topological Drawings meet Classical Theorems from Convex Geometry. *In Discrete & Computational Geometry*, 2022.

7 A. Biere. PicoSAT essentials. *Journal on Satisfiability, Boolean Modeling and Computation (JSAT)*, 4:75–97, 2008.

8 A. Biere. CaDiCaL at the SAT Race 2019. In *Proc. of SAT Race 2019 – Solver and Benchmark Descriptions*, volume B-2019-1 of *Department of Computer Science Series*, pages 8–9. University of Helsinki, 2019.

9 J. Fox and B. Sudakov. Density theorems for bipartite graphs and related Ramsey-type results. *Combinatorica*, 29:153–196, 2009.

10 R. Fulek. Estimating the number of disjoint edges in simple topological graphs via cylindrical drawings. *SIAM Journal on Discrete Mathematics*, 28(1):116–121, 2014.

11 R. Fulek and A. 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. ACM, 2013.

12 A. García Olaverri, J. Tejel Altarriba, and A. Pilz. On plane subgraphs of complete topological drawings. *Ars Mathematica Contemporanea*, 20(1):69–87, 2021.

13 M. Hoffmann and C. D. Tóth. Segment endpoint visibility graphs are Hamiltonian. *Computational Geometry*, 26(1):47–68, 2003.

14 J. Kynčl. Simple realizability of complete abstract topological graphs simplified. *Discrete & Computational Geometry*, 64:1–27, 2020.

15 J. Pach, J. Solymosi, and G. Tóth. Unavoidable configurations in complete topological graphs. *Discrete & Computational Geometry*, 30(2):311–320, 2003.

16 J. Pach and G. Tóth. Disjoint edges in topological graphs. In *Combinatorial Geometry and Graph Theory*, volume 3330 of *LNTCS*, pages 133–140. Springer, 2005.

17 N. H. Rafla. *The good drawings D_n of the complete graph K_n*. PhD thesis, McGill University, Montreal, 1988.

18 A. J. Ruiz-Vargas. Many disjoint edges in topological graphs. *Computational Geometry*, 62:1–13, 2017.

19 A. Suk. Disjoint edges in complete topological graphs. In *Proceedings of the Twenty-Eighth Annual Symposium on Computational Geometry*, SoCG '12, pages 383–386. ACM, 2012.

20 A. Suk and J. Zeng. Unavoidable patterns in complete simple topological graphs. In *Graph Drawing and Network Visualization*, volume 13764 of *LNCS*, pages 3–15. Springer, 2022.

21 N. Wetzler, M. J. H. Heule, and W. A. Hunt. DRAT-trim: Efficient checking and trimming using expressive clausal proofs. In *Theory and Applications of Satisfiability Testing – SAT 2014*, volume 8561 of *LNCS*, pages 422–429. Springer, 2014.

Many views of planar point sets*

Jan Kynčl[1] **and Jan Soukup**[1]

1 **Department of Applied Mathematics, Charles University, Faculty of Mathematics and Physics, Malostranské nám. 25, 118 00 Praha 1, Czech Republic**
`{kyncl,soukup}@kam.mff.cuni.cz`

Abstract

Given a set P of n points in the plane and two points x and y not in P, such that their union is in general position, we say that x and y have the *same view* of P if the points of P are visible in the same cyclic order from x and y. We show that for every set P of n points in strong general position in the plane, there are $\Omega(n^4)$ points with mutually distinct views of P, confirming a conjecture by Díaz-Báñez, Fabila-Monroy and Pérez-Lantero.

1 Introduction

Let P be a set of n points in the plane in general position (no three points of P are collinear). Let x be an additional reference point not in P, such that the set $P \cup \{x\}$ is in general position. The *view* of P from x, also called the *radial ordering* of P with respect to x [6], is the clockwise cyclic order in which a rotating ray with origin at x meets the points of P. How many points with mutually distinct views of P are there?

It is easy to see that the number of mutually distinct views of P is always at most $O(n^4)$: indeed, consider the arrangement of $\binom{n}{2}$ lines drawn through every pair of points from P, and observe that two points within the same cell of this arrangement have the same view of P [1, 4, 5, 6, 9, 10]. On the other hand, there are constructions of sets of n points with $\Omega(n^4)$ distinct views [6, 10]. The problem of estimating the maximum possible number of distinct views of a set of n points in the plane also appears as Exercise 6.1.7 in Matoušek's textbook [7].

How many distinct views are guaranteed in an arbitrary point set P? Díaz-Báñez, Fabila-Monroy and Pérez-Lantero [6] proved that an arbitrary set P of n points in strong general position in the plane has $\Omega(n^3)$ points with mutually distinct views of P, and conjectured that this lower bound can be improved to $\Omega(n^4)$. Here the condition of *strong general position* means that P is in general position, and three distinct lines determined by pairs of points in P cannot cross at a common point except at a point from P. Bieri and Schmidt [3] conjectured that the number of views of any set of n points in general position is $\Theta(n^4)$, and similarly for the number of "projective" cyclic orders obtained by rotating a line (rather than a ray), which we might call *projective views*.

We confirm these conjectures for point sets in strong general position.

▶ **Theorem 1.1.** *For every set P of n points in strong general position in the plane, there are $\Omega(n^4)$ points with mutually distinct views of P.*

We prove Theorem 1.1 in the following, slightly stronger, form, which guarantees the lower bound also for projective views.

* Supported by project 21-32817S of the Czech Science Foundation (GAČR) and by the grant SVV–2020–260578.

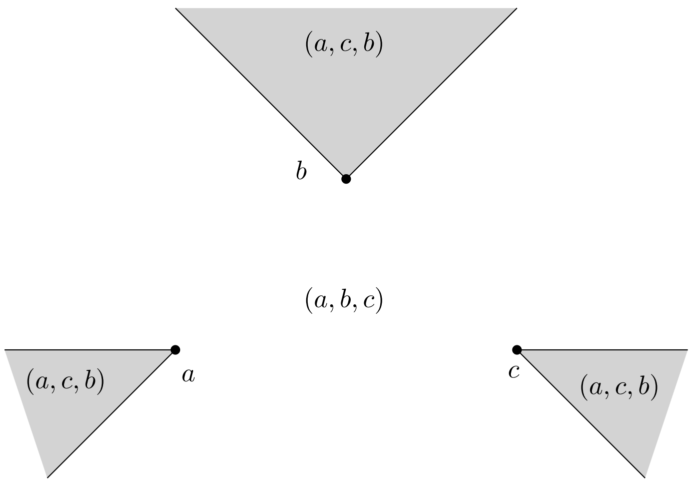


Figure 1 Six rays determined by three points a, b, c split the plane into four regions; the three “corner” regions have the same view (a, c, b), the remaining region has the view (a, b, c).

▶ **Theorem 1.2.** *For every set P of n points in strong general position in the plane, there is a subset $X \subseteq P$ and a set Y of $\Omega(n^4)$ points with mutually distinct views of X, such that the convex hull of Y is disjoint with the convex hull of X.*

We prove Theorem 1.2 in Section 3. The main idea of the proof is that every point set in general position contains a subset of linear size that “looks like” the special construction of a point set with $\Omega(n^4)$ views [6, 10].

2 Preliminaries

For every pair of distinct points a, b, we denote the line passing through them by $\overleftrightarrow{ab}$. Additionally, we denote the ray contained in $\overleftrightarrow{ab}$ with endpoint b and not containing a by $ab\rightarrow$, and the ray contained in $\overleftrightarrow{ab}$ with endpoint a and containing b by $\overrightarrow{ab}$.

Let $k \geq 3$ and let C be a sequence $x_1x_2 \dots x_k$ of k points in the plane, ordered by increasing x-coordinate. We say that C is a *concave chain* (also called a *k-cap*) if for every pair of consecutive points $x_i, x_{i+1} \in C$, all the remaining points of C lie below the line $\overleftrightarrow{x_ix_{i+1}}$. In particular, the points of C form the vertices of a convex polygon, ordered clockwise along its boundary. Similarly, we say that C is a *convex chain* (also called a *k-cup*) if for every pair of consecutive points $x_i, x_{i+1} \in C$, all the remaining points of C lie above the line $\overleftrightarrow{x_ix_{i+1}}$.

Let P be a set of n points in the plane. Let p, q be additional reference points not in P, such that the set $P \cup \{p, q\}$ is in general position. If the view of P from p is distinct from the view of P from q, then p and q have distinct views of some triple of points from P. The view of a three-point set $\{a, b, c\} \subseteq P$ forming a triangle with vertices in clockwise order (a, b, c) is determined by the arrangement of rays $ab\rightarrow, ba\rightarrow, ac\rightarrow, ca\rightarrow, bc\rightarrow, cb\rightarrow$. This arrangement has four cells, the view from three of these cells is (a, c, b), and the view from the remaining cell is (a, b, c); see Figure 1.

In the proof of Theorem 1.2, we will use the Positive Fraction Erdős–Szekeres Theorem by Bárány and Valtr [2, 8]. A finite planar point set is called a *convex k-clustering* if it is a disjoint union of k sets $X_1, \dots, X_k$ of equal sizes such that every k-tuple $(x_1, \dots, x_k)$, $x_1 \in X_1, \dots, x_k \in X_k$, forms a convex polygon with vertices in clockwise order $(x_1, \dots, x_k)$.

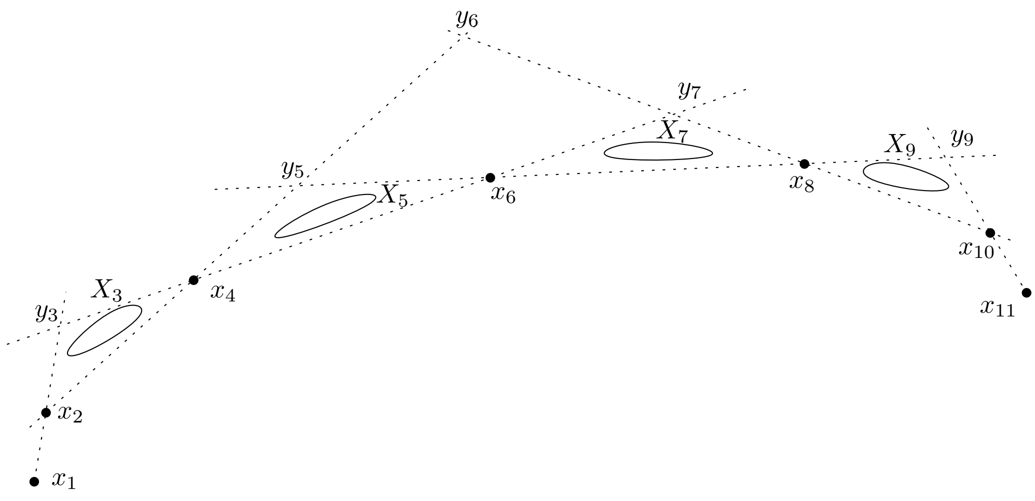


Figure 2 A configuration of linear size that exists in any point set of size n in general position.

▶ **Theorem 2.1** (Positive Fraction Erdős–Szekeres Theorem [2, 8]). *For every $k \geq 3$ there is an $\varepsilon_k > 0$ and an integer $f(k)$ such that if X is a finite set of points in general position in the plane with $|X| \geq f(k)$, then it contains a convex k-clustering of size at least $\varepsilon_k|X|$.*

3 Proof of Theorem 1.2

Let P be a set of n points in the plane in strong general position. By Theorem 2.1 there exists a convex 20-clustering of P of size $\Omega(n)$. In other words, there exist mutually disjoint subsets $X_1, \dots, X_{20} \subset P$, such that all X_i are of equal size $s = \Omega(n)$, and every 20-tuple $(x_1, \dots, x_{20})$ with $x_1 \in X_1, \dots, x_k \in X_{20}$ forms a convex 20-gon with vertices in clockwise order $(x_1, \dots, x_{20})$. We fix one such tuple $(x_1, \dots, x_{20})$. Eleven points from this tuple form either a convex or a concave chain; without loss of generality, let the points $x_1, \dots, x_{11}$ form a concave chain $x_1x_2 \dots x_{11}$. Let X be the subset of P containing the points $x_1, x_2, x_4, x_6, x_8, x_{10}, x_{11}$ and all the points in $X_3 \cup X_5 \cup X_7 \cup X_9$; see Figure 2 for an illustration of X. We will show that there exist $\Omega(n^4)$ distinct views of X.

We define five intersection points y_3, y_5, y_7, y_9, y_6 as follows:

$$\begin{aligned}
\{y_3\} &= x_1x_2{\rightarrow} \cap x_6x_4{\rightarrow},\\
\{y_5\} &= x_2x_4{\rightarrow} \cap x_8x_6{\rightarrow},\\
\{y_7\} &= x_4x_6{\rightarrow} \cap x_{10}x_8{\rightarrow},\\
\{y_9\} &= x_6x_8{\rightarrow} \cap x_{11}x_{10}{\rightarrow},\\
\{y_6\} &= x_2x_4{\rightarrow} \cap x_{10}x_8{\rightarrow}.
\end{aligned}$$

Since $x_1x_2x_4x_6x_8x_{10}x_{11}$ is a concave chain, all these intersections are well-defined. Let L be the set of all rays $l_3l_5{\rightarrow}$ where $l_3 \in X_3$ and $l_5 \in X_5$, and let R be the set of all rays $r_9r_7{\rightarrow}$ where $r_9 \in X_9$ and $r_7 \in X_7$. The sizes of L and R are both $s^2 = \Omega(n^2)$.

▶ **Observation 3.1.** *For every selection $l_3 \in X_3$, $l_5 \in X_5$, $r_7 \in X_7$, $r_9 \in X_9$, the sequence $x_1x_2l_3x_4l_5x_6r_7x_8r_9x_{10}x_{11}$ is a concave chain.*

Proof. The tuple $(x_1, x_2, l_3, x_4, l_5, x_6, r_7, x_8, r_9, x_{10}, x_{11})$ forms a convex 11-gon with vertices

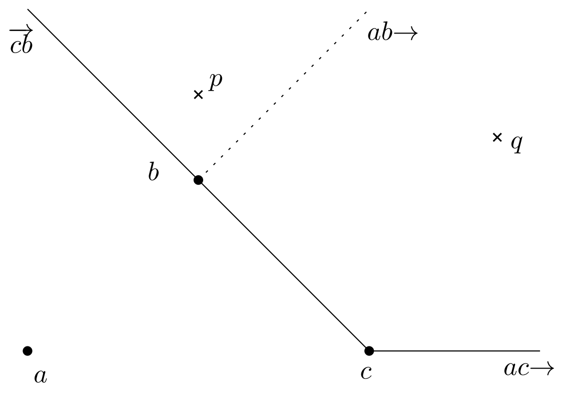


Figure 3 The points p and q on the opposite side of $ab\!\rightarrow$ and inside the convex region bounded by $\overrightarrow{cb}$ and $ac\!\rightarrow$ have distinct views of $\{a, b, c\}$.

in clockwise order by the selection of X. Since x_1, x_2, x_{10}, x_{11} are fixed and form a concave chain, it follows that $x_1x_2l_3x_4l_5x_6r_7x_8r_9x_{10}x_{11}$ is a concave chain. ◀

An immediate consequence is the following observation.

▶ **Observation 3.2.** *For every $i \in \{3, 5, 7, 9\}$ the set X_i lies inside the triangle $x_{i-1}y_ix_{i+1}$. Moreover, every ray from L intersects segments x_6y_5 and y_7y_6, and every ray from R intersects segments x_6y_7 and y_5y_6.*
Furthermore, every ray from L intersects with every ray from R inside the quadrilateral $x_6y_5y_6y_7$.

We now state the only tool we are using to distinguish different views.

▶ **Lemma 3.3.** *Let a, b, c, p, q be points in the plane in general position. Assume that abc is a triangle with vertices in clockwise order (a, b, c), the points p and q lie in the convex region bounded by $\overrightarrow{cb}$ and $ac\!\rightarrow$, and p lies on the opposite side of $ab\!\rightarrow$ from q. Then the view of $\{a, b, c\}$ from p is distinct from the view from q.*

Proof. The situation is illustrated in Figure 3. When we look at the cells in the arrangement of rays $ab\!\rightarrow, ba\!\rightarrow, ac\!\rightarrow, ca\!\rightarrow, bc\!\rightarrow, cb\!\rightarrow$ (as in Figure 1), we immediately see that p and q lie in cells with distinct views of $\{a, b, c\}$. ◀

It remains to find the reference points with distinct views of X. We look at the arrangement $\mathcal{L}$ of rays and lines from $L \cup R \cup \{\overleftrightarrow{x_6y_7}, \overleftrightarrow{y_5x_6}, \overleftrightarrow{y_6y_5}, \overleftrightarrow{y_6y_7}\}$ and for each cell in this arrangement that lies in the quadrilateral $x_6y_5y_6y_7$ we select a reference point from the interior of this cell. To complete the proof of the theorem, we need to prove that we selected $\Omega(n^4)$ points and that any two selected reference points have distinct views.

▶ **Observation 3.4.** *There are $\Omega(n^4)$ cells in the arrangement $\mathcal{L}$ that lie in the quadrilateral $x_6y_5y_6y_7$.*

Proof. The situation is illustrated in Figure 4. The quadrilateral $x_6y_5y_6y_7$ is a cell in the arrangement of lines $\overleftrightarrow{x_6y_7}, \overleftrightarrow{y_5x_6}, \overleftrightarrow{y_6y_5}, \overleftrightarrow{y_6y_7}$. Let r be a ray from R. By Observation 3.2, r intersects segments x_6y_7 and y_5y_6. Hence, r splits the quadrilateral $x_6y_5y_6y_7$ into two parts.

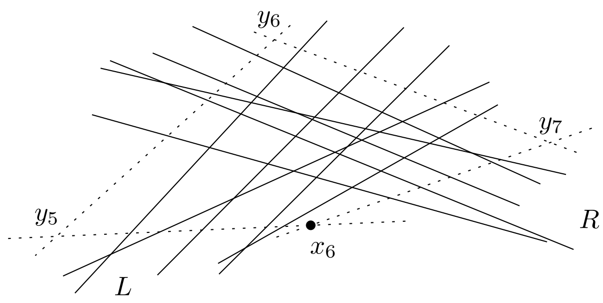


Figure 4 Sets of rays L and R split the quadrilateral $x_6y_5y_6y_7$ into a "grid-like" structure.

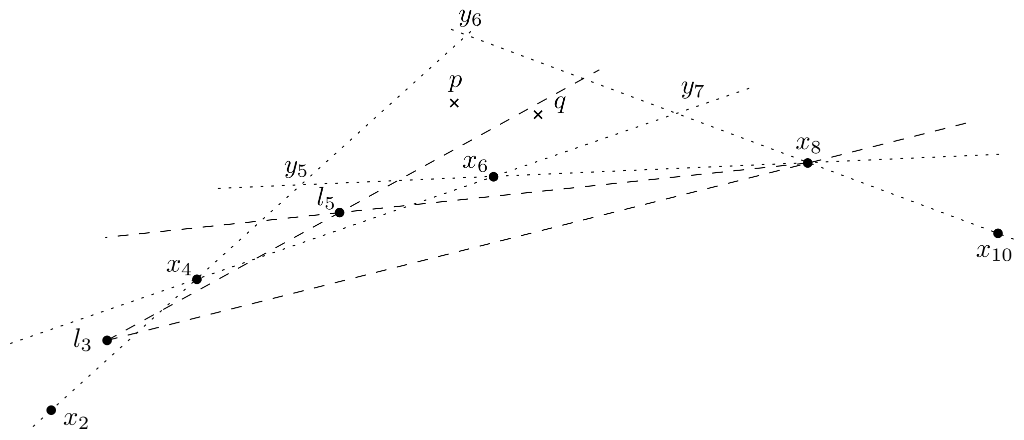


Figure 5 A situation from Observation 3.5. Points p and q have distinct views of the point set $\{l_3, l_5, x_8\}$.

Since there are $s^2 = \Omega(n^2)$ rays in R, the rays in R split the quadrilateral $x_6y_5y_6y_7$ into at least $s^2 + 1$ parts.

Let l be a ray from L. By Observation 3.2, l intersects segments y_5x_6 and y_6y_7. Additionally, by the same observation, it intersects with all the rays from R inside the quadrilateral $x_6y_5y_6y_7$. Since the points of X are in strong general position, all these intersections are distinct. Hence, l splits at least $s^2 + 1$ regions inside $x_6y_5y_6y_7$ bordered by $x_6y_5y_6y_7$ and rays from R. Since there are $s^2 = \Omega(n^2)$ rays in L, the whole arrangement $\mathcal{L}$ has at least $(s^2 + 1) \cdot (s^2 + 1) = \Omega(n^4)$ cells inside the quadrilateral $x_6y_5y_6y_7$. ◀

▶ **Observation 3.5.** *Let p and q be two reference points from two distinct cells of the arrangement $\mathcal{L}$ inside the quadrilateral $x_6y_5y_6y_7$. Then the view of X from p is distinct from the view from q.*

Proof. Since p and q are from distinct cells inside the convex quadrilateral $x_6y_5y_6y_7$ the segment pq intersects some ray from $L \cup R$; without loss of generality, let it be a ray l from L. The ray l is equal to some $l_3l_5\!\rightarrow$, $l_3 \in X_3$, $l_5 \in X_5$. We now compare the views of $\{l_3, l_5, x_8\}$ from p and q (we would take x_4 instead of x_8 if the ray was from R). See Figure 5 for an illustration.

The sequence $x_2l_3x_4l_5x_6x_8x_{10}$ is a concave chain by Observation 3.1. Since the points x_6 and y_5 lie on the ray $x_8x_6\!\rightarrow$, it follows that the points x_6 and y_5 lie above the lines $\overleftrightarrow{l_3x_8}$

and $\overleftrightarrow{l_5x_8}$. Similarly, the points y_7 and y_6 also lie above the lines $\overleftrightarrow{l_3x_8}$ and $\overleftrightarrow{l_5x_8}$. Hence, the whole quadrilateral $x_6y_5y_6y_7$ lies above these lines and thus in the convex region bounded by rays $\overrightarrow{x_8l_5}$ and $l_3x_8\rightarrow$.

By Lemma 3.3 applied on the triangle $l_3l_5x_8$ and reference points p, q, the view of $\{l_3, l_5, x_8\}$ from p is distinct from the view from q. Hence, also the views of X from p and q are distinct. ◀

This concludes the proof of Theorem 1.2.

▶ Remark. By the hyperplane separation theorem, the quadrilateral $x_6y_5y_6y_7$ can be separated from X by a line. Since the reference points are chosen inside the quadrilateral $x_6y_5y_6y_7$, the projective views of X from these reference points are the same as the views, and therefore also pairwise distinct.

References

1 Thomas Auer and Martin Held. Heuristics for the generation of random polygons. In *Proceedings of the 8th Canadian Conference on Computational Geometry*, pages 38–43, Ottawa, Ontario, Canada, 1996.

2 Imre Bárány and Pavel Valtr. A positive fraction Erdős–Szekeres theorem. *Discrete Comput. Geom.*, 19(3, Special Issue):335–342, 1998. Dedicated to the memory of Paul Erdős. doi:10.1007/PL00009350.

3 Hanspeter Bieri and Peter-Michael Schmidt. On the permutations generated by rotational sweeps of planar point sets. In *Proceedings of the 8th Canadian Conference on Computational Geometry*, pages 179–184, Ottawa, Ontario, Canada, 1996.

4 Linda Deneen and Gary Shute. Polygonizations of point sets in the plane. *Discrete Comput. Geom.*, 3(1):77–87, 1988. doi:10.1007/BF02187898.

5 Olivier Devillers, Vida Dujmović, Hazel Everett, Samuel Hornus, Sue Whitesides, and Steve Wismath. Maintaining visibility information of planar point sets with a moving viewpoint. *Internat. J. Comput. Geom. Appl.*, 17(4):297–304, 2007. doi:10.1142/S0218195907002343.

6 José Miguel Díaz-Báñez, Ruy Fabila-Monroy, and Pablo Pérez-Lantero. On the numbers of radial orderings of planar point sets. *Discrete Math. Theor. Comput. Sci.*, 16(3):291–304, 2014.

7 Jiří Matoušek. *Lectures on discrete geometry*, volume 212 of *Graduate Texts in Mathematics*. Springer-Verlag, New York, 2002. doi:10.1007/978-1-4613-0039-7.

8 Attila Pór and Pavel Valtr. The partitioned version of the Erdős–Szekeres theorem. *Discrete Comput. Geom.*, 28(4):625–637, 2002. Discrete and computational geometry and graph drawing (Columbia, SC, 2001). doi:10.1007/s00454-002-2894-1.

9 Benjamín Tovar, Luigi Freda, and Steven M. LaValle. Using a robot to learn geometric information from permutations of landmarks. In *Topology and robotics*, volume 438 of *Contemp. Math.*, pages 33–45. Amer. Math. Soc., Providence, RI, 2007. doi:10.1090/conm/438/08443.

10 Roland Ulber. On the number of star-shaped polygons and polyhedra. In *Proceedings 11th Canadian Conference on Computational Geometry CCCG'99*, page 4 pp, Vancouver, Canada, 1999.

Sometimes Two Irrational Guards are Needed

Lucas Meijer[1] and Till Miltzow[2]

1 Department of Information and Computing Sciences, Utrecht University
`l.meijer2@uu.nl`

2 Department of Information and Computing Sciences, Utrecht University
`t.miltzow@uu.nl`

Abstract

In the art gallery problem, we are given a closed polygon P, with rational coordinates and an integer k. We are asked whether it is possible to find a set (of guards) G of size k such that any point $p \in P$ is seen by a point in G. We say two points p, q see each other if the line segment pq is contained inside P. It was shown by Abrahamsen, Adamaszek, and Miltzow that there is a polygon that can be guarded with three guards, but requires four guards if the guards are required to have rational coordinates. In other words, an optimal solution of size three might need to be irrational. We show that an optimal solution of size two might need to be irrational. Note that it is well-known that any polygon that can be guarded with one guard has an optimal guard placement with rational coordinates.

Hence, our work closes the gap on when irrational guards are possible to occur.

Related Version arXiv:2212.01211

39th European Workshop on Computational Geometry, Barcelona, Spain, March 29–31, 2023.
This is an extended abstract of a presentation given at EuroCG'23. 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.

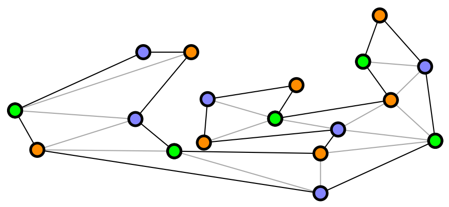


Figure 1 Any triangulation of a simple polygon can be three-colored. At least one of the color classes has at most $\lfloor n/3 \rfloor$ vertices. This color class also guards the entire polygon, as every triangle is incident to all three colors [7].

1 Introduction

In the art gallery problem, we are given a closed polygon P, on n vertices, with rational coordinates and an integer k. We are asked whether it is possible to find a set (of guards) G of size k such that any point $p \in P$ is seen by a point in G. We say two points p, q see each other if the line segment pq is contained inside P.

We show that an optimal solution of two guards might need to have irrational coordinates. In such a case, we say a polygon has *irrational guards*. Specifically, we construct a polygon that can be guarded by two irrational guards but requires three rational guards.

The art gallery problem was formulated in 1973 by Victor Klee. See, for example, the book by O'Rourke [8, page 2]. One of the earliest results states that every simple polygon on n vertices can always be guarded with $\lfloor n/3 \rfloor$ guards [4, 7].

Interestingly, it is actually very tough to find any positive algorithmic results on the art gallery problem. It seems like the art gallery problem is almost impenetrable. For instance, only in 2002, Micha Sharir pointed out that the problem was even decidable [5, 6, see acknowledgments]. The decidability of the art gallery problem is actually easy once you know methods from real algebraic geometry [3]. The idea is to reduce the problem to the first-order theory of the reals. We encode guard positions by variables, and then we check if every point in the polygon is seen by at least one guard. Note that this is easy to encode in the first-order theory of the reals, as we are allowed to use existential $(\exists g_1, g_2, \ldots)$ and universal quantifiers $(\forall p = (x, y))$. Since then, despite much research on the art gallery problem, no better algorithm appeared, as far as worst-case complexity is concerned. The underlying reason for the difficulty to find better algorithms can be explained by the fact that the art gallery problem is $\exists\mathbb{R}$-complete [9, 2]. In a nutshell, $\exists\mathbb{R}$-completeness precisely entails that there is no better method for the worst-case complexity of the problem. ($\exists\mathbb{R}$ can be defined as the class of problems that are equivalent to finding a real root to a multivariate polynomial with integer coordinates. See the full version for an introduction.) More specifically, it was shown that arbitrary algebraic numbers may be needed to describe an optimal solution to the art gallery problem. This may come as a surprise to some readers, and was clearly a surprise back then. Specifically, "in practice", it seems very rare that irrational guards are ever needed. The reason is that a typical situation is one of the following two. Either the guards have some freedom to move around and still see the entire polygon. Or if a guard has no freedom, it is forced to be on a line defined by vertices of the polygon. As the vertices of the polygon are at rational coordinates, the guards will be at rational coordinates in that case as well. Indeed, only in 2017, the first polygon requiring irrational guards was found [1]. Even though $\exists\mathbb{R}$-reductions exhibit an infinite number of polygons that require irrational guards, those polygons are not "concrete" in the naive sense of the word. And up to this day, this is the only "concrete" polygon [1] that we know does require irrational guards. In this work, we find a second polygon. It is superior to the first one in the sense that it shows that two guards are already enough to enforce irrational guards. As a single guard can always be chosen to have rational coordinates, we settle the question of the minimum number of guards required to have irrational guards. We summarize our results in the following theorem.

▶ **Theorem 1.1.** *There exists a polygon with rational coordinates, such that there is only one way of guarding this polygon optimally with two guards. Those two guards have irrational coordinates.*

Organization.

We provide background information in the full version. There we discuss our results from different angles, we give a selected overview of related research on the art gallery problem, and we add some background on the existential theory of the reals. In Section 2, we give an overview of how we constructed the polygon and what is the intuition behind the different parts. In Section 3, we give the polygon with coordinates of all vertices. Finally, in the full version we also provide a formal proof of correctness and explain how we constructed the polygon and what technical challenges we had to overcome.

2 Preparation

We aim to construct a polygon. This polygon should be guarded by two guards at irrational coordinates but requires three guards at rational coordinates. We must restrict the possible coordinates the guards can be positioned. In this section, we will explore the tools to restrict the possible positions of the two guards within the polygon.

2.1 Basic Definitions

Each guard g will be able to guard some region of the polygon: we call this region its *visibility polygon* $\text{vis}(g)$. The visibility polygon includes all points for which the line segment between the guard and the point is included in the polygon P. Notably, the union of the visibility polygons of the two guards must be the art gallery. Otherwise, the art gallery is not completely guarded.

A *window* is an edge of the visibility polygon $\text{vis}(g)$ that is not part of the boundary of P. We can find windows in the guard g's visibility polygon, by shooting rays from g to reflex vertices (the vertices of the polygon, with an interior angle larger than π). If these rays do not leave the polygon at the reflex vertex, a window will exist between the reflex vertex and the position where the ray does intersect the boundary of the polygon. Let the *window's end* be the intersection of the ray with an edge of the polygon.

Our final polygon consists of the core and a number of pockets, as shown in Figure 2. The *core* of the polygon is the square in the center. We will enforce that both guards are located in the core. As a square is a convex shape, this implies that both guards will guard the core. The *pockets* are all regions outside the core. We will use pockets that are either quadrilateral or triangular. Pockets are *attached* to either the core or another pocket: they have one edge that lies on the boundary of the core or on the boundary of another pocket. Quadrilateral pockets will always be attached to the core. Each quadrilateral pocket has one edge that is not on the boundary of the core, nor adjacent to it. We will call this edge the *wall* of a quadrilateral pocket. Similarly, triangular pockets will be attached to either the core or a quadrilateral pocket. We will use pockets as a tool to limit the locations of the two guards.

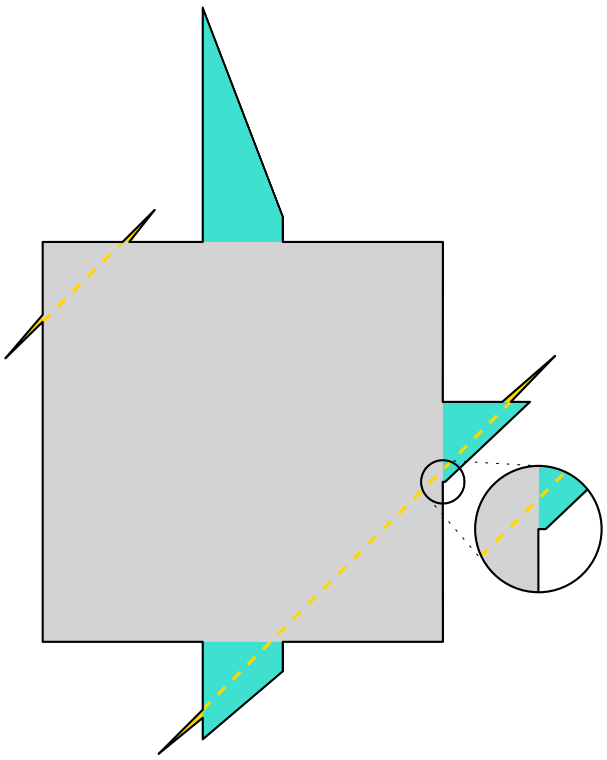


Figure 2 Our final polygon: it has a core (gray), three quadrilateral pockets (blue), and four narrow triangular pockets (yellow).

2.2 Guard Segments

We can force a guard to be positioned on a line segment within the polygon. Such a line segment is called a *guard segment*. Guard segments are commonly used in the context of the art gallery problem [1, 9]. In this section, we will describe how we construct a guard segment. We denote by s the segment and by ℓ its supporting line.

To make s a guard segment, we add two triangular pockets where ℓ intersects ∂P. Each of the triangular pockets has an edge on ℓ. Besides this one edge, the pockets lay on different sides of ℓ. Only a guard on the line segment between the two pockets can guard both triangular pockets at the same time.

We have two guards in our polygon and both will be on distinct guard segments. If the two guard segments are not intersecting, we can enforce that there must be one guard on each of them as follows. First, we introduce only four triangular pockets. Second, we make the triangular pockets sufficiently narrow. In this way, it is impossible to guard two of the triangular pockets outside of a guard segment. Thus at least one guard must be on each guard segment. A simple construction with two non-intersecting guard segments is shown in Figure 3.

2.3 Guarding Quadrilateral Pockets

We will now describe how given the position of guard l and a quadrilateral pocket Q will limit the position of guard t. See Figure 4 for an illustration of the following description. First, note that if l will not guard Q completely then there will remain some unguarded

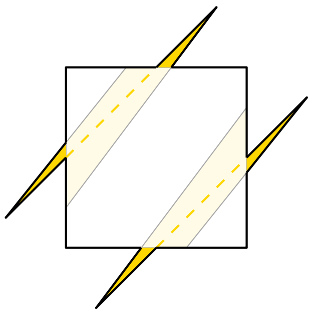


Figure 3 A small polygon that can only be guarded by two guards, because each guard segment (yellow dashed line) must contain a guard. The region where a guard could guard at least one pocket is shaded in light yellow.

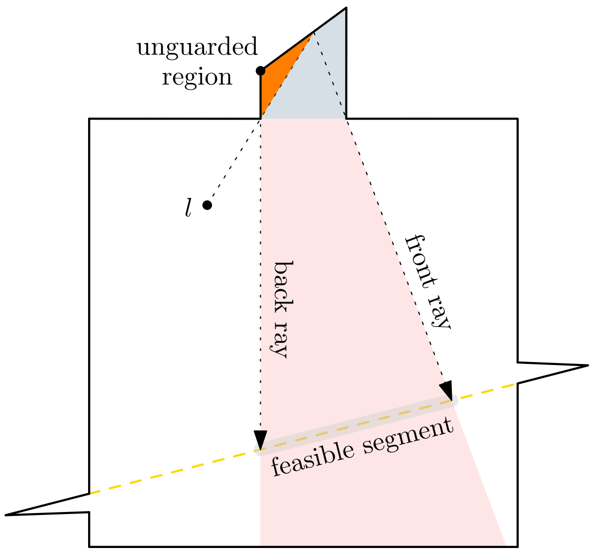


Figure 4 A polygon with guard l. The guard l defines an unguarded region in the quadrilateral pocket, a front ray and a back ray, and a feasible segment.

region (orange) in Q. The part of the guard segment of t where the unguarded region is visible is referred as the feasible segment. It is bounded from the back ray and the front ray. It is clear that t must be on the feasible segment.

We can compute the front ray by first computing the window end's s from l to the wall of Q and then shooting a ray from s in the direction of the second reflex vertex of Q.

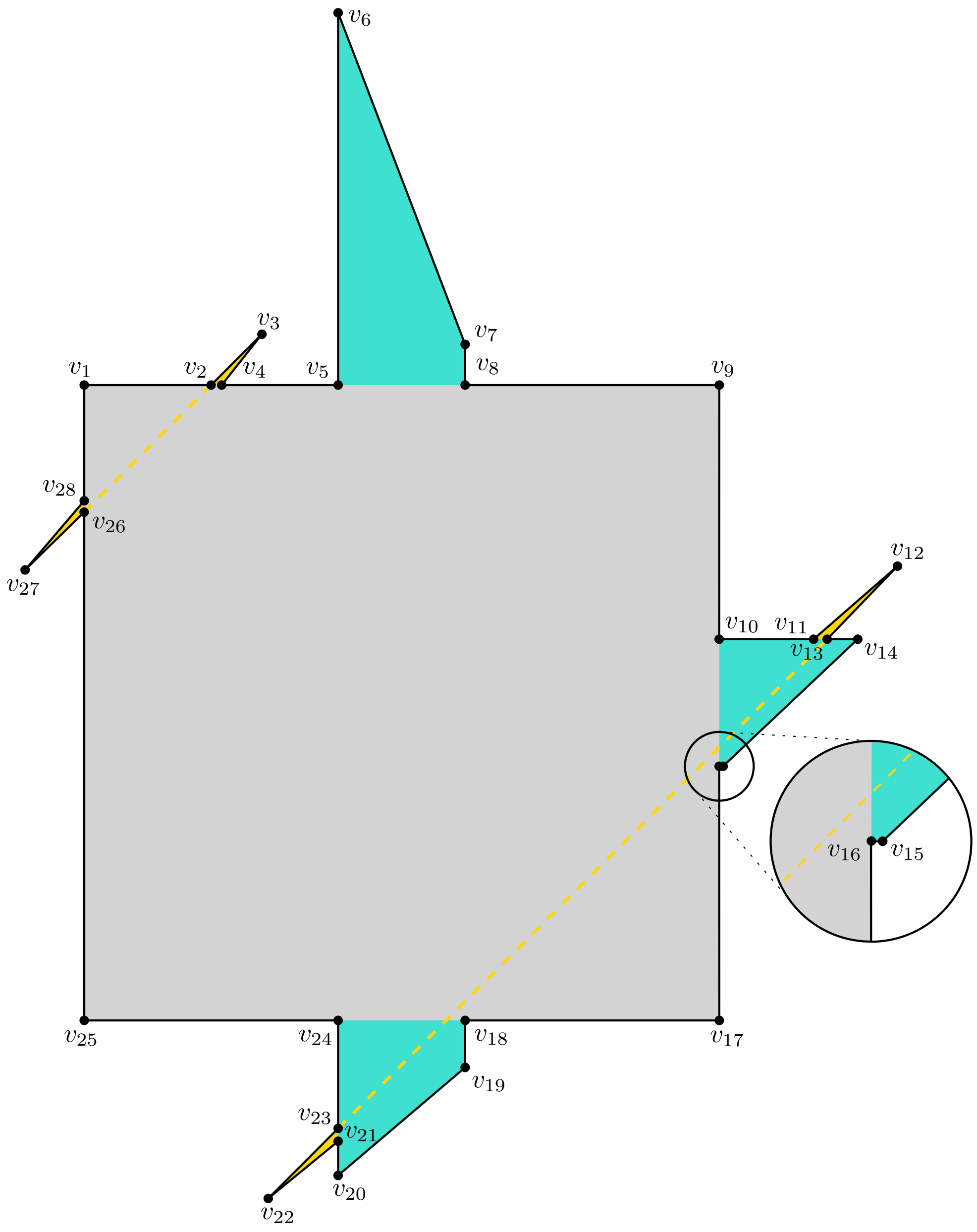


Figure 5 Our complete polygon. The art gallery is shaded according to the function of each region: gray is the core, yellow is the pockets used to create guard segments, and turquoise are other pockets. The yellow dashed lines represent the guard segments. The coordinates of important vertices are given.

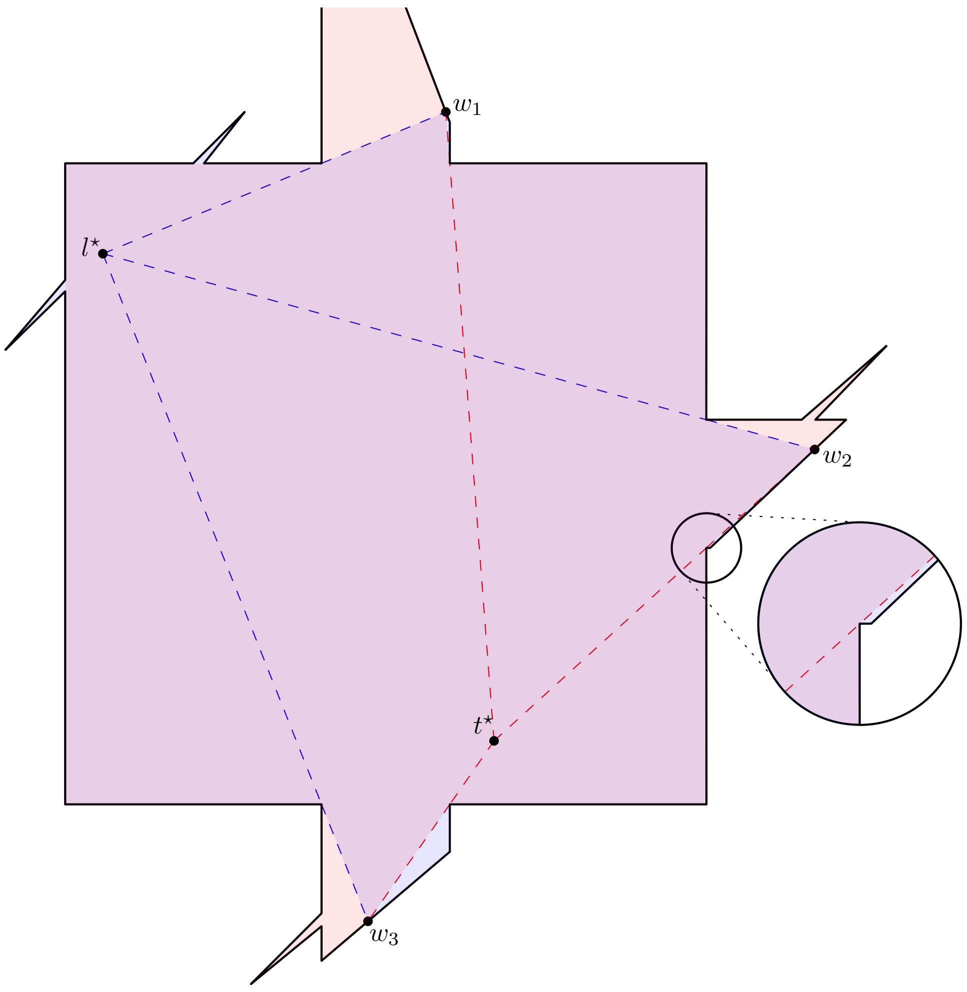


Figure 6 Our complete polygon. The optimal solution has two guards at irrational coordinates is shown. The light blue regions are guarded by the upper left guard; the light red regions are guarded by the bottom right guard; the purple (overlay of red and blue) regions are guarded by both. The dashed lines are rays shot from the guards through reflex vertices. For each pocket, these windows meet at a point on the art gallery's wall, of which the coordinates are also given.

Table 1 Coordinates of the vertices of the polygon ($v_1, \ldots, v_{28}$), the guards (l and t), and the window's ends (w_1, w_2, w_3).

v_1	$(0, 10)$	v_{12}	$(12.7, 7)$	v_{23}	$(4, -1.7)$
v_2	$(2, 10)$	v_{13}	$(11.7, 6)$	v_{24}	$(4, 0)$
v_3	$(3, 11)$	v_{14}	$(\frac{1230422}{101007}, 6)$	v_{25}	$(0, 0)$
v_4	$(2.3, 10)$	v_{15}	$(\frac{1016072}{101007}, 4)$	v_{26}	$(0, 8)$
v_5	$(4, 10)$	v_{16}	$(10, 4)$	v_{27}	$(-1, 7)$
v_6	$(4, \frac{465522}{29357})$	v_{17}	$(10, 0)$	v_{28}	$(0, 8.3)$
v_7	$(6, \frac{312388}{29357})$	v_{18}	$(6, 0)$	l^*	$(3.7 - 2.2 \cdot \sqrt{2}, 11.7 - 2.2 \cdot \sqrt{2})$
v_8	$(6, 10)$	v_{19}	$(6, \frac{-25442}{34407})$	t^*	$(7.4 - 0.5 \cdot \sqrt{2}, 1.7 - 0.5 \cdot \sqrt{2})$
v_9	$(10, 10)$	v_{20}	$(4, \frac{-84128}{34407})$	w_1	$(\frac{293570 \cdot \sqrt{2} + 8052346}{1425913}, \frac{-765670 \cdot \sqrt{2} + 16485384}{1425913})$
v_{10}	$(10, 6)$	v_{21}	$(4, 2)$	w_2	$(\frac{1071750 \cdot \sqrt{2} + 29733818}{2673483}, \frac{1010070 \cdot \sqrt{2} + 13370606}{2673483})$
v_{11}	$(11.4, 6)$	v_{22}	$(3, -2.7)$	w_3	$(\frac{344070 \cdot \sqrt{2} + 3108526}{760803}, \frac{293430 \cdot \sqrt{2} + 1804526}{760803})$

3 Complete Polygon

In this section, we will present our complete polygon: a polygon that can be guarded by two guards if and only if both guards are situated at irrational points.

3.1 The Polygon

As we described in Section 2 and displayed in Figure 5, the polygon consists of a core and some pockets. The polygon has four triangular pockets defining two guard segments. The two guard segments lie on the lines $y = x + 8$ and $y = x - 5.7$. Furthermore, the polygon has three quadrilateral pockets. In Table 1, the coordinates of the vertices of the polygon, the coordinates of the two guards, and the coordinates of the window's ends are given.

The walls of the three quadrilateral pockets have the supporting lines:

1. Top pocket: $y = \frac{-76567 \cdot x + 771790}{29357}$.
2. Right pocket: $y = \frac{101007 \cdot x - 587372}{107175}$.
3. Bottom pocket: $y = \frac{29343 \cdot x - 201500}{34407}$.

In the full version, we prove that this polygon can be guarded by two guards, if and only if the guards are at irrational coordinates in and we discuss the difficulties we encountered while searching for this polygon.

4 Acknowledgments.

We would like to thank Thekla Hamm and Ivan Bliznets for their helpful comments on the presentation. Lucas Meijer is generously supported by the Netherlands Organisation for Scientific Research (NWO) under project no. VI.Vidi.213.150. Tillmann Miltzow is generously supported by the Netherlands Organisation for Scientific Research (NWO) under project no. 016.Veni.192.250 and no. VI.Vidi.213.150.

References

1 Mikkel Abrahamsen, Anna Adamaszek, and Tillmann Miltzow. Irrational guards are sometimes needed. In *SoCG 2017*, pages 3:1–3:15, 2017. Arxiv 1701.05475.

2 Mikkel Abrahamsen, Anna Adamaszek, and Tillmann Miltzow. The Art Gallery Problem is ∃ℝ-complete. *Journal of the ACM*, 69(1):1–70, 2022. `doi:10.1145/3486220`.

3 Sauguta Basu, Richard Pollack, and Marie-Françoise Roy. *Algorithms in Real Algebraic Geometry*, volume 10 of *Algorithms and Computation in Mathematics*. Springer, Berlin, Heidelberg, 2006. `doi:10.1007/3-540-33099-2`.

4 Václav Chvátal. A combinatorial theorem in plane geometry. *Journal of Combinatorial Theory, Series B*, 18(1):39–41, 1975.

5 Alon Efrat and Sariel Har-Peled. Guarding galleries and terrains. In *IFIP 2002*, pages 181–192, 2002.

6 Alon Efrat and Sariel Har-Peled. Guarding galleries and terrains. *Inf. Process. Lett.*, 100(6):238–245, 2006. `doi:10.1016/j.ipl.2006.05.014`.

7 Steve Fisk. A short proof of Chvátal's watchman theorem. *J. Comb. Theory, Ser. B*, 24(3):374, 1978. URL: `http://dx.doi.org/10.1016/0095-8956(78)90059-X`, `doi:10.1016/0095-8956(78)90059-X`.

8 Joseph O'Rourke. *Art Gallery Theorems and Algorithms*. Oxford University Press, 1987.

9 Jack Stade. Complexity of the boundary-guarding art gallery problem. *arXiv preprint arXiv:2210.12817*, 2022.

On the Complexity of Recognizing Nerves of Convex Sets

Patrick Schnider[1] **and Simon Weber**[2]

1 **Department of Computer Science, ETH Zurich**
`patrick.schnider@inf.ethz.ch`
2 **Department of Computer Science, ETH Zurich**
`simon.weber@inf.ethz.ch`

Abstract

We study the problem of recognizing whether a given abstract simplicial complex K is the k-skeleton of the nerve of j-dimensional convex sets in $\mathbb{R}^d$. We denote this problem by $R(k, j, d)$. As a main contribution, we unify the results of many previous works under this framework and show that many of these works in fact imply stronger results than explicitly stated. This allows us to settle the complexity status of $R(1, j, d)$, which is equivalent to the problem of recognizing intersection graphs of j-dimensional convex sets in $\mathbb{R}^d$, for any j and d. Furthermore, we point out some trivial cases of $R(k, j, d)$, and demonstrate that $R(k, j, d)$ is $\exists\mathbb{R}$-complete for $j \in \{d-1, d\}$ and $k \geq d$.

1 Introduction

Let $G = (V, E)$ be a graph. We say that G is an *intersection graph* of convex sets in $\mathbb{R}^d$ if there is a family $\mathcal{F}$ of convex sets in $\mathbb{R}^d$ and a bijection $V \to \mathcal{F}$ mapping each vertex v_i to a set s_i with the property that the sets s_i and s_j intersect if and only if the corresponding vertices v_i and v_j are connected in G, that is, $\{v_i, v_j\} \in E$. Such graphs are instances of *geometric intersection graphs*, whose study is a core theme of discrete and computational geometry. Historically, intersection graphs have mainly been considered for convex sets in $\mathbb{R}^1$, in which case they are called *interval graphs*, or for convex sets or segments in the plane.

A fundamental computational question for geometric intersection graphs is the *recognition problem* defined as follows: given a graph G, and some (infinite) collection of geometric objects C, decide whether G is an intersection graph of objects of C. While the recognition problem for interval graphs can be solved in linear time [4], the recognition of segment intersection graphs in the plane is significantly harder. In fact, Matoušek and Kratochvíl have shown that this problem is complete for the complexity class $\exists\mathbb{R}$ [11]. Their proof was later simplified by Schaefer [15], see also the streamlined presentation by Matoušek [12].

The complexity class $\exists\mathbb{R}$ was introduced by Schaefer and Štefankovič [16]. It can be thought of as an analogue of NP over the reals. More formally, the class is defined via a canonical problem called ETR, short for *Existential Theory of the Reals*. The problem ETR is a decision problem whose input consists of an integer n and a sentence of the form

$$\exists X_1, \ldots, X_n \in \mathbb{R} : \varphi(X_1, \ldots, X_n),$$

where φ is a quantifier-free formula consisting only of polynomial equations and inequalities connected by logical connectives. The decision problem is to decide whether there exists an assignment of real values to the variables $X_1, \ldots, X_n$ such that the formula φ is true.

It is known that $\mathsf{NP} \subseteq \exists\mathbb{R} \subseteq \mathsf{PSPACE}$, where both inclusions are conjectured to be strict. Many problems in computational geometry have been shown to be $\exists\mathbb{R}$-complete, such as the realizability of abstract order types [14], the art gallery problem [1], the computation of rectilinear crossing numbers [3], geometric embeddings of simplicial complexes [2], and the recognition of several types of geometric intersection graphs [5, 7, 10, 13].

In this work, we extend the recognition problem of intersection graphs of convex sets to the recognition problem of skeletons of nerves of convex sets. Let us introduce the relevant notions. An (abstract) *simplicial complex* on a finite ground set V is a family of subsets of V, called *faces*, that is closed under taking subsets. The *dimension* of a face is the number of its elements minus one. The dimension of a simplicial is the maximum dimension of any of its faces. In particular, a 1-dimensional simplicial complex is just a graph. The *k-skeleton* of a simplicial complex K is the subcomplex of all faces of dimension at most k. Let $\mathcal{F}$ be a family of convex sets in $\mathbb{R}^d$. The *nerve* of $\mathcal{F}$, denoted by $N(\mathcal{F})$ is the simplicial complex with ground set $\mathcal{F}$ where $\{F_1, \ldots, F_m\} \subset \mathcal{F}$ is a face whenever $F_1 \cap \ldots \cap F_m \neq \emptyset$. In other words, the intersection graph of a family $\mathcal{F}$ of convex sets is the 1-skeleton of the nerve $N(\mathcal{F})$. Consider now the following decision problem, which we denote by $R(k, j, d)$: given a simplicial complex K by its maximal faces, decide whether there exists a family $\mathcal{F}$ of j-dimensional convex sets in $\mathbb{R}^d$ such that K is the k-skeleton of $N(\mathcal{F})$.

In some cases, the k-skeleton of a nerve of convex sets uniquely determines the entire nerve: recall *Helly's theorem* [9] which states that for a finite family $\mathcal{F}$ of convex sets, if every $d+1$ of its members have a common intersection, then all sets in $\mathcal{F}$ have a common intersection. Phrased in the language of nerves, this says that if the d-skeleton of the nerve $N(\mathcal{F})$ is complete, then $N(\mathcal{F})$ is an $|\mathcal{F}|$-simplex. In other words, we can retrieve the nerve of a family of convex sets in $\mathbb{R}^d$ from its d-skeleton by filling in higher-dimensional faces whenever all of their d-dimensional faces are present.

▶ Remark. The following Helly-type theorem implies the analogous statement that a nerve of j-dimensional convex sets can be retrieved from its $(j+1)$-skeleton.

▶ Theorem 1. *Let $\mathcal{F}$ be a finite family of j-dimensional convex sets in $\mathbb{R}^d$. Assume that any $j+2$ or fewer members of $\mathcal{F}$ have a common intersection. Then all sets in $\mathcal{F}$ have a common intersection.*

This result is likely known, however we could not find a reference for it, so we include a short proof. The proof requires some algebraic topology, in particular the notion of *homology.* For background on this, we refer to the many textbooks on algebraic topology, for instance the excellent work by Hatcher [8]. For readers not familiar with this concept, the idea of the proof can still be seen by the intuitive notion that $H_k(X) = 0$ means that the space X has no holes of dimension k.

Proof. We want to show that the nerve $N(\mathcal{F})$ is an $|\mathcal{F}|$-simplex. Consider a subfamily $\mathcal{F}' \subseteq \mathcal{F}$ and its induced sub-nerve $N(\mathcal{F}')$. By the nerve theorem (see e.g. [8], Corollary 4G.3), the sub-nerve $N(\mathcal{F}')$ is homotopy-equivalent to the union $\bigcup \mathcal{F}'$ of the sets in $\mathcal{F}'$, implying that the two objects have isomorphic homology groups. As $\bigcup \mathcal{F}'$ has dimension at most j, and $\mathcal{F}$ (and thus also $\mathcal{F}'$) is finite, we have that $H_k(\bigcup \mathcal{F}') = 0$ for all $k \geq j+1$. Thus $H_k(N(\mathcal{F}')) = 0$ for all $k \geq j+1$ and all $\mathcal{F}' \subseteq \mathcal{F}$. On the other hand, the assumption that any $j+2$ or fewer sets have a common intersection implies that the $(j+1)$-skeleton of $N(\mathcal{F})$ is complete and thus $H_k(N(\mathcal{F}')) = 0$ for all $1 \leq k \leq j$ and *all* subfamilies $\mathcal{F}' \subseteq \mathcal{F}$. Thus, $N(\mathcal{F})$ must be a simplex. ◀

2 Containment results

We start by showing that all considered problems are in the complexity class $\exists\mathbb{R}$.

▶ **Theorem 2.** *For all k, j and d, we have $R(k, j, d) \in \exists\mathbb{R}$.*

Proof. Similarly to NP, containment in $\exists\mathbb{R}$ can be proven by providing a certificate consisting of a polynomial number of real values, and a verification algorithm running on the real RAM computation model which verifies these certificates [6]. As a certificate, we use the coordinates of some point in $\mathbb{R}^d$ for each maximal face of the input complex K. These points then describe a family $\mathcal{F}$ of convex sets: Each set F is the convex hull of all points representing maximal faces S of K such that $F \in S$.

Note that if K is the k-skeleton of $N(\mathcal{F})$ for some family $\mathcal{F}$ of j-dimensional convex sets in $\mathbb{R}^d$, such a certificate must exist: The points can be placed in the maximal intersections of $\mathcal{F}$, and shrinking each set to the convex hull of these points cannot change $N(\mathcal{F})$.

Such a certificate can be verified by testing that each set F is j-dimensional (e.g., using linear programming), and by testing that the k-skeleton of $N(\mathcal{F})$ is indeed K. The latter can be achieved in polynomial time by computing the intersection of each subfamily $\mathcal{F}' \subseteq \mathcal{F}$ of at most $\min(k+1, d+1)$ sets. If $k \leq d$, this determines the k-skeleton of $N(\mathcal{F})$. If $k > d$, the k-skeleton of $N(\mathcal{F})$ is determined by the d-skeleton of $N(\mathcal{F})$ by Helly's theorem [9]. ◀

▶ **Lemma 3.** *$R(k, 1, 1)$ is in P for any $k \geq 1$.*

Proof. $R(1, 1, 1)$ is equivalent to recognizing interval graphs, and can thus be solved in polynomial time (see [4]). Since we are considering a family $\mathcal{F}$ of intervals in $\mathbb{R}^1$, the 1-skeleton of $N(\mathcal{F})$ uniquely determines $N(\mathcal{F})$. By Helly's theorem, $N(\mathcal{F})$ must be the clique complex of its 1-skeleton. Thus, $R(k, 1, 1)$ can be solved as follows: Build the graph G given by the 1-skeleton of the input complex K. Test the following four properties: (i) G is an interval graph, (ii) K is at most k-dimensional, (iii) every maximal face of K is a clique of G, and (iv) every clique of size $< k$ in G is contained in some maximal face of K. Return yes if the answer to all these tests is yes, otherwise return no. All tests can be performed in polynomial time, thus $R(k, 1, 1) \in$ P. ◀

For some constellations of k, j, d, any simplicial complex of dimension at most k can be realized as the k-skeleton of the nerve of j-dimensional convex sets in $\mathbb{R}^d$. In this case we say that the problem $R(k, j, d)$ is *trivial.* Evans et al. prove triviality for $R(1, 2, 3)$:

▶ **Lemma 4** ([7]). *$R(1, 2, 3)$ is trivial.*

Furthermore, we can show that if the dimensions j and d get large enough compared to k, the problem also becomes trivial.

▶ **Lemma 5.** *$R(k, 2k+1, 2k+1)$ is trivial.*

Proof. Wegner has shown that every k-dimensional simplicial complex is the nerve of convex sets in $\mathbb{R}^{2k+1}$ [18]. In particular, it is also the k-skeleton of a nerve. ◀

Finally, we prove the following lifting result.

▶ **Lemma 6.** *If $R(k, j, d)$ is trivial, $R(k, j', d')$ is trivial for all $d' \geq d$ and $j \leq j' \leq d'$.*

Proof. We prove that both j and d can be increased by one without destroying triviality, from which the lemma follows.

Any simplicial complex that can be realized in dimension d can also be realized in a d-dimensional subspace of $\mathbb{R}^{d+1}$, thus increasing d by one preserves triviality.

To see that j can be increased, consider a realization of a simplicial complex as the k-skeleton of the nerve of a family $\mathcal{F}$ of j-dimensional convex sets in $\mathbb{R}^d$. Now, consider any two subfamilies $\mathcal{F}_1, \mathcal{F}_2$ of $\mathcal{F}$, such that $\left(\bigcap_{F\in\mathcal{F}_1} F\right) \cap \left(\bigcap_{F\in\mathcal{F}_2} F\right) = \emptyset$. The two intersections $\bigcap_{F\in\mathcal{F}_1} F$ and $\bigcap_{F\in\mathcal{F}_2} F$ must have some distance ϵ. Consider ϵ_{min}, the minimum of all such ϵ over all pairs of subfamilies $\mathcal{F}_1, \mathcal{F}_2$. We extrude every object in $\mathcal{F}$ in some direction not yet spanned by the object by some ϵ' small enough that no intersection $\bigcap_{F\in\mathcal{F}'} F$ for $\mathcal{F}' \subseteq \mathcal{F}$ grows by more than $\epsilon_{min}/3$. This process can not introduce any additional intersections, and thus the nerve of this family of $j+1$-dimensional sets is the same as the nerve of $\mathcal{F}$. We conclude that triviality of $R(k, j, d)$ for $j < d$ implies triviality of $R(k, j+1, d)$. ◀

3 Existing $\exists\mathbb{R}$-Hardness Results

▶ **Lemma 7.** *$R(k, 1, d)$ is $\exists\mathbb{R}$-hard for $k \geq 1$ and $d \geq 2$.*

Proof. For $k = 1$ and $d = 2$, this is equivalent to recognizing segment intersection graphs in the plane, which Schaefer [15] proved to be $\exists\mathbb{R}$-hard by reduction from stretchability. Evans et al. [7] generalize Schaefer's proof for intersection graphs of segments in $\mathbb{R}^3$ ($k = 1$ and $d = 3$). Their proof works by arguing that all segments of their constructed graph must be coplanar. Since the argument implies coplanarity no matter the dimension of the ambient space, the proof also implies $\exists\mathbb{R}$-hardness for $k = 1$ and $d > 3$. Furthermore, for any "yes"-instance of stretchability, the constructed graph can be drawn using segments with no triple intersections. Thus, the proof implies $\exists\mathbb{R}$-hardness for $R(k, 1, d)$ for $k > 1$, as well. ◀

Schaefer [15] furthermore proved that $R(1, 2, 2)$ is $\exists\mathbb{R}$-hard. In the proof of this result, again no triple intersections occur in the representations of "yes"-instances. Thus the same proof applies to the following lemma.

▶ **Lemma 8.** *$R(k, 2, 2)$ is $\exists\mathbb{R}$-hard for any $k \geq 1$.*

This solves the complexity status of $R(1, j, d)$ for all j and d. We summarize these results in the following corollary.

▶ **Corollary 9.** *For $k = 1$, $R(k, j, d)$ is*

- *in* P, *if $j = d = 1$.*
- *$\exists\mathbb{R}$-complete, if $j = 1$ and $d > 2$, or if $j = d = 2$.*
- *trivial in all other cases.*

4 Lifting to Higher Dimensions

We can extend a lifting result due to Tancer [17] to our setting. For this, the *suspension* of a simplicial complex K with ground set V and face family F is the simplicial complex $S(K)$ with ground set $V \cup \{a, b\}$ and faces $F \cup \{f \cup \{a\} \mid f \in F\} \cup \{f \cup \{b\} \mid f \in F\}$.

▶ **Lemma 10.** *Let K be a simplicial complex and let $j \geq d - 1$. Then K is a nerve of j-dimensional convex sets in $\mathbb{R}^d$ if and only if $S(K)$ is a nerve of $(j+1)$-dimensional convex sets in $\mathbb{R}^{d+1}$.*

Proof. We first show that if K is a nerve of convex sets in $\mathbb{R}^d$ then $S(K)$ is a nerve of convex sets in $\mathbb{R}^{d+1}$. For this, let $\mathcal{F}$ be a family of sets in $\mathbb{R}^d$ whose nerve is K and embed them on the hyperplane $x_{d+1} = 0$ in $\mathbb{R}^{d+1}$. For each set $F \in \mathcal{F}$ define F' as the cartesian product of F and the segment defined by $-2 \leq x_{d+1} \leq 2$. Adding the hyperplanes $x_{d+1} = -1$ and $x_{d+1} = 1$, it is easy to see that the nerve of the resulting set family is $S(K)$.

In the other direction, consider a family $\mathcal{F}'$ of $(j+1)$-dimensional convex sets in $\mathbb{R}^{d+1}$ whose nerve is $S(K)$. Let A and B be the convex sets that correspond to the vertices a and

b, respectively. As a and b are not connected in $S(K)$, the sets A and B must be disjoint. In particular, they can be separated by a hyperplane h. For each other set $F' \in \mathcal{F}'$, consider $F := F' \cap h$ and let $\mathcal{F}$ be the family of these intersections. Note that $\mathcal{F}$ is a family of j-dimensional convex sets in $\mathbb{R}^d$. We claim that the nerve of $\mathcal{F}$ is K. Indeed, as K is a subcomplex of $S(K)$, every face of $N(\mathcal{F})$ must be a face of K. On the other hand, for every face f of K, there are points p_a and p_b in A and B, respectively, which lie in the intersection corresponding to faces $f \cup \{a\}$ and $f \cup \{b\}$ of $S(K)$, respectively. The intersection of the segment $p_a p_b$ with h lies in the intersection of the sets corresponding to f, showing that every face of K must be a face of $N(\mathcal{F})$. ◀

Combined with the fact that the d-skeleton determines the entire nerve, we get the following reduction.

▶ **Corollary 11.** *Let $j \in \{d-1, d\}$. If $R(d, j, d)$ is $\exists\mathbb{R}$-hard, then so is $R(d+1, j+1, d+1)$.*

Using the $\exists\mathbb{R}$-hardness of $R(2, 1, 2)$ and $R(2, 2, 2)$ implied by Lemmas 7 and 8, we thus deduce the following

▶ **Theorem 12.** *For any $d \geq 2$ and $k \geq d$, the problems $R(k, d-1, d)$ and $R(k, d, d)$ are $\exists\mathbb{R}$-complete.*

This strengthens a result of Tancer who has shown that $R(d, d, d)$ is NP-hard [17].

5 Conclusion

We have introduced a generalization of the recognition problem of intersection graphs of convex sets and have seen that several existing results in the literature of intersection graphs imply stronger statements in this setting. In particular, the computational complexities of recognizing intersections graphs of convex sets is completely settled. For small k, j, d, the current state of knowledge is summarized in the tables in Figure 1. As can be seen, for many decision problems $R(k, j, d)$, the computational complexity is still open. We conjecture that these cases are either $\exists\mathbb{R}$-complete or trivial, determining which of the two remains an interesting open problem. Of course, the analogous problems can be defined for objects other than convex sets, giving rise to many interesting open problems.

$k = 1$

j \ d	1	2	3	4	5	6	7	8
1	P	$\exists\mathbb{R}$	$\exists\mathbb{R}$	$\exists\mathbb{R}$	$\exists\mathbb{R}$	$\exists\mathbb{R}$	$\exists\mathbb{R}$	$\exists\mathbb{R}$
2		$\exists\mathbb{R}$	T	T	T	T	T	T
3			T	T	T	T	T	T
4				T	T	T	T	T
5					T	T	T	T
6						T	T	T
7							T	T
8								T

$k = 2$

j \ d	1	2	3	4	5	6	7	8
1	P	$\exists\mathbb{R}$	$\exists\mathbb{R}$	$\exists\mathbb{R}$	$\exists\mathbb{R}$	$\exists\mathbb{R}$	$\exists\mathbb{R}$	$\exists\mathbb{R}$
2		$\exists\mathbb{R}$	?	?	?	?	?	?
3			?	?	?	?	?	?
4				?	?	?	?	?
5					T	T	T	T
6						T	T	T
7							T	T
8								T

$k = 3$

j \ d	1	2	3	4	5	6	7	8
1	P	$\exists\mathbb{R}$	$\exists\mathbb{R}$	$\exists\mathbb{R}$	$\exists\mathbb{R}$	$\exists\mathbb{R}$	$\exists\mathbb{R}$	$\exists\mathbb{R}$
2		$\exists\mathbb{R}$	$\exists\mathbb{R}$	?	?	?	?	?
3			$\exists\mathbb{R}$	?	?	?	?	?
4				?	?	?	?	?
5					?	?	?	?
6						?	?	?
7							T	T
8								T

$k = 4$

j \ d	1	2	3	4	5	6	7	8
1	P	$\exists\mathbb{R}$	$\exists\mathbb{R}$	$\exists\mathbb{R}$	$\exists\mathbb{R}$	$\exists\mathbb{R}$	$\exists\mathbb{R}$	$\exists\mathbb{R}$
2		$\exists\mathbb{R}$	$\exists\mathbb{R}$	?	?	?	?	?
3			$\exists\mathbb{R}$	$\exists\mathbb{R}$	?	?	?	?
4				$\exists\mathbb{R}$	?	?	?	?
5					?	?	?	?
6						?	?	?
7							?	?
8								?

Figure 1 The complexity status of $R(k, j, d)$ for $k \leq 4$ and $d, j \leq 8$. P denotes containment in P, $\exists\mathbb{R}$ denotes $\exists\mathbb{R}$-completeness, T denotes triviality, and ? indicates open cases.

References

1 Mikkel Abrahamsen, Anna Adamaszek, and Tillmann Miltzow. The art gallery problem is $\exists\mathbb{R}$-complete. *ACM Journal of the ACM (JACM)*, 69(1):1–70, 2021.

2 Mikkel Abrahamsen, Linda Kleist, and Tillmann Miltzow. Geometric embeddability of complexes is $\exists\mathbb{R}$-complete. *arXiv preprint arXiv:2108.02585*, 2021.

3 Daniel Bienstock. Some provably hard crossing number problems. In *Proceedings of the sixth annual symposium on Computational geometry*, pages 253–260, 1990.

4 Kellogg S Booth and George S Lueker. Testing for the consecutive ones property, interval graphs, and graph planarity using pq-tree algorithms. *Journal of computer and system sciences*, 13(3):335–379, 1976.

5 Jean Cardinal, Stefan Felsner, Tillmann Miltzow, Casey Tompkins, and Birgit Vogtenhuber. Intersection graphs of rays and grounded segments. In *International Workshop on Graph-Theoretic Concepts in Computer Science*, pages 153–166. Springer, 2017.

6 Jeff Erickson, Ivor Van Der Hoog, and Tillmann Miltzow. Smoothing the gap between NP and ER. *SIAM Journal on Computing*, (0):FOCS20–102, 2022.

7 William Evans, Paweł Rzążewski, Noushin Saeedi, Chan-Su Shin, and Alexander Wolff. Representing graphs and hypergraphs by touching polygons in 3d. In *International Symposium on Graph Drawing and Network Visualization*, pages 18–32. Springer, 2019.

8 Allen Hatcher. *Algebraic topology.* Cambridge Univ. Press, Cambridge, 2000.

9 Eduard Helly. Über mengen konvexer körper mit gemeinschaftlichen punkten. *Jahresbericht der Deutschen Mathematiker-Vereinigung*, 32:175–176, 1923.

10 Ross J Kang and Tobias Müller. Sphere and dot product representations of graphs. In *Proceedings of the twenty-seventh annual symposium on Computational geometry*, pages 308–314, 2011.

11 Jan Kratochvíl and Jirí Matoušek. Intersection graphs of segments. *Journal of Combinatorial Theory, Series B*, 62(2):289–315, 1994.

12 Jirí Matoušek. Intersection graphs of segments and $\exists\mathbb{R}$. *arXiv preprint arXiv:1406.2636*, 2014.

13 Colin McDiarmid and Tobias Müller. Integer realizations of disk and segment graphs. *Journal of Combinatorial Theory, Series B*, 103(1):114–143, 2013.

14 Nikolai E Mnëv. The universality theorems on the classification problem of configuration varieties and convex polytopes varieties. In *Topology and geometry—Rohlin seminar*, pages 527–543. Springer, 1988.

15 Marcus Schaefer. Complexity of some geometric and topological problems. In *International Symposium on Graph Drawing*, pages 334–344. Springer, 2009.

16 Marcus Schaefer and Daniel Štefankovič. Fixed points, nash equilibria, and the existential theory of the reals. *Theory of Computing Systems*, 60(2):172–193, 2017.

17 Martin Tancer. d-collapsibility is NP-complete for $d \geq 4$. *Chic. J. Theoret. Comput. Sci*, 3:1–28, 2010.

18 Gerd Wegner. *Eigenschaften der Nerven homologisch-einfacher Familien im $\mathbb{R}^n$.* PhD thesis, Universität Göttingen, 1967.

Spanners under the Hausdorff and Fréchet Distances[*]

Tsuri Farhana[1] and Matthew J. Katz[2]

1 **Department of Computer Science, Ben-Gurion University of the Negev, Beer Sheva, Israel**
tsurif@post.bgu.ac.il

2 **Department of Computer Science, Ben-Gurion University of the Negev, Beer Sheva, Israel**
matya@cs.bgu.ac.il

Abstract

We initiate the study of spanners under the Hausdorff and Fréchet distances. We show that any t-spanner of a planar point-set S is a $\frac{\sqrt{t^2-1}}{2}$-Hausdorff-spanner and a $\min\{\frac{t}{2}, \frac{\sqrt{t^2-t}}{\sqrt{2}}\}$-Fréchet-spanner. We also prove that for any $t > 1$, there exist a set of points S and an ε_1-Hausdorff-spanner of S and an ε_2-Fréchet-spanner of S, where ε_1 and ε_2 are constants, such that neither of them is a t-spanner.

1 Introduction

Let S be a set of points in $\mathbb{R}^d$. The *Euclidean graph* over S, denoted G_S, is the complete graph over S, in which the weight of an edge (u, v) is the Euclidean distance between its endpoints, denoted $d(u, v)$. A subgraph H of G_S is a *t-spanner* of S, for a real number $t \geq 1$, if it is a t-spanner of G_S, that is, if for any pair of points $u, v \in S$, the length of the shortest path between u and v in H is at most $t \cdot d(u, v)$. In general, a path in G_S between u and v whose length is at most $t \cdot d(u, v)$, is called a *t-path*.

Geometric spanners, i.e., t-spanners of G_S, have been studied extensively over the years (see [2]), where the goal is often to construct a t-spanner, for a given $t > 1$, with some desirable properties, such as, small number of edges, small weight, small degree, and small diameter (i.e., the maximum number of edges in a minimum-hop t-path).

In this paper we initiate the study of Hausdorff-spanners and Fréchet-spanners. A path $P(u, v)$ in G_S between u and v is an *ε-Hausdorff-path* (*ε-Fréchet-path*) if $d_H(P(u, v), \overline{uv}) \leq \varepsilon \cdot d(u, v)$ ($d_F(P(u, v), \overline{uv}) \leq \varepsilon \cdot d(u, v)$), where d_H (d_F) denotes the Hausdorff (Fréchet) distance. Thus, a subgraph H of G_S is an *ε-Hausdorff-spanner* (*ε-Fréchet-spanner*) if there exists in H an ε-Hausdorff-path (ε-Fréchet-path) between any two points u and v.

Let S be a set of points in the plane. We show that a t-spanner is an ε_1-Hausdorff-spanner, for $\varepsilon_1 = f(t)$, and an ε_2-Fréchet-spanner, for $\varepsilon_2 = f(t)$. We also prove that for any $t > 1$, there exist an ε_1-Hausdorff-spanner H_1 and an ε_2-Fréchet-spanner H_2, where ε_1 and ε_2 are constants, such that both H_1 and H_2 are *not* a t-spanner.

[*] Supported by Grant 2019715/CCF-20-08551 from the US-Israel Binational Science Foundation/US National Science Foundation, and by the Lynne and William Frankel Center for Computer Science at Ben-Gurion University.

2 Hausdorff spanners

2.1 t-spanners are ε-Hausdorff-spanners

In this section we show that a t-spanner is an ε-Hausdorff-spanner, for $\varepsilon = f(t)$.

Let H be a t-spanner, $t > 1$, and let $u, v \in S$. We assume, with loss of generality, that $d(u, v) = 1$ and that $u = (0, 0)$ and $v = (1, 0)$. Let $P(u, v)$ be a t-path in H between u and v. Then, $P(u, v)$'s length is at most t.

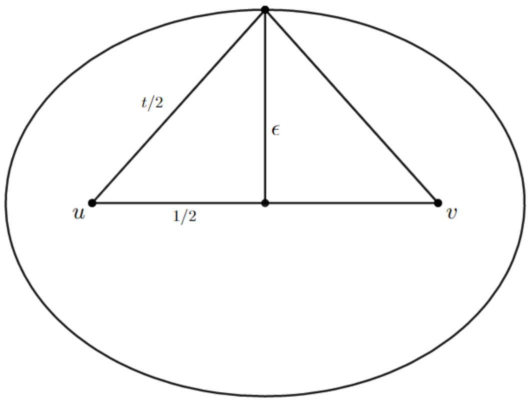


Figure 1 The furthest possible distance from a t-path to the segment between its endpoints.

▶ **Observation 2.1.** *Let E be the ellipse with foci points at u and v, for which $d(p, u) + d(p, v) = t \cdot d(u, v) = t$, for every point p on its boundary. Then, $P(u, v) \subset E$. Moreover, the points of E that are furthest from $\overline{uv}$ are the boundary points above and below the midpoint of $\overline{uv}$; their distance to $\overline{uv}$ is $\frac{\sqrt{t^2-1}}{2}$ (see Figure 1).*

▶ **Lemma 2.2.** *$P(u, v)$ is a $\frac{\sqrt{t^2-1}}{2}$-Hausdorff-path.*

Proof. Since $d(u, v) = 1$, we need to show that $d_H(P(u, v), \overline{uv}) \leq \frac{\sqrt{t^2-1}}{2}$. On the one hand, by Observation 2.1, the distance from any point p on $P(u, v)$ to $\overline{uv}$ is at most $\frac{\sqrt{t^2-1}}{2}$. On the other hand, let $q = (q_x, 0)$ be a point on $\overline{uv}$. Then, since $P(u, v)$ is a path between u and v, there exists a point p on $P(u, v)$ with x-coordinate q_x. Now, by Observation 2.1, the distance from q to p is at most $\frac{\sqrt{t^2-1}}{2}$. We conclude that $d_H(P(u, v), \overline{uv}) \leq \frac{\sqrt{t^2-1}}{2}$. ◀

We have shown that any t-path in H is a $\frac{\sqrt{t^2-1}}{2}$-Hausdorff-path, and therefore H is a $\frac{\sqrt{t^2-1}}{2}$-Hausdorff-spanner. We conclude that

▶ **Corollary 2.3.** *Any t-spanner is an ε-Hausdorff-spanner, for $\varepsilon = \frac{\sqrt{t^2-1}}{2}$.*

2.2 ε-Hausdorff-spanners are not necessarily t-spanners

In this section we show that not every Hausdorff spanner is a Euclidean spanner. More precisely, we present an infinite sequence of graphs, such that all of them are c-Hausdorff-spanners, for some fixed constant $c > 0$, but for any $t \geq 1$, there exists a graph in the sequence that is not a t-spanner. Formally, we prove the following theorem.

▶ **Theorem 2.4.** *There exists a constant $c > 0$, such that for any $t \geq 1$, one can construct a graph that is a c-Hausdorff-spanner and is not a t-spanner.*

To prove the theorem, we define a sequence of graphs $F_0, F_1, F_2 \dots$ as follows: Let F_0 be the unit line segment with endpoints $(0,0)$ and $(1,0)$, that is, the endpoints are the vertices of F_0 and the segment is its single edge. Now, for any $n > 0$, we construct F_n from F_{n-1}, by considering each segment (i.e., edge) s of F_{n-1} and (i) partitioning s into three subsegments of equal length by adding two new vertices, (ii) forming an equilateral triangle with the middle subsegment as its base by adding a vertex on the outer side of the middle subsegment and connecting it to the endpoints of the middle subsegmet, and (iii) removing the middle subsegment. That is, the edge s is replaced by four edges obtained by adding three new vertices; see Figure 2.

The curve that is obtained by applying this construction indefinitely is the fractal known as the Koch curve [1]; it is one of the three curves forming the Koch snowflake. It is well known that the length of the Koch curve is unbounded, that is, for every $l > 0$, there exists an integer n, such that the length of F_n, i.e., the sum of its edge lengths, is greater than l.

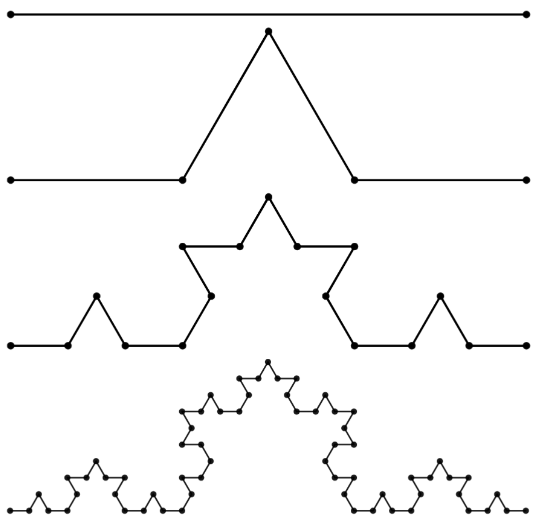


Figure 2 The graphs F_0, F_1, F_2 and F_3.

Since the length of the path between the extreme vertices of F_n, i.e., between its vertices at $(0,0)$ and $(1,0)$, can be made arbitrarily long, we conclude that for any $t \geq 1$, there exists an integer n, such that F_n is not a t-spanner.

We next show that there exists a constant $c > 0$, such that for any $n \geq 0$, the graph F_n is a c-Hausdorff-spanner. We actually show that 6 is such a constant.

Notation and definitions.

Let $F_n = (V, E)$. The *level* of a vertex $v \in V$, denoted $l(v)$, is the smallest index $0 \leq i \leq n$, such that v is already a vertex in F_i. If $l(v) \leq i-1$, we write $l(v) = i^-$.

For a pair of vertices $u, v \in V$, we denote the path between u and v by $P(u,v) = (u, v_1, v_2, \dots, v)$ and its corresponding sequence of levels by $P_l(u,v)$, that is, $P_l(u,v) = (l(u), l(v_1), l(v_2), \dots, l(v))$. The *level* of u, v, denoted $l(u,v)$, is now the smallest level i such that there are at least two elements in the sequence $P_l(u,v)$ that are smaller or equal to i. For example, if $P_l(u,v) = (1,3,3,3,2,3,3,3)$, then $l(u,v) = 2$.

The path from the leftmost vertex to the rightmost vertex induces a natural order on the vertices of F_n. We say that vertex u of F_n precedes/succeeds vertex v of F_n if u appears before/after v in this path.

Next, we define the *bounding rectangle* of three consecutive vertices $v_1, v_2, v_3 \in V$; see Figure 3. If the angle between $\overline{v_1v_2}$ and $\overline{v_2v_3}$ is 240°, then the bounding rectangle of v_1, v_2, v_3 is the rectangle such that (i) v_1v_3 is one of its diagonals and (ii) if $l(v_1) < l(v_3)$, the edge $\overline{v_1v_2}$ is contained in one of its long edges. Otherwise, the edge $\overline{v_2v_3}$ is contained in one of its long edges. If the angle between $\overline{v_1v_2}$ and $\overline{v_2v_3}$ is 60°, then the bounding rectangle of v_1, v_2, v_3 is the rectangle such that (i) one of its edges is $\overline{v_1v_3}$, and v_2 is on the opposite edge. Notice that the bounding rectangle is not always parallel to the axes, rather it is parallel to either $\overline{v_1v_2}$ or $\overline{v_2v_3}$, in the former case, or to the line segment $\overline{v_1v_3}$, in the latter case.

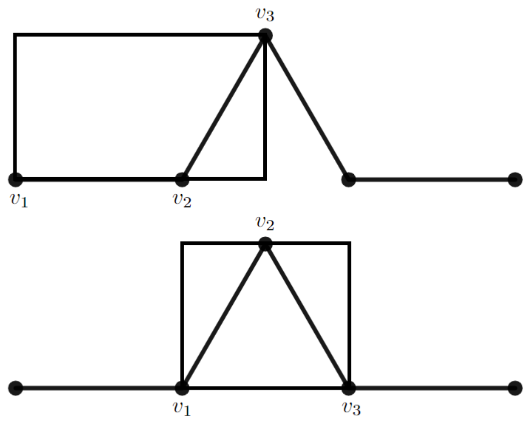


Figure 3 Top: The bounding rectangle when the angle between $\overline{v_1v_2}$ and $\overline{v_2v_3}$ is 240°. Bottom: The bounding rectangle when the angle between $\overline{v_1v_2}$ and $\overline{v_2v_3}$ is 60°

.

▶ **Observation 2.5.** *In F_n, for any two adjacent vertices, at least one of them is of level n.*

▶ **Observation 2.6.** *In the graph F_n and for $1 \le i \le n$, let v_1, v_2 be two vertices that are adjacent to each other in the graph F_{i-1}. Then there are exactly three vertices of level i between v_1 and v_2.*

▶ **Observation 2.7.** *In the graph F_n and for $1 \le i \le n$, let v_1, v_2, v_3 be three vertices that are consecutive in the graph F_i and let R be their bounding rectangle (with respect to F_i). Then $P(v_1, v_3) \subseteq R$, that is, the path between v_1 and v_3 in F_n is contained in R.*

▶ **Observation 2.8.** *Consider the rectangle R from Observation 2.7. Then, the length of its diagonal is at most*

$$\sqrt{\left(\frac{3}{2} \cdot \frac{1}{3^i}\right)^2 + \left(\frac{\sqrt{3}}{2} \cdot \frac{1}{3^i}\right)^2} = \frac{\sqrt{3}}{3^i},$$

which is the length of its diagonal assuming it is the longer of the two possible rectangles, see Figure 3 (top).

We now bound (from above) the Hausdorff distance (denoted d_H) between a path $P(u, v)$ in F_n and the line segment $\overline{uv}$ as a function of $l(u, v)$. The proof of the following lemma can be found in the full version of this paper.

▶ **Lemma 2.9.** *Consider the graph $F_n = (V, E)$ and let $u, v \in V$ such that $l(u, v) = i$, then $d_H(P(u, v), \overline{uv}) \le \frac{\sqrt{3}}{3^{i-1}}$.*

Next, we bound $d(u, v)$ for vertices u and v of F_n (from below) as a function of $l(u, v)$.

▶ **Lemma 2.10.** *Consider the graph $F_n = (V, E)$ and let $u, v \in V$ such that $l(u, v) = i$, then $d(u, v) \geq \frac{1}{3^i} \cdot \frac{\sqrt{3}}{2}$.*

Proof. Let u' (v') be the level-i^- vertex of F_n that precedes u (succeeds v). Since $l(u, v) = i$, there is at most one vertex in $P(u, v)$ of level i^-. We distinguish between three cases. A full description of Cases 2 and 3 can be found in the full version of this paper.

Case 1: There is no vertex of level i^- in $P(u, v)$. By Observation 2.6, there are exactly three vertices of level i between u' and v'. Moreover, at least two of them are in $P(u, v)$ (since $l(u, v) = i$ and there is no vertex of level i^- in $P(u, v)$). Assume, without loss of generality, that the middle and the right of these level-i vertices are in $P(u, v)$.

We draw two parallel lines as depicted in Figure 4a. The first line passes through the left and middle level-i vertices, and the second line passes through the right level-i vertex and is parallel to the first line. Next, we observe that u lies on one side of these lines and v lies on the other side, and therefore the distance between u and v is at least the height of the level-i triangle, formed by the three level-i vertices. It is easy to verify that this height is $\frac{1}{3^i} \cdot \frac{\sqrt{3}}{2}$. We conclude that $d(u, v) \geq \frac{1}{3^i} \cdot \frac{\sqrt{3}}{2}$.

Case 2: There is a 60°-vertex w of level i^- in $P(u, v)$. As in Case 1, we draw two parallel lines (see Figure 4b) that separate between u and v, and conclude that $d(u, v) \geq \frac{1}{3^i} \cdot \frac{\sqrt{3}}{2}$.

Case 3: There is a 240°-vertex w of level i^- in $P(u, v)$. As in Case 1, we draw two parallel lines (see Figure 4c) that separate between u and v, and conclude that $d(u, v) \geq \frac{1}{3^i} \geq \frac{1}{3^i} \cdot \frac{\sqrt{3}}{2}$. ◀

We are now ready to prove Theorem 2.4. We first show that for any $n \geq 0$, the graph F_n is a 6-Hausdorff-spanner. Let u, v be two vertices of F_n and set $i = l(u, v)$. Then, on the one hand by Lemma 2.9, $d_H(P(u, v), \overline{uv}) \leq \frac{\sqrt{3}}{3^{i-1}}$, and on the other hand by Lemma 2.10, $d(u, v) \geq \frac{1}{3^i} \cdot \frac{\sqrt{3}}{2}$. Therefore,

$$\frac{d_H(P(u, v), \overline{uv})}{d(u, v)} \leq \frac{\frac{\sqrt{3}}{3^{i-1}}}{\frac{1}{3^i} \cdot \frac{\sqrt{3}}{2}} = 6 \,.$$

Next, let $t \geq 1$. Then, there exists an integer $n > 1$, such that the length of the path P between the extreme vertices of F_n (i.e., the vertices at $(0, 0)$ and $(1, 0)$) is greater than t, and therefore F_n is not a t-spanner (since the length of P over the length of the segment between the extreme vertices of F_n is simply the length of P).

3 Fréchet spanners

3.1 t-spanners are ε-Fréchet-spanners

In this section we show that a t-spanner is an ε-Fréchet-spanner, for $\varepsilon = f(t)$.

Let H be a t-spanner, $t > 1$, and let $u, v \in S$. We assume, with loss of generality, that $d(u, v) = 1$ and that $u = (0, 0)$ and $v = (1, 0)$. Let $P(u, v)$ be a t-path in H between u and v. Then, $P(u, v)$'s length is at most t.

We first prove that $P(u, v)$ is an ε-Fréchet-path, for $\varepsilon = \frac{t}{2}$. This bound is useful when t is 'large', but, since it is never smaller than 1/2, it is less useful when t approaches 1, in which case we would like to show (if possible) that ε approaches 0. To address this issue, we prove a second bound on ε, which is better when $1 < t < 2$. Specially, we prove that $P(u, v)$ is also an ε-Fréchet-path, for $\varepsilon = \frac{\sqrt{t^2 - t}}{\sqrt{2}}$.

3.1.1 Bound 1 — The bound for large t

▶ **Lemma 3.1.** *$P(u,v)$ is a $\frac{t}{2}$-Fréchet-path.*

Proof. Let p be the middle point of $P(u,v)$, that is, the distance from u to p (through $P(u,v)$) is equal to the distance from p to v (through $P(u,v)$). Consider a dog and its owner, both walking from u to v, where the dog is walking along $P(u,v)$ and the owner is walking along $\overline{uv}$. Their hike consists of three stages. In the first, the owner is at u and the dog advances to p; in the second, the owner advances from u to v, while the dog stays at p; and in the third, the owner is at v and the dog advances from p to v.

We show that at any point along their hike, the distance between the dog and its owner does not exceed $t/2$. Indeed, let p^- be a point on $P(u,v)$, anywhere between u and p. Then, the distance from u to p^- (through $P(u,v)$) is at most $t/2$, and therefore $d(p^-,u) \le t/2$. Similarly, let p^+ be a point on $P(u,v)$, anywhere between p and v. Then, the distance from p^+ to v (through $P(u,v)$) is at most $t/2$, and therefore $d(p^+,v) \le t/2$. Finally, let q be a point on $\overline{uv}$. We need to show that $d(p,q) \le t/2$. Indeed, if $q_x \le p_x$, then $d(p,q) \le d(p,u) \le t/2$, and if $q_x \ge p_x$, then $d(p,q) \le d(p,v) \le t/2$. ◀

▶ **Corollary 3.2.** *Any t-spanner is an ε-Fréchet-spanner, for $\varepsilon = \frac{t}{2}$.*

3.1.2 Bound 2 — The bound for small t

Recalling Observation 2.1, we observe that

▶ Observation 3.3. Let p be any point on $P(u,v)$, then (i) $-\frac{t-1}{2} \le p_x \le \frac{t-1}{2}$ and (ii) $-\frac{\sqrt{t^2-1}}{2} \le p_y \le \frac{\sqrt{t^2-1}}{2}$.

▶ **Lemma 3.4.** *P(u,v) is a $\frac{\sqrt{t^2-t}}{\sqrt{2}}$-Fréchet-path.*

Proof. Consider a dog and its owner, both walking from u to v, where the dog is walking along $P(u,v)$ and the owner is walking along $\overline{uv}$. We denote the location of the dog at time $0 \le \tau \le 1$ by $P(\tau)$, where $P(0) = u$, $P(1) = v$, and, for any $\tau_1 < \tau_2$, the point $P(\tau_2)$ does not precede the point $P(\tau_1)$ (on $P(u,v)$). Moreover, we denote the x-coordinate of the rightmost point visited by the dog by time τ by x_τ, that is, $x_\tau = \max\{P(\tau')_x \mid 0 \le \tau' \le \tau\}$.

The location of the person is determined by the location of the dog. More precisely, at time τ the person is at $(\max\{x_\tau - \frac{t-1}{2}, 0\}, 0)$. Finally, if at time $\tau = 1$ the person is not yet at v, then she advances directly to v. Clearly, the person never moves backwards, since the function x_τ is non-decreasing.

We now prove that the distance between the dog and its owner never exceeds $\frac{\sqrt{t^2-t}}{\sqrt{2}}$. Let $0 \le \tau \le 1$. We distinguish between three cases.

Case 1: $\max\{x_\tau - \frac{t-1}{2}, 0\} \le P(\tau)_x$.

$$\begin{aligned} d(P(\tau),&(\max\{x_\tau - \frac{t-1}{2},0\},0)) \le \sqrt{(P(\tau)_x - (x_\tau - \frac{t-1}{2}))^2 + P(\tau)_y^2} \\ &\le \sqrt{(x_\tau - (x_\tau - \frac{t-1}{2}))^2 + P(\tau)_y^2} \le \sqrt{\left(\frac{t-1}{2}\right)^2 + \left(\frac{\sqrt{t^2-1}}{2}\right)^2} \\ &= \sqrt{\frac{t^2-2t+1}{4} + \frac{t^2-1}{4}} = \sqrt{\frac{2t^2-2t}{4}} = \frac{\sqrt{t^2-t}}{\sqrt{2}}, \end{aligned}$$

where the first inequality is true, since, if $\max\{x_\tau - \frac{t-1}{2}, 0\} \ne x_\tau - \frac{t-1}{2}$, then $x_\tau - \frac{t-1}{2} < 0$ and both $P(\tau)_x$ and $(P(\tau)_x - (x_\tau - \frac{t-1}{2}))$ are non-negative.

Case 2: $x_\tau - \frac{t-1}{2} > 0$ and $P(\tau)_x < x_\tau - \frac{t-1}{2}$. We first observe that $P(\tau)_x \geq x_\tau - (t-1)$, since $t \geq x_\tau + (x_\tau - P(\tau)_x) + |1 - P(\tau)_x| \geq x_\tau + (1 - P(\tau)_x)$. Now,

$$\begin{aligned} d(P(\tau),&(\max\{x_\tau - \frac{t-1}{2}, 0\}, 0)) = d(P(\tau), (x_\tau - \frac{t-1}{2}, 0)) \\ &= \sqrt{(P(\tau)_x - (x_\tau - \frac{t-1}{2}))^2 + P(\tau)_y^2} = \sqrt{((x_\tau - \frac{t-1}{2}) - P(\tau)_x)^2 + P(\tau)_y^2} \\ &\leq \sqrt{((x_\tau - \frac{t-1}{2}) - (x_\tau - (t-1)))^2 + P(\tau)_y^2} \\ &\leq \sqrt{\left(\frac{t-1}{2}\right)^2 + \left(\frac{\sqrt{t^2-1}}{2}\right)^2} = \frac{\sqrt{t^2-t}}{\sqrt{2}}. \end{aligned}$$

Case 3: $x_\tau - \frac{t-1}{2} \leq 0$ and $P(\tau)_x < 0$. By Observations 3.3, we have $-\frac{t-1}{2} \leq P(\tau)_x < 0$.

$$\begin{aligned} d(P(\tau),&(\max\{x_\tau - \frac{t-1}{2}, 0\}, 0)) = d(P(\tau), (0,0)) = \sqrt{P(\tau)_x^2 + P(\tau)_y^2} \\ &\leq \sqrt{\left(-\frac{t-1}{2}\right)^2 + \left(\frac{\sqrt{t^2-1}}{2}\right)^2} = \frac{\sqrt{t^2-t}}{\sqrt{2}}. \end{aligned}$$

We have shown that in all cases the distance between the dog and its owner is at most $\frac{\sqrt{t^2-t}}{\sqrt{2}}$. Moreover, if the dog reaches v first, then during the last part of the person's hike, this distance only decreases. We thus conclude that P(u,v) is a $\frac{\sqrt{t^2-t}}{\sqrt{2}}$-Fréchet-path. ◀

▶ **Corollary 3.5.** *Any t-spanner is an ε-Fréchet-spanner, for $\varepsilon = \frac{\sqrt{t^2-t}}{\sqrt{2}}$.*

3.2 ε-Fréchet-spanners are not necessarily t-spanners

Consider the graph F_n, defined in Section 2.2. One can prove that F_n is a c-Fréchet-spanner, for $c \leq 6$, in essentially the same way as we proved that it is a c-Hausdorff-spanner, for $c \leq 6$. More precisely, referring to Lemma 2.9, since both $P(u,v)$ and $\overline{uv}$ are contained in the rectangle R, the Fréchet distance between them is at most the length of R's diagonal, that is, $d_F(P(u,v), \overline{uv}) \leq \frac{\sqrt{3}}{3^{i-1}}$. We thus conclude that

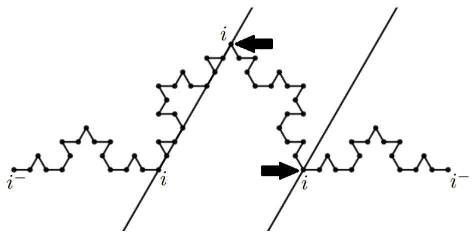
▶ **Theorem 3.6.** *There exists a constant $c > 0$, such that for any $t > 1$, one can construct a graph that is a c-Fréchet-spanner and is not a t-spanner.*

Acknowledgment. We thank Boris Aronov for helpful discussions on the connection between t-spanners and ε-Hausdorff-spanners.

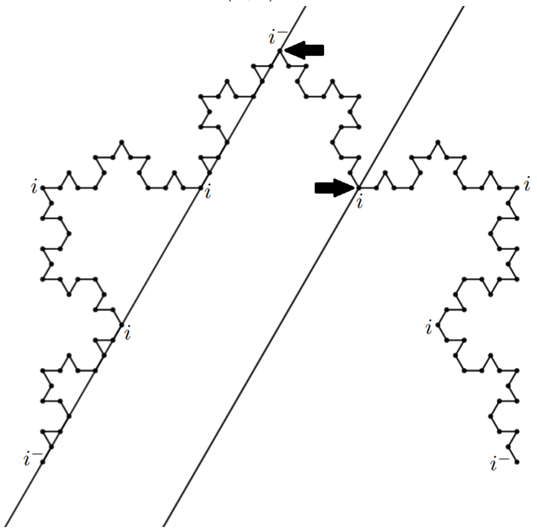
References

1 Helge von Koch. Sur une courbe continue sans tangente, obtenue par une construction géométrique élémentaire. *Arkiv för matematik, astronomi och fysik* (in French). 1:681–704, 1904.

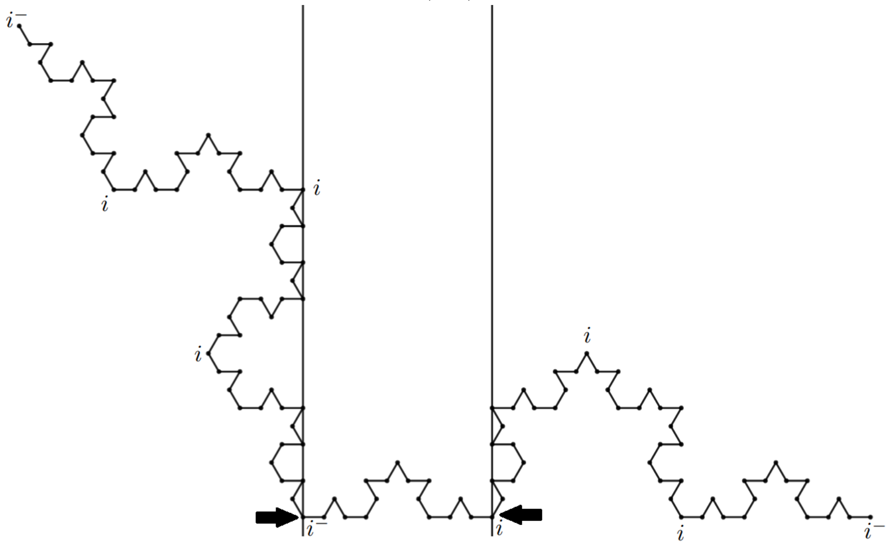
2 Giri Narasimhan and Michiel Smid. *Geometric Spanner Networks*. Cambridge University Press, 2009.



(a) Case 1: There is no vertex of level i^- in $P(u,v)$.



(b) Case 2: There is a 60°-vertex of level i^- in $P(u,v)$.



(c) Case 3: There is a 240°-vertex of level i^- in $P(u,v)$.

Figure 4 Proof of Lemma 2.10. The arrows mark the vertices that are known to be in $P(u,v)$.

Flip Graphs for Arrangements of Pseudocircles[*]

Stefan Felsner[1], Johannes Obenaus[2], Sandro Roch[3], Manfred Scheucher[4], and Birgit Vogtenhuber[5]

1 **Institut für Mathematik, Technische Universität Berlin, Germany**
`felsner@math.tu-berlin.de`

2 **Institut für Informatik, Freie Universität Berlin, Germany**
`johannes.obenaus@fu-berlin.de`

3 **Institut für Mathematik, Technische Universität Berlin, Germany**
`roch@math.tu-berlin.de`

4 **Institut für Mathematik, Technische Universität Berlin, Germany**
`scheucher@math.tu-berlin.de`

5 **Institute of Software Technology, Graz University of Technology, Austria**
`bvogt@ist.tugraz.at`

Abstract

An arrangement of pseudocircles is a finite collection of simple closed curves in the plane such that every pair of curves is either disjoint or intersects in two crossing points. We study flip graphs of families of pseudocircle arrangements. We prove that triangle flips induce a connected flip graph (i) on *intersecting* arrangements and (ii) on *cylindrical intersecting* arrangements. Our constructions make essential use of variants of the sweeping lemma for pseudocircle arrangements due to Snoeyink and Hershberger (Proc. SoCG 1989: 354–363). We also study cylindrical arrangements in their own right and provide new combinatorial characterizations of this class of pseudocircle arrangements.

1 Introduction

Reconfiguration is a widely studied topic in discrete mathematics and theoretical computer science [9]. In many cases, reconfiguration problems can be stated in terms of a *flip graph*. For a class of objects, the flip graph has a vertex for each object and adjacencies are determined by a local flip operation, which transforms one object into another. Typically, the first question is whether a flip graph is connected. In the affirmative case, more refined questions regarding diameter, the degree of connectivity, or Hamiltonicity can be of interest. Hamiltonicity of flip graphs is related to Gray codes, cf. [8]. For further details on flip-graphs in general we also refer the reader to the survey [2].

Ringel [10] showed flip-connectivity for arrangements of *pseudolines* under triangle flips. There, an arrangement of pesudolines is a set of bi-infinite curves that pairwise intersect exactly once. A triangle flip then corresponds to moving a pseudoline incident to a triangular cell over the crossing of the two other pseudolines. If pseudolines are also allowed to be disjoint, flipping triangles is not enough, as one also needs to allow two pseudolines to become intersecting or non-intersecting. Snoeyink and Hershberger [11] showed flip-connectivity for such arrangements of pseudolines with these three operations.

[*] This work was initiated at a workshop of the collaborative DACH project *Arrangements and Drawings* in Lutherstadt Wittenberg in April 2022. We thank all the participants for the inspiring atmosphere. S.F. was supported by DFG Grant FE 340/13-1. J.O. was supported by ERC StG 757609. S.R. was supported by DFG-Research Training Group 'Facets of Complexity' (DFG-GRK 2434). M.S. was supported by DFG Grant SCHE 2214/1-1.

Similarly, in the context of arrangements of pseudocircles, which have been first studied by Grünbaum [7], flipping triangles is not enough if disjoint pseudocircles are allowed. In such cases, the set of flips is extended by *digon-create* to allow two initially disjoint pseudocircles to start intersecting in a digon and the reverse operation, called *digon-collapse* (see Section 2 for definitions).

With all the three flips, the flip-connectivity of arrangements of proper circles is evident since one can shrink all the circles until they have pairwise disjoint interiors. Essentially the same idea works for arrangements of pseudocircles. In this case, however, the fact that a pseudocircle can be shrunk is based on the sweeping lemma of Snoeyink and Hershberger [11]. Allowing only triangle flips, Felsner and Scheucher [5] showed flip-connectivity for classes of arrangements of proper circles and conjectured that the results persist for pseudocircles:

▶ **Conjecture 1** ([5, Conjecture 8.6]). *For every $n \in \mathbb{N}$:*

(1) The flip graph of intersecting arrangements of n pseudocircles is connected.
(2) The flip graph of digon-free intersecting arrangements of n pseudocircles is connected.

As the main result of this article we prove part (1) of Conjecture 1.

▶ **Theorem 1.1.** *The flip graph of arrangements of n pairwise intersecting pseudocircles is connected.*

For our proof of Theorem 1.1, we use *cylindrical arrangements*. These are arrangements of pseudocircles in the plane such that the bounded interiors of all the pseudocircles have a common intersection, which we call the *center*. We first show that every cylindrical intersecting arrangement can be flipped into a canonical arrangement by only using triangle flips and without leaving the class of cylindrical arrangements.

▶ **Theorem 1.2.** *The flip graph of cylindrical arrangements of n pairwise intersecting pseudocircles is connected.*

Showing that every intersecting arrangement $\mathcal{A}$ can be flipped into some cylindrical arrangement then completes the proof of Theorem 1.1. We further study the diameter of flip graphs. In each of the two considered settings (cylindrical / general arrangements of pairwise intersecting pseudocircles), we obtain asymptotically tight bounds for the diameter.

▶ **Proposition 1.3.** *The flip graph of cylindrical arrangements of n pairwise intersecting pseudocircles has diameter at least $2\binom{n}{3}$ and at most $4\binom{n}{3}$.*

▶ **Proposition 1.4.** *The flip graph of arrangements of n pairwise intersecting pseudocircles has diameter $\Theta(n^3)$.*

Last but not least, we present the following equivalent characterizations of cylindrical arrangements of pseudocircles (cf. the full version of this paper). Item (2) uses a special arrangement of three pseudocircles that we call NonKrupp(3) (see Section 2), for items (3-4), we orient pseudocircles counterclockwise, which induces an orientation on the edges of the arrangement, and the eccentricity in item (5) refers to the dual graph of the arrangement: The *eccentricity* of a face in an arrangement of pseudocircles is the maximum distance to any other face, where the *distance* between two faces z, z' is the minimum number of pseudocircles that a curve starting in the interior of z and ending in the interior of z' must cross.

▶ **Proposition 1.5.** *Let $\mathcal{A}$ be an arrangement of n pseudocircles with pairwise overlapping interiors. Then, the following five statements are equivalent:*

(1) $\mathcal{A}$ is cylindrical.
(2) $\mathcal{A}$ does not contain a NonKrupp(3) as a subarrangement.
(3) There is no clockwise oriented cycle in $\mathcal{A}$.
(4) There is no clockwise oriented face in $\mathcal{A}$.
(5) The unbounded face has eccentricity n.

2 Preliminaries

A *pseudocircle* is a simple closed curve C which partitions the plane into a bounded region, the interior $\text{int}(C)$, and an unbounded region, the exterior $\text{ext}(C)$. An *arrangement of pseudocircles* is a finite collection of pseudocircles such that every two pseudocircles either are disjoint or they intersect in two points, where the curves cross properly. Furthermore, no three pseudocircles intersect in a common point. An arrangement partitions the plane into *vertices* (the intersection points), *edges* (maximal contiguous vertex-free pieces of pseudocircles), and *faces* (connected components of the plane after removing all pseudocircles).

A face with k edges along its boundary is a *k-face*, a 2-face is a *digon* (some authors call it *empty lens*), and a 3-face is a *triangle*. It is an instructive exercise to verify that there are exactly four arrangements of three pairwise intersecting pseudocircles (shown in Figure 1).

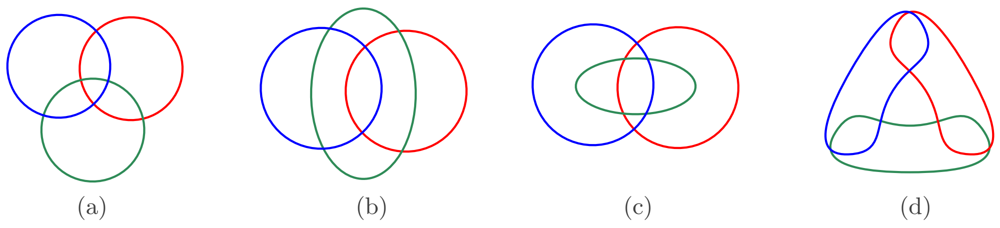


Figure 1 The four non-isomorphic arrangements of 3 pairwise intersecting pseudocircles in the plane. (a) shows the Krupp and (b)–(d) show the three types of NonKrupp arrangements.

Following [5], we call the arrangement, depicted in Figure 1(a), with 8 triangles the *Krupp* arrangement and the other ones *NonKrupp*. To make them distinguishable we write NonKrupp(k) to denote the NonKrupp arrangement whose unbounded face has complexity k, e.g., NonKrupp(2) is the arrangement shown in Figure 1(c). Note that among the arrangements of Figure 1, the arrangement (d), i.e., the NonKrupp(3), is the only non-cylindrical.

2.1 Sweeps and Flips

Snoeyink and Hershberger [11] studied continuous transformations of curves and, in particular, of pseudocircles. More precisely, they define the *sweep* of a pseudocircle as a continuous process to expand or shrink the pseudocircle. However, crucially, they also argue that this continuous process can be viewed as a discrete process as the combinatorics of the arrangement changes with one of the following operations, called *flips*:

- a pseudocircle moves over the crossing of two others (*triangle flip*),
- a pseudocircle gains two intersections with a pseudocircle (*digon-create*), or
- a pseudocircle loses its two intersections with a pseudocircle (*digon-collapse*).

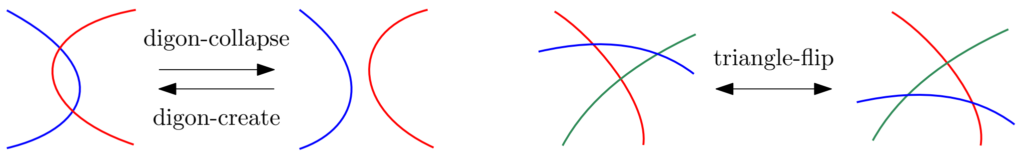


Figure 2 An illustration of the three flip operations.

Figure 2 depicts the three flip operations. Whenever we speak of *flips* without further specification we refer to these three flips, the term *digon flip* refers to the two flips involving a digon, and otherwise we use the precise term when referring to a specific type of flip.

Snoeyink and Hershberger prove a sweeping lemma (cf. [11, Lemma 3.2]) for families of simple curves that pairwise intersect at most twice and are either bi-infinite or closed (i.e., pseudocircles). The flip operations for bi-infinite curves are defined analogously.

▶ **Lemma 2.1** (Sweeping Lemma [11]). *Let $\mathcal{A}$ be an arrangement of pseudocircles or an arrangement of bi-infinite curves that pairwise intersect at most twice. Then $\mathcal{A}$ can be swept starting from any curve C in A by using three operations: triangle flips, digon-create, and digon-collapse.*

Note that in the context of pseudocircles we sweep towards the *inside* or *outside*, whereas in the context of bi-infinite curves we sweep *upwards* or *downwards*. It is straight-forward to verify that Lemma 2.1 implies the flip-connectivity for arrangements of pseudocircles.

It will be convenient to have a separate sweeping lemma for lenses. A *lens* in an arrangement is a maximally bounded region in a subarrangement formed by two intersecting pseudocircles. An *arc* is a contiguous subset of a pseudocircle, starting and ending at a vertex of the arrangement.

Let $\mathcal{A}$ be an arrangement of pseudocircles and let Q be (the closure of) a lens bounded by two pseudocircles C_L and C_R. We denote by $L = C_L \cap Q$ and $R = C_R \cap Q$ the two boundary arcs on Q belonging to C_L and C_R, respectively. An *arc of Q* is a maximal connected piece of the intersection of a pseudocircle C with Q, where $C \notin \{C_L, C_R\}$. In other words, an arc of Q is always a contiguous subset of C which has both endpoints on the boundary of Q and whose relative interior lies completely in the interior of Q. If an arc a of Q has both endpoints on L or both endpoints on R, then a forms a lens with L or R, respectively. Otherwise the arc has one endpoint on L and one on R, in this case we call the arc *transversal* (see Figure 3).

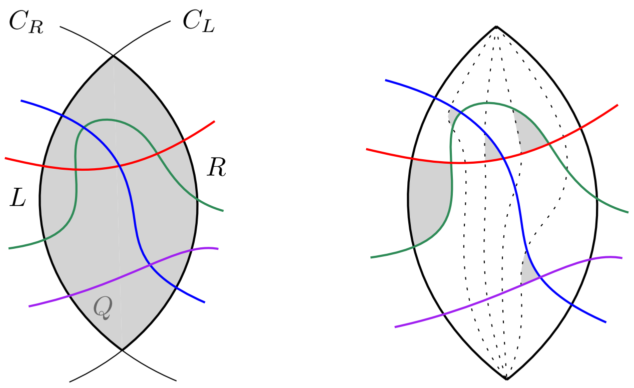


Figure 3 Illustration of Lemma 2.2: a lens Q with four transversal arcs and a sweep of Q.

▶ **Lemma 2.2.** *If Q is a lens and all the arcs of Q are transversal, then, using only triangle flips, L can be swept towards R until the interior of Q does not contain a vertex of the arrangement anymore.*

2.2 Cylindrical Arrangements

An *arrangement of pseudoparabolas* is a finite collection of x-monotone curves defined over a common interval such that every two curves are either disjoint or intersect in two points where the curves cross. The following result gives a reversible mapping from cylindrical arrangements to arrangements of pseudoparabolas. The result has been announced by Bultena et al. [3, Lemma 1.3] and a full proof has been given by Agarwal et al. [1, Lemma 2.11].

▶ **Proposition 2.3** ([3, 1]). *A cylindrical arrangement of pseudocircles $\mathcal{C}$ can be mapped to an arrangement of pseudoparabolas $\mathcal{A}$ in an axis-aligned rectangle B such that $\mathcal{C}$ is isomorphic to the arrangement obtained by identifying the two vertical sides of B and mapping the resulting cylindrical surface homeomorphically to a ring in the plane.*

3 Flip Graphs on Pseudocircle Arrangements

We here sketch the flip-connectivity (Theorem 1.2 and Theorem 1.1).

3.1 Proof Sketch of Theorem 1.2: Connectivity Cylindrical

We prove the flip-connectivity for cylindrical arrangements by showing that any given arrangement can be flipped to a *canonical* arrangement which is depicted in Figure 4.

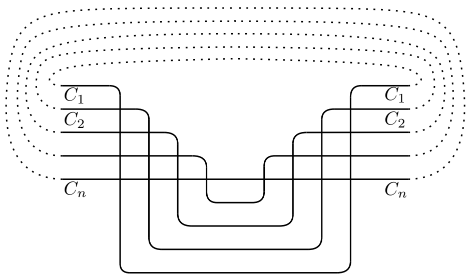


Figure 4 The canonical arrangement for cylindrical arrangements.

Let $\mathcal{A}$ be an intersecting, cylindrical arrangement of n pseudocircles. Using Proposition 2.3, we can represent $\mathcal{A}$ as a pseudoparabola arrangement, in which we label the curves from top to bottom by $C_1, \ldots, C_n$. The idea is to flip the pseudoparabolas downwards one by one in the order of increasing indices, as illustrated in Figure 5. The availability of suitable triangle flips follows from Lemma 2.1 and our construction to consider pseudoparabolas from top to bottom. This completes the proof sketch for Theorem 1.2.

3.2 Proof Sketch of Theorem 1.1: Connectivity Intersecting

Let $\mathcal{A}$ be an arrangement of n pairwise intersecting pseudocircles. We show by induction on n that $\mathcal{A}$ can be transformed into a cylindrical arrangement with a finite number of triangle flips. The flip-connectivity then follows from Theorem 1.2.

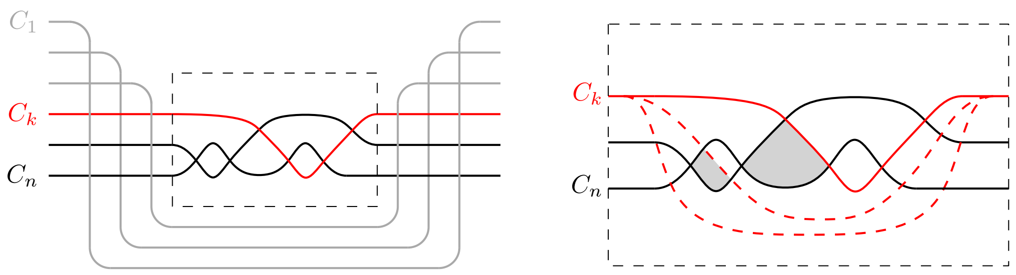


Figure 5 Illustration of the proof of Theorem 1.2: flipping the pseudoparabola C_k downwards.

The induction base is trivially fulfilled for $n = 2$. For the induction step, we choose a designated point p which lies inside a maximum number of pseudocircles. If p lies in the interior of all pseudocircles, then $\mathcal{A}$ is already cylindrical and we are done.

Hence, we may assume that there exists a pseudocircle C which does not contain p in its interior. We show how to expand C until containing p, using only triangle flips. First observe that Lemma 2.1 guarantees the existence of a flip to expand C. Since C already intersects all other pseudocircles, this must be a triangle or digon-collapse flip. As long as there exists a triangle flip expanding C, we perform it and transform $\mathcal{A}$ accordingly.

Suppose now that C does not yet contain p and can only be expanded by collapsing a digon formed with another pseudocircle C'. Then all remaining pseudocircles must intersect the lens $\mathrm{int}(C) \cap \mathrm{ext}(C')$ transversally (see Figure 6). Hence, using Lemma 2.2, we can expand C' until C and C' are parallel. We say that two pseudocircles C and C' are *parallel* in $\mathcal{A}$ if every vertex of $\mathcal{A} - \{C, C'\}$ lies in $(\mathrm{int}(C) \cap \mathrm{int}(C'))$ or in $(\mathrm{ext}(C) \cap \mathrm{ext}(C'))$.

Next consider the arrangement $\mathcal{A}' := \mathcal{A} - C'$ which is obtained by deleting C' from $\mathcal{A}$. By the induction hypothesis, $\mathcal{A}'$ can be transformed into a cylindrical arrangement by a finite sequence of triangle flips. We now carefully mimic this flip sequence on $\mathcal{A}$, while maintaining that C and C' are parallel. Suppose that a triangle T in $\mathcal{A}'$ is flipped. If none of the edges of T belongs to C, we can directly apply this triangle flip also in $\mathcal{A}$. If one of the edges e of T belongs to C and e is crossed by C' then the digon D is located along e. In this case, we apply two triangle flips to C' so that the digon is transferred to one of the two neighboring edges of C as illustrated in Figure 6, obtaining that e is not crossed by C' (without changing $\mathcal{A}'$). Finally, if e is not crossed by C' then we apply the according triangle flip twice, namely, once for C and once for C'. This completes the proof sketch.

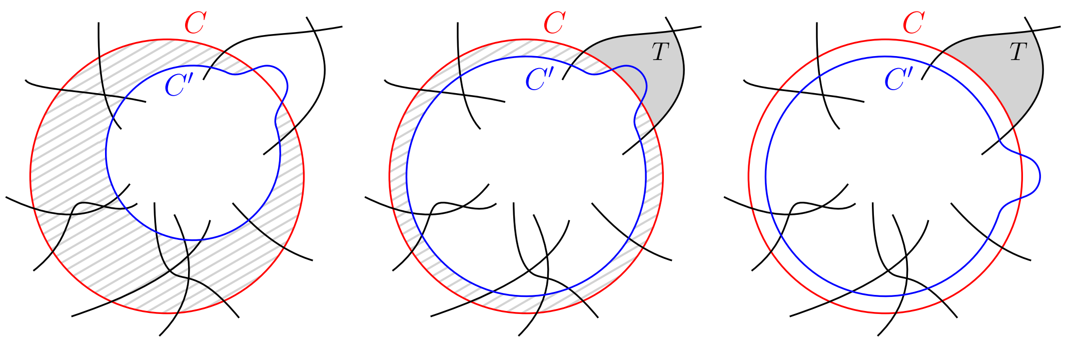


Figure 6 *Left:* C forms a digon with C' that is in $\mathrm{ext}(C) \cap \mathrm{int}(C')$. *Middle:* flip C' so that it becomes parallel to C. *Right:* flip C' so that T becomes also a triangle in $\mathcal{A}$.

4 Conclusion

While we have proven part (1) of Conjecture 1, part (2) remains a challenging open question. Also, several questions concerning the structure of the flip graphs remain open, such as Hamiltonicity or the degree of connectivity. The maximum degree Δ of the flip graph corresponds to the maximum number of triangles among all arrangements and $\Delta = \frac{4}{3}\binom{n}{2} + O(n)$ is known [6]. The minimum degree δ corresponds to the minimum number of triangles among all arrangements. For (not necessarily digon-free) intersecting arrangements $\frac{2n}{3} \leq \delta \leq n-1$ is known and $\delta = n-1$ is conjectured for $n \geq 3$. For digon-free intersecting arrangements $\delta = \max\{8, \lceil\frac{4n}{3}\rceil\}$ holds for $n \geq 3$ [4, 6].

References

1 Pankaj K. Agarwal, Eran Nevo, János Pach, Rom Pinchasi, Micha Sharir, and Shakhar Smorodinsky. Lenses in arrangements of pseudo-circles and their applications. *Journal of the ACM*, 51(2):139–186, 2004. `doi:10.1145/972639.972641`.

2 Prosenjit Bose and Ferran Hurtado. Flips in planar graphs. *Computational Geometry*, 42(1):60–80, 2009. `doi:10.1016/j.comgeo.2008.04.001`.

3 Bette Bultena, Branko Grünbaum, and Frank Ruskey. Convex drawings of intersecting families of simple closed curves. In *Proceedings of the 11th Canadian Conference on Computational Geometry (CCCG'99)*, pages 18–21, 1999. URL: `http://www.cccg.ca/proceedings/1999/c14.pdf`.

4 Stefan Felsner, Sandro Roch, and Manfred Scheucher. Arrangements of pseudocircles: On digons and triangles, 2022. To appear in Proc. of GD'22. arXiv:2208.12110.

5 Stefan Felsner and Manfred Scheucher. Arrangements of Pseudocircles: On Circularizability. *Discrete & Computational Geometry, Ricky Pollack Memorial Issue*, 64:776–813, 2020. `doi:10.1007/s00454-019-00077-y`.

6 Stefan Felsner and Manfred Scheucher. Arrangements of Pseudocircles: Triangles and Drawings. *Discrete & Computational Geometry*, 65:261–278, 2021. `doi:10.1007/s00454-020-00173-4`.

7 B. Grünbaum. *Arrangements and Spreads*, volume 10 of *CBMS Regional Conference Series in Mathematics*. American Mathematical Society, 1972. `doi:10.1090/cbms/010`.

8 Torsten Mütze. Combinatorial Gray codes-an updated survey, 2022. arXiv:2202.01280.

9 Naomi Nishimura. Introduction to reconfiguration. *Algorithms*, 11(4):52, 2018. `doi:10.3390/a11040052`.

10 Gerhard Ringel. Teilungen der Ebene durch Geraden oder topologische Geraden. *Mathematische Zeitschrift*, 64:79–102, 1956. `doi:10.1007/BF01166556`.

11 Jack Snoeyink and John Hershberger. Sweeping arrangements of curves. In Jacob Eli Goodman, Richard Pollack, and William L. Steiger, editors, *Discrete & Computational Geometry: Papers from the DIMACS Special Year*, volume 6 of *DIMACS*, pages 309–349. AMS, 1991. `doi:10.1090/dimacs/006/21`.

On the number of iterations of the DBA algorithm

Frederik Brüning[1], Anne Driemel[2], Alperen Ergür[3], and Heiko Röglin[4]

1 **Department of Computer Science, University of Bonn, Germany**
2 **Hausdorff Center for Mathematics, University of Bonn, Germany**
3 **The University of Texas at San Antonio, partially supported by NSF CCF 2110075**
4 **Department of Computer Science, University of Bonn, Germany**

Abstract

The DTW Barycenter Averaging (DBA) algorithm is a widely used algorithm for estimating the mean of a given set of point sequences. In this context, the mean is defined as a point sequence that minimises the sum of dynamic time warping distances (DTW). In our paper, we aim to initiate a theoretical study of the number of iterations that the algorithm performs until convergence. We assume the algorithm is given n sequences of m points in $\mathbb{R}^d$ and a parameter k that specifies the length of the mean sequence to be computed. In an attempt to better understand the performance of the algorithm on non-degenerate input, we study DBA in the model of smoothed analysis, upper-bounding the expected number of iterations in the worst case under random perturbations of the input. Our smoothed upper bound is polynomial in k, n and d, and in case n is a constant, also polynomial in m.

1 Introduction

The DTW Barycenter Averaging (DBA) algorithm [11] was introduced by Petitjean, Ketterlin and Gançarski in 2011 and has since been used in many applications for clustering time series data. The objective is to find a representative time series that optimally summarizes a given set of input time series. Here, the optimality is measured using the sum of dynamic time warping (DTW) distances to the input sequences. The DBA algorithm has found application in the context of optimization of energy systems [15], forecasting of household electricity demand [14], and human activity recognition [13] to name a few. The representation computed by the DBA algorithm is used to speed up classification tasks on time series [10] and to improve the training of neural networks [5, 6]. Despite its popularity, the DBA algorithm is a heuristic in the double meaning that it neither comes with any guarantees on the quality of the solution nor on the running time.

Objective In this paper, we initiate a study of the number of iterations that the DBA algorithm performs until convergence. While convergence properties of the algorithm have been studied before [12], it seems that the running time of the algorithm has not been the subject of rigorous study up to now. We follow a line of thought that has proved successful for the closely related k-means algorithm by Stuart Lloyd [9]. For this algorithm it is known that the number of iterations in the worst case is exponential [2, 7, 8, 16]. However, on most practical instances k-means is reported to converge very fast. Moreover, using smoothed analysis, it has been shown that the expected running time under random perturbations of the input is merely polynomial [1].

39th European Workshop on Computational Geometry, Barcelona, Spain, March 29–31, 2023.

Preliminaries For $n \in \mathbb{N}$, we define $[n]$ as the set $\{1, \ldots, n\}$. We call an ordered sequence of points $p_1, \ldots, p_m$ in $\mathbb{R}^d$ a ***point sequence*** of length m. For two points p, q in $\mathbb{R}^d$, we denote with $\|p-q\|^2$ the ***squared Euclidean distance***, where $\|.\|$ is the standard Euclidean norm. For $m_1, m_2 \in \mathbb{N}$, each sequence $(1,1) = (i_1, j_1), (i_2, j_2), \ldots, (i_M, j_M) = (m_1, m_2)$ such that $i_k - i_{k-1}$ and $j_k - j_{k-1}$ are either 0 or 1 for all k is a ***warping path*** from $(1,1)$ to (m_1, m_2). We denote with $\mathcal{W}_{m_1,m_2}$ the set of all warping paths from $(1,1)$ to (m_1, m_2). For any two point sequences $\gamma_1 = (\gamma_{1,1}, \ldots, \gamma_{1,m_1}) \in (\mathbb{R}^d)^{m_1}$ and $\gamma_2 = (\gamma_{2,1}, \ldots, \gamma_{2,m_2}) \in (\mathbb{R}^d)^{m_2}$, we also write $\mathcal{W}_{\gamma_1,\gamma_2} = \mathcal{W}_{m_1,m_2}$, and call elements of $\mathcal{W}_{\gamma_1,\gamma_2}$ warping paths between γ_1 and γ_2. The ***dynamic time warping distance*** between the sequences γ_1 and γ_2 is defined as

$$d_{\mathrm{DTW}}(\gamma_1, \gamma_2) = \min_{w \in \mathcal{W}_{\gamma_1,\gamma_2}} \sum_{(i,j) \in w} \|\gamma_{1,i} - \gamma_{2,j}\|^2$$

A warping path that attains the above minimum is also called an ***optimal warping path*** between γ_1 and γ_2. We denote with $\mathcal{W}^*_{m_1,m_2} \subset \mathcal{W}_{m_1,m_2}$ the set of warping paths w such that there exist point sequences $\gamma_1 \in (\mathbb{R}^d)^{m_1}$ and $\gamma_2 \in (\mathbb{R}^d)^{m_2}$ with this optimum warping path w. Let $X = \{\gamma_1, \ldots, \gamma_n\} \subset (\mathbb{R}^d)^m$ be a set of n point sequences of length m and $C \in (\mathbb{R}^d)^k$ be a point sequence of length k. We call a sequence π of tuples (p, c) where p is an element of some point sequence in X and c is an element of the point sequence C an ***assignment map*** between X and C. We call an assignment map between X and C ***valid*** if for each $1 \leq i \leq n$ the sequence of all tuples (p, c) of π for which p is a point of γ_i forms a warping path $w(\pi)_i$ between γ_i and C. We call a valid assignment map ***optimal*** if for each $1 \leq i \leq n$ the formed warping path is an optimal warping path. We define the ***cost*** of an ***assignment map*** π as $\Phi(\pi) = \sum_{(p,c) \in \pi} \|p - c\|^2$. Similarly, we define the ***total warping distance*** of a valid assignment map π with respect to a point sequence $x = (x_1, \ldots, x_k)$ as $\Psi_\pi(x) = \sum_{i=1}^n \sum_{(j_1,j_2) \in w(\pi)_i} \|\gamma_{i,j_1} - x_{j_2}\|^2$.

1.1 The DBA Algorithm

Let X be a set of n point sequences $\gamma_1, \ldots, \gamma_n \in (\mathbb{R}^d)^m$. Let $C \in (\mathbb{R}^d)^k$ be another point sequence and $w^{(1)}, \ldots, w^{(n)} \in \mathcal{W}_{m,k}$ be chosen such that $w^{(i)}$ is an optimal warping path between γ_i and C. Then $w^{(1)}, \ldots, w^{(n)}$ define an optimal assignment map π between X and C. The assignment map π can be represented by sets $S_1(\pi), S_2(\pi), \ldots, S_k(\pi)$, where

$$S_i(\pi) = \cup_{j=1}^n \{\gamma_{j,t} \mid (i,t) \in w^{(j)}\}$$

is the union of points over all n point sequences that are assigned to the i'th point of the average point sequence C. By construction, the assignment map π minimizes the DTW distance between the input point sequences $\gamma_1, \ldots, \gamma_n \in (\mathbb{R}^d)^m$ and C.

In the opposite direction, we can also create an average point sequence, that minimizes the sum of DTW distances for a fixed assignment map π. To this end, we define

$$C_\pi = (c_1(\pi), c_2(\pi), c_3(\pi), \ldots, c_k(\pi)) \text{ where } c_i(\pi) := \frac{1}{|S_i(\pi)|} \sum_{p \in S_i(\pi)} p.$$

The DBA algorithm alternately computes assignment maps and average point sequences in the manner described above. It goes as follows.

1. Let π_0 be an initial assignment map (e.g. defined by $w_0^{(1)}, \ldots, w_0^{(n)}$ drawn uniformly at random from $\mathcal{W}_{m,k}$). Let $j \leftarrow 0$.
2. Let $j \leftarrow j + 1$. Compute the average point sequence $C_{\pi_{j-1}}$ based on π_{j-1}.

3. Compute optimal warping paths $w_j^{(i)}$ between γ_i and $C_{\pi_{j-1}}$ for all $1 \leq i \leq n$. The warping paths define an optimal assignment map π_j between X and $C_{\pi_{j-1}}$.
4. If $\Phi(\pi_j) \neq \Phi(\pi_{j-1})$, then go to Step 2. Otherwise, terminate.

2 Upper bound based on geometric assumptions on the input data

We denote the set of valid assignment maps from n input point sequences of length m to a point sequence of length k with $\mathcal{A}_{n,m,k}$. It is $|\mathcal{A}_{n,m,k}| \leq m^{kn}$. For an assignment map $\pi \in \mathcal{A}_{n,m,k}$ and a point sequence $x = (x_1, x_2, \dots, x_k)$ with $x_i \in \mathbb{R}^d$, we can rewrite the total warping distance as

$$\Psi_\pi(x) := \sum_{i=1}^{k} \sum_{y \in S_i(\pi)} \|y - x_i\|^2$$

Our first observation is that for all x we have $\Psi_\pi(x) \geq \Psi_\pi(C_\pi)$. We would like to express $\Psi_\pi(x)$ in a way that resembles an inertia. We set $I_\pi := \sum_{i=1}^{k} \sum_{y \in S_i(\pi)} (\|y\|^2 - \|c_i(\pi)\|^2)$. Observe that $I_\pi = \Psi_\pi(C_\pi)$. We see I_π as the inertia of the assignment map π, and we have

$$\Psi_\pi(x) = I_\pi + \sum_{i=1}^{k} |S_i(\pi)| \, \|x_i - c_i(\pi)\|^2$$

This section uses two assumptions on the input data: First assumption is a basic normalization assumption, and the second one is a non-degeneracy assumption.
Normalization Assumption: We assume the input data is normalized so that for any vector y in any input point sequence we have $\|y\|^2 \leq B$.
Separation Assumption: For any two different assignment maps $\alpha, \beta \in \mathcal{A}_{n,m,k}$ we have

$$\|C_\alpha - C_\beta\|^2 \geq \varepsilon$$

With the help of the following lemma, we can derive an upper bound in Theorem 2.2.

▶ **Lemma 2.1.** *For every $\alpha \in \mathcal{A}_{n,m,k}$, we have $I_\alpha \leq Bn(m + k)$.*

Proof.

$$I_\alpha = \sum_{i=1}^{k} \sum_{y \in S_i(\alpha)} (\|y\|^2 - \|c_i(\alpha)\|^2) \leq B \sum_{i=1}^{k} |S_i(\alpha)|$$

Note that every warping path between the average point sequence of length k and an input point sequence of length m consists of at most $m + k$ many steps. This means every point sequence contributes to the sum $\sum_{i=1}^{k} |S_i(\alpha)|$ with at most $m + k$ elements, and hence we have $\sum_{i=1}^{k} |S_i(\alpha)| \leq n(m + k)$. ◀

▶ **Theorem 2.2.** *If the input data satisfies the normalization assumption with parameter B and separation assumption with parameter ε, then the number of steps performed by the DBA algorithms is at most*

$$\frac{B(m + k)}{\varepsilon}$$

Proof. Suppose the DBA algorithm has started from assignment map π_0. If the DBA algorithm takes a step from some assignment map α to some assignment map β this means

$$I_\alpha = \Psi_\alpha(C_\alpha) > \Psi_\beta(C_\alpha) = I_\beta + \sum_{i=1}^{k} |S_i(\beta)| \, \|c_i(\alpha) - c_i(\beta)\|^2$$

Since $|S_i(\beta)| \geq n$ for all i, this implies $I_\alpha > I_\beta + n\|C_\alpha - C_\beta\|^2$. Using the seperation assumption on the data and Lemma 2.1 , we get

$$I_\alpha > I_\beta + n\varepsilon \geq I_\beta + \frac{\varepsilon}{B(m+k)} I_{\pi_0} \tag{1}$$

Let π_T be the assignment map after T steps of DBA, then we have by (1) that

$$I_{\pi_T} < I_{\pi_0} - T\frac{\varepsilon}{B(m+k)} I_{\pi_0}$$

◀

3 Smoothed Analysis

Our randomness model is as follows: An adversary specifies an instance $X' \in ([0,1]^d)^{nm}$ of n point sequences $\gamma_1, \ldots, \gamma_n$ of length m in $[0,1]^d$, where each sequence γ_i is given by its m points $\gamma_i = (\gamma_{i,1}, \ldots, \gamma_{i,m})$. Then we add to each vertex of X' an independent d-dimensional random vector with independent Gaussian coordinates of mean 0 and standard deviation σ. The resulting vectors form the input point sequences. We assume without loss of generality that $\sigma \leq 1$, since the case $\sigma > 1$ corresponds to a scaled down instance $X' \in ([0, \frac{1}{\sigma}]^d)^{nm}$ with additive d-dimensional Gaussian random vectors with mean 0 and standard deviation 1. We call this randomness model m-length sequences with $\mathcal{N}(0,\sigma)$ perturbation.

We note that the results in this section hold for a more general family of random input models (See Section 1.5 of [4] or Section 3.1 of [3]). We conduct the analysis only for Gaussian perturbation for the sake of simplicity and obtain the following theorem.

▶ **Theorem 3.1.** *Suppose $d \geq 2$, then the expected number of iterations until DBA converges is at most*

$$O\left(\frac{n^2 m^{8\frac{n}{d}+6} d^4 k^6 \ln(nm)^4}{\sigma^2}\right).$$

To proof Theorem 3.1, we first bound the probability that the normalization assumption and the separation assumption hold for suitable parameters.

▶ **Lemma 3.2.** *Let $\gamma_1, \gamma_2, \ldots, \gamma_n$ be independent m-length sequences with $\mathcal{N}(0,\sigma)$ perturbation. Then, we have*

$$\mathbb{P}\{\max_{1\leq i\leq n} \max_{y\in\gamma_i} \|y\| \geq \sqrt{d} + t\sigma\sqrt{2d\ln nm}\} \leq e^{1-t^2}$$

Proof. By our assumptions every vector in input sequences is given by $D + Y$ where D is a deterministic vector with norm at most $\sqrt{d}$ and Y is a random vector with Gaussian i.i.d coordinates $\mathcal{N}(0,\sigma)$. By triangle inequality $\|D+Y\| \leq \sqrt{d} + \|Y\|$. Since Y has Gaussian i.i.d coordinates $\mathcal{N}(0,\sigma)$, we can apply the standard tail bound $\mathbb{P}\{\|Y\| \geq t\sigma\sqrt{d}\} \leq e^{1-\frac{t^2}{2}}$. ◀

▶ **Lemma 3.3.** *Let $\gamma_1, \gamma_2, \ldots, \gamma_n \in \mathbb{R}^{dm}$ be m-length point sequences with $\mathcal{N}(0,\sigma)$ perturbation. Let C_α and C_β be two different average point sequences corresponding to assingment maps α and β. Then, we have $\mathbb{P}\{\|C_\alpha - C_\beta\|^2 \leq \varepsilon\} \leq \left(\frac{nm\sqrt{\varepsilon}}{\sigma}\right)^d$. Furthermore, the separation assumption with parameter ε holds on $\gamma_1, \gamma_2, \ldots, \gamma_n$ with probability at least*

$$1 - m^{4n}\left(\frac{nm\sqrt{\varepsilon}}{\sigma}\right)^d.$$

Proof. Since the instances are perturbed and the assignment maps are different, we have with probability 1 that there exists an $i \in [k]$ such that $c_i(\alpha) \neq c_i(\beta)$. The event $\|C_\alpha - C_\beta\|^2 \leq \varepsilon$ further implies $\|c_i(\alpha) - c_i(\beta)\| \leq \sqrt{\varepsilon}$. We first bound the probability that this event $\|c_i(\alpha) - c_i(\beta)\| \leq \sqrt{\varepsilon}$ occurs for the fixed $c_i(\alpha)$ and $c_i(\beta)$.

Let $S_i(\alpha)$ and $S_i(\beta)$ denote the sets of points in X that were assigned to $c_i(\alpha)$ and $c_i(\beta)$ respectively. Since $c_i(\alpha) \neq c_i(\beta)$, it immediately follows that $S_i(\alpha) \neq S_i(\beta)$. So we can fix a point $s \in S_i(\alpha) \triangle S_i(\beta)$. We let an adversary fix all points in $S_i(\alpha) \cup S_i(\beta) \setminus \{s\}$. In order for $c_i(\alpha)$ and $c_i(\beta)$ to be $\sqrt{\epsilon}$-close, we need s to fall into a hyperball of radius $nm\sqrt{\epsilon}$. Because s is drawn from a Gaussian distribution with standard deviation σ, this happens with probability at most $\left(\frac{nm\sqrt{\epsilon}}{\sigma}\right)^d$. So in total, we have

$$\mathbb{P}\{\|C_\alpha - C_\beta\|^2 \leq \varepsilon\} \leq \mathbb{P}\{\|c_i(\alpha) - c_i(\beta)\| \leq \sqrt{\varepsilon}\} \leq \left(\frac{nm\sqrt{\varepsilon}}{\sigma}\right)^d$$

To prove the second claim, we apply a union bound over all possible choices for $c_i(\alpha)$ and $c_i(\beta)$. Since each $c_i(\alpha)$ and $c_i(\beta)$ is uniquely determined by its assigned points $S_i(\alpha)$ and $S_i(\beta)$ it suffices to bound these. For each input point sequence, there are at most $\binom{m}{2}$ possible choices for the set of points that get assigned to a fixed center point: This is the case since all points that get assigned to the same center point have to be consecutive. So the points that get assigned to the center point are uniquely determined by the first and the last point that gets assigned to the center point. For n input point sequences all possible assignments to an arbitrary center point are therefore bounded by $\binom{m}{2}^n$. Since we choose two center points $c_i(\alpha)$ and $c_i(\beta)$, there are at most

$$\binom{m}{2}^{2n} \leq m^{4n}$$

possible choices for the assigned points $S_i(\alpha)$ and $S_i(\beta)$ that determine $c_i(\alpha)$ and $c_i(\beta)$. The statement follows by applying the union bound over possible choices for $c_i(\alpha)$ and $c_i(\beta)$. ◀

As a combination of Lemma 3.2 and Lemma 3.3, we get the following Lemma.

▶ **Lemma 3.4.** *Let $\gamma_1, \gamma_2, \ldots, \gamma_n$ be independent m-length sequences with $\mathcal{N}(0,\sigma)$ perturbation, and suppose $d \geq 2$ and $k \leq m$. Then, the DBA algorithm implemented on the input data $\gamma_1, \gamma_2, \ldots, \gamma_n$ converges in at most*

$$s\left(\frac{a_1 n^2 m^{8\frac{n}{d}+5} d^3 k^5 \ln(nm)^3}{\sigma^2}\right)$$

steps with probability at least $1 - s^{-\frac{d}{2}} - (2n)^{-dk} - 2^{-8mdk^2}$ where a_1 is constant.

Proof. In Lemma 3.2, we set $t = 2\ln((2n)^{dk}2^{8mdk^2}) \leq 16mdk^2\ln(4n)$ to get that the normalization assumption is fulfilled for some $B \leq a_2\sigma^2 d^3 k^4 m^2 \ln(nm)^3$ with probability at least $1 - (2n)^{-dk} - 2^{-8mdk^2}$ where a_2 is constant. Then we use Lemma 3.3 with

$$\varepsilon = \frac{\sigma^2}{sn^2 m^{8\frac{n}{d}+2}}$$

and apply Theorem 2.2. ◀

With the help of the Lemma 3.4, we are now ready to prove Theorem 3.1.

Proof of Theorem 3.1. Let X be the number of steps that DBA performs. In the full version, we show with some classical tools from real algebraic geometry that $X \leq a_2(2n)^{dk}2^{8mdk^2}$ for some constant a_2. Let´s set $M := a_2(2n)^{dk}2^{8mdk^2}$ for simplicity. We have

$$\mathbb{E}[X] = \sum_{i=1}^{M} \mathbb{P}\{X \ \geq i\} \leq K + \sum_{t=K}^{M} \mathbb{P}\{X \ \geq t\}$$

for any K. We set $K := a_1 n^2 m^{8\frac{n}{d}+5} d^3 k^5 \ln(nm)^3$. By Lemma 3.4, it is

$$\mathbb{P}\{X \ \geq sK\} \leq s^{-\frac{d}{2}} + (2n)^{-dk} + m^{-8mdk^2}$$

for all $s \geq 1$. Therefore, we have

$$\sum_{t=K}^{M} \mathbb{P}\{X \geq t\} \leq K \cdot \sum_{s=1}^{\frac{M}{K}} s^{-\frac{d}{2}} + (2n)^{-dk} + m^{-8mdk^2}$$

Since $d \geq 2$, we have $s^{\frac{-d}{2}} \leq \frac{1}{s}$. Moreover, $\frac{M}{K}\left((2n)^{-dk} + m^{-8mdk^2}\right) \leq 1$. So, we have

$$\sum_{t=K}^{M} \mathbb{P}\{X \geq t\} \leq K\left(1 + \sum_{s=1}^{\frac{M}{K}} \frac{1}{s}\right) \leq K + K \ln \frac{M}{K}$$

Hence,

$$\mathbb{E}X = \sum_{t=1}^{M} \mathbb{P}\{X \geq t\} \leq K + \sum_{t=K}^{M} \mathbb{P}\{X \geq t\} \leq 2K + K \ln \frac{M}{K} \leq 2K + Kmdk^2 \ln\left(\frac{4a_2}{a_1} n\right)$$

◀

▶ Remark. Note that for the discrete case, where the positions of the points in the center point sequence are restricted to the input points, we would get an upper bound on the number of iterations which would be polynomial in n, instead of exponential in n, since for the positions of $c_i(\alpha)$ and $c_i(\beta)$ in the proof of Lemma 3.3, there are only $\binom{nm}{2}$ instead of $\binom{m}{2}^{2n}$ possible choices.

In the full version, we complement these results with an exponential worst-case lower bound. More specifically, we show that there is an instance of two point sequences with length $m = \Theta(k)$ in the plane such that the DBA algorithm needs $2^{\Omega(k)}$ iterations to converge. Our techniques used to construct this lower bound borrow from earlier work of Vattani [16] on the k-means algorithm. Interestingly, our construction only needs $n = 2$ sequences. Note that in case n is a constant, our smoothed upper bound is polynomial in k, m, n and d, avoiding these artificially constructed boundary cases.

References

1 David Arthur, Bodo Manthey, and Heiko Röglin. Smoothed analysis of the k-means method. *J. ACM*, 58(5), October 2011.

2 David Arthur and Sergei Vassilvitskii. How slow is the k-means method? In *Proceedings of the Twenty-Second Annual Symposium on Computational Geometry*, SCG '06, page 144–153, New York, NY, USA, 2006. Association for Computing Machinery.

3 Felipe Cucker, Alperen A Ergür, and Josue Tonelli-Cueto. Plantinga-vegter algorithm takes average polynomial time. In *Proceedings of the 2019 on international symposium on symbolic and algebraic computation*, pages 114–121, 2019.

4 Alperen Ergür, Grigoris Paouris, and J Rojas. Smoothed analysis for the condition number of structured real polynomial systems. *Mathematics of Computation*, 2021.

5 Hassan Ismail Fawaz, Germain Forestier, Jonathan Weber, Lhassane Idoumghar, and Pierre-Alain Muller. Data augmentation using synthetic data for time series classification with deep residual networks. *arXiv preprint arXiv:1808.02455*, 2018.

6 Germain Forestier, François Petitjean, Hoang Anh Dau, Geoffrey I Webb, and Eamonn Keogh. Generating synthetic time series to augment sparse datasets. In *2017 IEEE international conference on data mining (ICDM)*, pages 865–870. IEEE, 2017.

7 Sariel Har-Peled and Bardia Sadri. How fast is the k-means method? *Algorithmica*, 41(3):185–202, 2005.

8 Mary Inaba, Naoki Katoh, and Hiroshi Imai. Variance-based k-clustering algorithms by voronoi diagrams and randomization. *IEICE Transactions on Information and Systems*, 83(6):1199–1206, 2000.

9 Stuart Lloyd. Least squares quantization in pcm. *IEEE transactions on information theory*, 28(2):129–137, 1982.

10 François Petitjean, Germain Forestier, Geoffrey I Webb, Ann E Nicholson, Yanping Chen, and Eamonn Keogh. Faster and more accurate classification of time series by exploiting a novel dynamic time warping averaging algorithm. *Knowledge and Information Systems*, 47(1):1–26, 2016.

11 François Petitjean, Alain Ketterlin, and Pierre Gançarski. A global averaging method for dynamic time warping, with applications to clustering. *Pattern Recognition*, 44(3):678–693, 2011.

12 David Schultz and Brijnesh Jain. Nonsmooth analysis and subgradient methods for averaging in dynamic time warping spaces. *Pattern Recognition*, 74:340–358, 2018.

13 Skyler Seto, Wenyu Zhang, and Yichen Zhou. Multivariate time series classification using dynamic time warping template selection for human activity recognition. In *2015 IEEE symposium series on computational intelligence*, pages 1399–1406. IEEE, 2015.

14 Thanchanok Teeraratkul, Daniel O'Neill, and Sanjay Lall. Shape-based approach to household electric load curve clustering and prediction. *IEEE Transactions on Smart Grid*, 9(5):5196–5206, 2017.

15 Holger Teichgraeber and Adam R Brandt. Clustering methods to find representative periods for the optimization of energy systems: An initial framework and comparison. *Applied energy*, 239:1283–1293, 2019.

16 Andrea Vattani. k-means requires exponentially many iterations even in the plane. *Discrete & Computational Geometry*, 45(4):596–616, 2011.

On Degeneracy in the P-Matroid Oriented Matroid Complementarity Problem[*]

Michaela Borzechowski[1] and Simon Weber[2]

1 Department of Mathematics and Computer Science, FU Berlin
michaela.borzechowski@fu-berlin.de

2 Department of Computer Science, ETH Zurich
simon.weber@inf.ethz.ch

Abstract

We investigate degeneracy in the P-Matroid Oriented Matroid Complementarity Problem (P-OMCP) and its impact on the reduction of this problem to sink-finding in Unique Sink Orientations (USOs). On one hand, this understanding of degeneracies allows us to prove a linear lower bound for sink-finding in *P-matroid USOs*. On the other hand, it allows us to prove a promise preserving reduction from P-OMCP to USO sink-finding, where we can drop the assumption that the given P-OMCP is non-degenerate. This places the promise version of P-OMCP in the complexity class PromiseUEOPL.

Related Version arXiv:2302.14585

1 Introduction

Degenerate input can be an issue in structural analysis and algorithm design for many algebraic and geometric problems. It is often swept under the rug by assuming the input to be non-degenerate. For example, one often assumes all input points of a geometric problem to be in general position. In some problems (e.g., the minimum convex partition [7]), such an assumption is inappropriate as it makes the problem considerably easier. In other cases, degenerate inputs can be solved easily by resolving degeneracy using *perturbation* techniques.

In this paper, we investigate degeneracy in the context of the P-Matroid Oriented Matroid Complementarity Problem (P-OMCP). Assuming non-degeneracy, this problem can be solved by converting it into a Unique Sink Orientation of the hypercube graph, and finding a sink within that orientation. Oriented matroids are abstractions for many types of configurations of geometric objects, such as (pseudo-)hyperplane arrangements or point configurations. Just like these geometric configurations, oriented matroids can exhibit degeneracies. In this paper, we analyze the effects of these degeneracies on the reduction from P-OMCP to Unique Sink Orientation sink-finding.

Both the P-OMCP as well as Unique Sink Orientations are combinatorial abstractions of the P-Matrix linear complementarity problem (P-LCP). The complexity status of the P-LCP remains an interesting and relevant open question, since the problem can be used to solve many optimization problems, such as Linear Programming [6], and binary Simple Stochastic Games [5, 10]. Sink-finding in Unique Sink Orientations can also be used to solve geometric problems such as the problem of finding the smallest enclosing ball of a set of balls [4].

[*] Michaela Borzechowski is supported by the German Research Foundation DFG within the Research Training Group GRK 2434 *Facets of Complexity*. Simon Weber is supported by the Swiss National Science Foundation under project no. 204320.

2 Background

2.1 P-OMCP

We consider *oriented matroids* $\mathcal{M} = (E_{2n}, \mathcal{C})$ given in circuit representation, where the ground set $E_{2n} = S \cup T$ is made up of two parts $S = \{s_1, \ldots, s_n\}$ and $T = \{t_1, \ldots, t_n\}$, $S \cap T = \emptyset$. The set $\mathcal{C}$ is the collection of circuits of $\mathcal{M}$. We only introduce the notation specific to this setting, and refer readers unfamiliar with oriented matroids to the full version of this paper or textbooks like [9].

We call a set $J \subseteq E_{2n}$ *complementary*, if it contains no *complementary pair* s_i, t_i.

▶ **Definition 2.1** (P-matroid). An oriented matroid $\mathcal{M} = (E_{2n}, \mathcal{C})$ is a *P-matroid* if S is a basis and there is no *sign-reversing circuit*. A sign-reversing circuit is a circuit X such that for each complementary pair s_i, t_i contained in $\underline{X}$, $X_{s_i} = -X_{t_i}$.

Let q be such that $q \notin E_{2n}$. Then $\widehat{E_{2n}} := S \cup T \cup \{q\}$, and $\widehat{\mathcal{M}} = (\widehat{E_{2n}}, \widehat{\mathcal{C}})$ is called an *extension of* $\mathcal{M}$, if its minor $\widehat{\mathcal{M}} \setminus q := (E_{2n}, \{X \mid X \in \widehat{\mathcal{C}} \text{ and } X_q = 0\})$ is equal to $\mathcal{M}$.

Given an extension $\widehat{\mathcal{M}} = (\widehat{E_{2n}}, \widehat{\mathcal{C}})$ of a P-matroid, the goal of the *P-Matroid Oriented Matroid Complementarity Problem (P-OMCP)* is to find a circuit $X \in \widehat{\mathcal{C}}$ with $X \geq 0$, $X_q = +$, and $X_{s_i} X_{t_i} = 0$ for every $i \in [n]$. The matroid extension is given to an algorithm by a circuit oracle, which given a set $B \subset \widehat{E_{2n}}$ and another element $e \in \widehat{E_{2n}} \setminus B$ either returns that B is not a basis of $\widehat{\mathcal{M}}$, or returns the *fundamental circuit* $C(B, e)$, which is the unique circuit $X \in \widehat{\mathcal{C}}$ with $X_e = +$ and $\underline{X} \subseteq B \cup \{e\}$. It is known that in P-matroids and P-matroid extensions, every complementary set $B \subset S \cup T$ of size n is a basis [8].

A P-matroid extension (a P-OMCP instance) is *non-degenerate*, if for every complementary basis B, the circuit $C(B, q)$ is non-zero on all elements in $B \cup \{q\}$. Every P-OMCP instance has a unique solution [13].

2.2 Unique Sink Orientation (USO)

The *n-dimensional hypercube graph Q_n* (*n-cube*) is the undirected graph on the vertex set $V(Q_n) = \{0,1\}^n$, where two vertices are connected by an edge if they differ in exactly one coordinate. An *orientation* $O : V(Q_n) \to \{-,+\}^n$ assigns each vertex an orientation of its incident edges, where $O(v)_i = +$ denotes an outgoing edge from vertex v in dimension i and $O(v)_i = -$ denotes an incoming edge. A *Unique Sink Orientation (USO)* is an orientation, such that every non-empty subcube contains exactly one sink, i.e., a unique vertex v with $O(v)_i = -$ for all dimensions i in the subcube [12].

▶ **Lemma 2.2** (Szabó-Welzl Condition [12]). *An orientation O of Q_n is USO if and only if for all pairs of distinct vertices $v, w \in V(Q_n)$, we have: $\exists i \in [n] : (v_i \neq w_i) \wedge (O(v)_i \neq O(w)_i)$.*

The classical algorithmic problem associated to USOs is that of finding the unique global sink v with $\forall i : O(v)_i = -$, with as few as possible queries to an oracle computing O.

2.3 Classical Reduction

Todd [13] showed that a non-degenerate P-OMCP given by a matroid $\widehat{\mathcal{M}} = (\widehat{E_{2n}}, \widehat{\mathcal{C}})$ can be translated to an USO of the n-cube. Every vertex v of the cube is associated with a complementary basis $B(v) \subset S \cup T$. For each $i \in [n]$, $s_i \in B(v)$ if $v_i = 0$, otherwise $t_i \in B(v)$. The orientation $O(v)$ is then computed using the fundamental circuit $C := C(B(v), q)$:

$$O(v)_i := \begin{cases} + & \text{if } C_{s_i} = - \text{ or } C_{t_i} = -, \\ - & \text{if } C_{s_i} = + \text{ or } C_{t_i} = +. \end{cases}$$

As the P-OMCP instance is non-degenerate, no other case can occur. Todd showed that the computed orientation O is USO, and that its sink v corresponds to a fundamental circuit $C(B(v), q)$ which is positive on all elements and thus a solution to the P-OMCP instance.

3 The Effect of Degeneracy on the Resulting USOs

In the above reduction, if the P-OMCP instance is degenerate, we can sometimes not decide which way to orient an edge since $C_{s_i} = C_{t_i} = 0$. For now, we leave these edges unoriented. This leads to a *partial orientation* of the hypercube, which is a function $O : V(Q_k) \to \{-, 0, +\}^k$ where $O(v)_i = 0$ denotes an unoriented edge. We call such a partial orientation arising from a degenerate P-OMCP a *partial P-matroid USO (PPU).* In this section we aim to understand the structure of unoriented edges in PPUs. All proofs can be found in the full version.

Not every partial orientation can be turned into an USO by directing the unoriented edges. We thus state the following condition inspired by the Szabó-Welzl condition:

▶ **Definition 3.1.** A partial orientation O is said to be *partially Szabó-Welzl* if for any two distinct vertices $v, w \in V(Q_k)$, either

$$O(v)_i = O(w)_i = 0 \text{ for all } i \text{ with } v_i \neq w_i, \text{ or} \tag{1}$$

$$\exists i : v_i \neq w_i \wedge \big((O(v)_i = + \wedge O(w)_i = -) \vee (O(v)_i = - \wedge O(w)_i = +)\big). \tag{2}$$

▶ **Lemma 3.2.** *A partial orientation O which is partially Szabó-Welzl can be extended to an USO by orienting all unoriented edges towards the endpoint with fewer 1s, i.e., downwards.*

▶ **Lemma 3.3.** *A partial P-matroid USO is partially Szabó-Welzl.*

▶ **Lemma 3.4.** *In a partial P-matroid USO, the unoriented edges form a set of vertex-disjoint faces. In each such face, the orientation is the same at every vertex.*

Lemmas 3.2 to 3.4, and [11, Corollary 6] imply that the unoriented subcubes of a PPU can in fact be oriented according to *any* USO.

4 Constructions Based on Degeneracy and Perturbations

In this section we show how existing constructions of oriented matroid extensions can be interpreted as constructions of (partial) P-matroid USOs. An extension $\widehat{\mathcal{M}}$ of an oriented matroid $\mathcal{M}$ can be uniquely described by a *localization*, a function σ from the set $\mathcal{C}^*$ of cocircuits of $\mathcal{M}$ to the set $\{-, 0, +\}$. Note that not every function $f : \mathcal{C}^* \to \{-, 0, +\}$ describes a valid extension and thus not every such function is a localization. We give some more background about localizations as well as all omitted proofs in the full version. The following lemma connects a localization to the circuits relevant to the resulting (partial) P-matroid USO.

▶ **Lemma 4.1.** *Let $\mathcal{M}$ be a P-matroid and let σ be a localization for $\mathcal{M}$ describing the extension $\widehat{\mathcal{M}}$. Then, for any complementary basis B of $\mathcal{M}$ (and thus also of $\widehat{\mathcal{M}}$), and every element $e \in B$, the sign of e in the fundamental circuit $C(B, q)$ of $\widehat{\mathcal{M}}$ is the opposite of the sign assigned by σ to the fundamental cocircuit $C^*(B, e)$ of $\mathcal{M}$.*

Las Vergnas [9] showed that the set of localizations is closed under composition, i.e., given two localizations σ_1, σ_2, the following function is a localization too:

$$\forall c \in \mathcal{C}^* : (\sigma_1 \circ \sigma_2)(c) := \begin{cases} \sigma_1(c), & \text{if } \sigma_1(c) \neq 0, \\ \sigma_2(c), & \text{otherwise.} \end{cases}$$

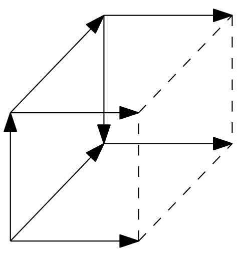


Figure 1 The form of the PPU given by a lexicographic extension of a uniform P-matroid.

Lemma 4.1 allows us to understand the effect of such composition on the resulting (partial) P-matroid USO: For localizations σ_1, σ_2 and their corresponding PPUs O_1, O_2, the PPU O' given by the localization $\sigma_1 \circ \sigma_2$ is

$$\forall v \in V(Q_k), i \in [k] : O'(v)_i = \begin{cases} O_1(v)_i, & \text{if } O_1(v)_i \neq 0, \\ O_2(v)_i, & \text{otherwise.} \end{cases}$$

This can be seen as filling in all unoriented subcubes of O_1 with the orientation O_2.

Furthermore, Las Vergnas [9] describes *lexicographic extensions* of oriented matroids. Lexicographic extensions of *uniform* P-matroids give rise to PPUs in which all edges of some dimension are oriented the same way, and one half is left unoriented while the other half is completely oriented (see Figure 1).

▶ **Lemma 4.2.** *Let $\mathcal{M} = (E_{2n}, \mathcal{C})$ be a P-matroid. Let $\widehat{\mathcal{M}}$ be the lexicographic extension of $\mathcal{M}$ by $[+ \cdot t_i]$. Then, in the partial P-matroid USO O defined by $\widehat{\mathcal{M}}$, the upper i-facet (the facet of vertices v with $v_i = 1$) is an unoriented subcube, and all i-edges point towards this facet. Furthermore, if $\mathcal{M}$ is uniform, the lower i-facet is completely oriented.*

We can use these two construction techniques to prove a lower bound on the number of queries needed by deterministic sink-finding algorithms on P-matroid USOs.

▶ **Theorem 4.3.** *Let $\mathcal{M} = (E_{2n}, \mathcal{C})$ be a uniform P-matroid. Then, for every deterministic sink-finding algorithm $\mathcal{A}$, there exists a non-degenerate extension $\widehat{\mathcal{M}}$ of $\mathcal{M}$ such that $\mathcal{A}$ requires at least n queries to find the sink of the P-matroid USO given by $\widehat{\mathcal{M}}$.*

In essence, to prove this theorem, we successively build a localization by composition with lexicographic extensions. The construction keeps the invariant that there exists an unoriented subcube guaranteed to contain the global sink. The dimension of this subcube is reduced by at most one with every query, thus at least n queries are required.

Previously, the best known lower bound for sink-finding on P-matroid USOs was $\Omega(\log n)$ queries [14]. In contrast, the stronger, almost-quadratic lower bound of Schurr and Szabó [11] does not apply to P-matroid USOs (for a proof of this see Appendix B of the full version).

The P-Matrix Linear Complementarity Problem (P-LCP) is an algebraic analogue of P-OMCP. We discuss our results (Sections 3 and 4) in the context of P-LCPs in Appendix A of the full version.

5 The Search Problem Complexity of P-OMCP

An instance of Unique End of Potential Line consists of an implicitly given exponentially large graph G, in which each vertex has a positive cost and in- and out-degree at most one. Thus, the graph is a collection of directed paths called *lines*. The computational task is as follows: if the nodes of G form a single line (that starts in some given start vertex) with strictly increasing cost, then find the unique end node of this line — a *sink*. Otherwise, either find *some* sink in G or a *violation certificate* that shows that G does not consist of a single line. Unique End of Potential Line is a total search problem, i.e., there always exists a sink or a violation. Note that there might exist a sink and a violation simultaneously.

▶ **Definition 5.1.** The search complexity class Unique End of Potential Line (UniqueEOPL) contains all problems that can be reduced in polynomial time to Unique End of Potential Line. Thus, the complexity class UniqueEOPL captures all total search problems where the space of candidate solutions has the structure of a unique line with increasing cost.

UniqueEOPL was introduced in 2018 by Fearnley et al. [2]. UniqueEOPL is a subclass of PPAD $\cap$ PLS [1]. Problems in UniqueEOPL are not known to be solvable in polynomial time.

The promise version of a total search problem with violations is to find a solution under the promise that no violations exist for the given instance. PromiseUEOPL is the promise version of the search problem class UniqueEOPL.

A search problem reduction from a problem R to a problem T is *promise preserving*, if every violation of T is mapped back to a violation of R and every valid solution of T is mapped back to a valid solution or a violation of R. Promise preserving reductions are transitive. When containment of a search problem R in UniqueEOPL is shown via a polynomial time, promise preserving reduction, the promise version of R is contained in PromiseUEOPL.

We now state the problem of USO sink-finding as a total search problem with a violation.

▶ **Definition 5.2.** Given an orientation function $O : \{0,1\}^n \to \{+,-\}^n$, the task of the total search problem Unique Sink Orientation Sink-Finding (USO-SF) is to find:

(U1) A vertex $v \in \{0,1\}^n$ such that $\forall i \in [n] : O(v)_i = -$. The vertex v is a sink.
(UV1) Two distinct vertices $v, w \in \{0,1\}^n$ with $\forall i \in [n] : \ (v_i = w_i) \vee (O(v)_i = O(w)_i)$. The orientation O does not fulfill the Szabó-Welzl condition and thus is not USO.

▶ **Lemma 5.3** ([2]). *USO-SF is in UniqueEOPL and its promise version is in PromiseUEOPL.*

Next, we define the P-OMCP problem as a total search problem with violations.

▶ **Definition 5.4.** Let $\widehat{\mathcal{M}} = (\widehat{E_{2n}}, \widehat{\mathcal{C}})$ be an oriented matroid with the set S being a basis. The task of the total search version of P-OMCP is to find one of the following:

(M1) A circuit $X \in \widehat{\mathcal{C}}$ such that $X \geq 0$, $X_q = +$ and $\forall i \in [n] : X_{s_i} X_{t_i} = 0$.
(MV1) A circuit $Z \in \mathcal{C}$ which is sign-reversing.
(MV2) A complementary set $B \subset E_{2n}$ of size n which is not a basis of $\widehat{\mathcal{M}}$.
(MV3) Two distinct, complementary circuits $X, Y \in \mathcal{C}$ with $X_q = Y_q = +$ and $\forall i \in [n] : X_{s_i} Y_{t_i} = X_{t_i} Y_{s_i} = 0$, or $X_{s_i} = Y_{t_i}$ and $X_{t_i} = Y_{s_i}$.

Note that a violation *(MV3)* implies that $\widehat{\mathcal{M}}$ is not a P-matroid extension. Technically, the violation *(MV1)* would be enough to make this search problem total, but our reduction to USO-SF detects only violations of type *(MV2)* and *(MV3)*. Note that as Fearnley et al. [3] already observed, there may be a difference in the complexity of a total search problem depending on the violations chosen. There is no trivial way known to the authors to transform a violation of type *(MV3)* or *(MV2)* to a violation of type *(MV1)*.

With the help of Lemmas 3.2 and 3.3 we now adapt Todd's reduction of non-degenerate P-OMCP instances to USO (recall Section 2.3) to also work with degenerate instances and their respective total search versions.

Given a P-OMCP instance $\widehat{\mathcal{M}} = (\widehat{E_{2n}}, \widehat{\mathcal{C}})$ (note that $\widehat{\mathcal{M}}$ is possibly not a P-matroid extension, or degenerate), we define the orientation $O : V(Q_n) \to \{+,-\}^n$:

$$O(v)_i := \begin{cases} - & \text{if } B(v) \text{ is not a basis,} \\ - & \text{if } v_i = 0 \text{ and } C_{s_i} = 0, \\ + & \text{if } v_i = 1 \text{ and } C_{t_i} = 0, \\ + & \text{if } v_i = 0 \text{ and } C_{s_i} = - \\ & \text{or } v_i = 1 \text{ and } C_{t_i} = -, \\ - & \text{otherwise,} \end{cases}$$

with $B(v)$ and $C := C(B(v), q)$ defined as in Section 2.3. Furthermore, using Lemmas 3.2 and 3.3 we know that O is USO if $\widehat{\mathcal{M}}$ is a P-matroid extension.

▶ **Theorem 5.5.** *The construction above is a polynomial time, promise preserving reduction from P-OMCP to USO-SF.*

The proof can be found in the full version. While the proof requires careful formality, correctness largely follows from the statements Section 3. It follows that P-OMCP as defined in Definition 5.4 is in UniqueEOPL and its promise version is in PromiseUEOPL.

References

1 John Fearnley, Paul W. Goldberg, Alexandros Hollender, and Rahul Savani. The complexity of gradient descent: CLS = PPAD ∩ PLS. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2021, page 46–59, New York, NY, USA, 2021. Association for Computing Machinery. doi:10.1145/3406325.3451052.

2 John Fearnley, Spencer Gordon, Ruta Mehta, and Rahul Savani. Unique end of potential line. *Journal of Computer and System Sciences*, 114:1 – 35, 2020. doi:10.1016/j.jcss.2020.05.007.

3 John Fearnley, Spencer Gordon, Ruta Mehta, and Rahul Savani. Unique end of potential line. *Journal of Computer and System Sciences*, 114:1–35, 2020. doi:10.1016/j.jcss.2020.05.007.

4 Kaspar Fischer and Bernd Gärtner. The smallest enclosing ball of balls: Combinatorial structure and algorithms. *International Journal of Computational Geometry and Applications (IJCGA)*, 14(4–5):341–387, 2004. doi:10.1142/S0218195904001500.

5 Bernd Gärtner and Leo Rüst. Simple stochastic games and p-matrix generalized linear complementarity problems. In Maciej Liśkiewicz and Rüdiger Reischuk, editors, *Fundamentals of Computation Theory*, pages 209–220, Berlin, Heidelberg, 2005. Springer. doi:10.1007/11537311_19.

6 Bernd Gärtner and Ingo Schurr. Linear programming and unique sink orientations. In *Proc. 17th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 749–757, 2006. doi:10.5555/1109557.1109639.

7 Nicolas Grelier. Hardness and approximation of minimum convex partition. In *38th International Symposium on Computational Geometry (SoCG 2022)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2022.

8 Lorenz Klaus. *A fresh look at the complexity of pivoting in linear complementarity.* PhD thesis, ETH Zürich, 2012. doi:10.3929/ethz-a-007604201.

9 Michel Las Vergnas. Extensions ponctuelles d'une géométrie combinatoire orienté. In *Problèmes Combinatoires et Théorie des Graphes*, volume 260, pages 265–270, 1978.

10 Leo Rüst. *The P-matrix linear complementarity problem: generalizations and specializations*. PhD thesis, ETH Zürich, 2007.

11 Ingo Schurr and Tibor Szabó. Finding the sink takes some time: An almost quadratic lower bound for finding the sink of unique sink oriented cubes. *Discrete Comput. Geom.*, 31(4):627–642, 2004. `doi:10.1007/s00454-003-0813-8`.

12 Tibor Szabó and Emo Welzl. Unique sink orientations of cubes. In *Foundations of Computer Science, 2001. Proceedings. 42nd IEEE Symposium on*, pages 547–555. IEEE, 2001. `doi:10.1109/SFCS.2001.959931`.

13 Michael J. Todd. Complementarity in oriented matroids. *SIAM Journal on Algebraic Discrete Methods*, 5(4):467–485, 1984.

14 Simon Weber and Joel Widmer. Realizability makes a difference: A complexity gap for sink-finding in usos, 2022. `doi:10.48550/ARXIV.2207.05985`.

Holes in convex drawings*

Helena Bergold[1], Manfred Scheucher[2], and Felix Schröder[2]

1 **Department of Computer Science, Freie Universität Berlin**
2 **Institut für Mathematik, Technische Universität Berlin**

Abstract

The Erdős–Szekeres theorem states that, for every positive integer k, every sufficiently large point set in general position contains a subset of k points in convex position – a k-gon. In the same vein, Erdős later asked for the existence of k-holes which are k-gons with no additional points in their convex hulls. Today it is known that every sufficiently large point set in general position contains 6-holes, while there exist arbitrarily large point sets without 7-holes.

Harborth (1978) started the investigation of empty triangles in simple drawings of the complete graph. In a simple drawing, vertices are mapped to points in the plane and edges are drawn as simple curves connecting the corresponding endpoints such that any two edges intersect in at most one point, which is either a common vertex or a proper crossing. For the subclass of convex drawings, which in particular includes point sets, Arroyo et al. (2018) showed that quadratically many empty triangles exist.

In this article, we generalize the concept of k-holes to simple drawings of the complete graph K_n and investigate their existence. We provide arbitrarily large simple drawings without 4-holes, show that convex drawings contain quadratically many 4-holes, and generalize the Empty Hexagon theorem (Gerken 2006; Nicolás 2007) by proving the existence of 6-holes in sufficiently large convex drawings.

1 Introduction

The study of holes in point sets was motivated by the Erdős–Szekeres theorem [12] and continues to be an active research branch. The classic theorem states that for every $k \in \mathbb{N}$ every sufficiently large point set in general position (i.e., no three points on a line) contains a subset of k points in convex position – a so called *k-gon*. A variation suggested by Erdős [11] is about the existence of holes. A *k-hole H* in a point set S is a k-gon with the property that there are no points of S in the interior of the convex hull of H.

In this article, we investigate holes in simple drawings of the complete graph K_n. Even though the notation of holes generalizes to simple drawings in a natural manner, we have to introduce some basic notation before we can talk about these structures and our results.

In a *simple drawing*, vertices are mapped to distinct points in the plane (or on the sphere) and edges are mapped to simple curves connecting the corresponding points such that two edges have at most one point in common which is either a common endpoint or a proper intersection. Furthermore we assume that no three edges cross in a common point. Simple drawings can be considered as a generalization of point sets because a set of n points in general position yields a *geometric drawing* of K_n where the vertices are the points and the edges are the straight-line segments connecting the vertices.

Moreover, we investigate the subclass of convex drawings introduced by Arroyo et al. [4]. To define convexity, we consider *triangles* which are subdrawings of K_3 induced by three

* We thank the anonymous reviewers for their valuable comments and Meghana M. Reddy for fruitful discussions. H. Bergold was supported by the DFG-Research Training Group 'Facets of Complexity' (DFG-GRK 2434). M. Scheucher was supported by the DFG Grant SCHE 2214/1-1.

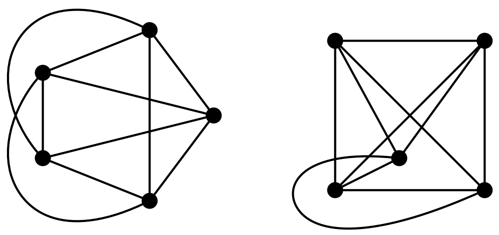


Figure 1 Drawing of the two forbidden subconfigurations of convex drawings. Note that the right drawing is the T_5.

vertices. Since the three edges of a triangle do not cross, the triangle separates the plane (resp. the sphere) into two connected components. The closure of each of the components is called a *side* of the triangle. A side S is *convex*, if for every two vertices in S, the connecting edge is fully contained in S. A simple drawing is *convex* if every triangle has a convex side. Furthermore, a convex drawing is *f-convex* if there is a marking point f in the plane such that for all triangles the side not containing f is convex. A *pseudolinear drawing* is a simple drawing in the plane such that all edges can be extended to pseudolines such that two have at most one point in common. A *pseudoline* is a simple curve partitioning the plane in two unbounded components. As shown by Arroyo et al. [3], a simple drawing of K_n is convex if and only if the two non-convex drawings of the K_5 (see Figure 1) do not appear as a subdrawing. Furthermore, they showed that a simple drawing of K_n is pseudolinear if and only if it is f-convex and the marking point f is in the unbounded cell. For more information about the convexity hierarchy, we refer the reader to [3, 4, 9].

Next, we introduce the notions of k-gons in simple drawings of the complete graph. A *k-gon* C_k is a subdrawing isomorphic to the geometric drawing of k points in convex position, see Figure 2(left). Two simple drawings are *isomorphic* if there exists a bijection on the vertex sets such that the same pairs of edges cross. Note that isomorphism is independent of the choice of the outer cell. Thus, in terms of crossings, a k-gon C_k is a (sub)drawing with vertices $v_1, \dots, v_k$ such that $\{v_i, v_\ell\}$ crosses $\{v_j, v_m\}$ for $i < j < \ell < m$. In contrast to the geometric setting where every sufficiently large geometric drawing contains a k-gon, simple drawings of complete graphs do not necessarily contain k-gons [16]. For example, the perfect twisted drawing T_n depicted in Figure 2(right) does not contain any 5-gon. In terms of crossings, T_n can be characterized as a drawing of K_n with vertices $v_1, \dots, v_n$ such that $\{v_i, v_j\}$ crosses $\{v_\ell, v_m\}$ for $i < j < \ell < m$. However, a theorem by Pach, Solymosi and Tóth [21] states that every sufficiently large drawing of K_n contains a k-gon or a T_k. The currently best known bound is due to Suk and Zeng [23] who showed that every simple drawing of K_n with $n > 2^{9 \cdot \log_2(a) \log_2(b) a^2 b^2}$ contains a C_a or a T_b. Since convex drawings do not contain T_5 as a subdrawing, every convex drawing of K_n with n sufficiently large contains a k-gon.

To eventually define k-holes for general k, let us first consider the special case of 3-holes, which are also known as empty triangles. A triangle is *empty* if one of its two sides does not contain any vertices in its interior. For general simple drawings Harborth [16] proved that there are at least two empty triangles and conjectured that the minimum among all simple drawings on n vertices is $2n - 4$, which is obtained by T_n. García et al. [13] recently showed that the conjecture holds for a class containing the perfect twisted drawings, the so called *generalized twisted drawing*. However, the conjecture remains open in general. The best known lower bound is by Aichholzer et al. [2], who proved that there are at least n empty triangles.

In the geometric setting, the number of empty triangles behaves quite differently: every point set has a quadratic number of empty triangles, and this bound is asymptotically optimal [6]. Moreover, determining the minimum number remains a challenging problem [10, Chapter 8.4]. For the current bounds, see [1]. The class of convex drawings behaves similarly to the geometric setting: the minimum number of empty triangles is asymptotically quadratic as shown by Arroyo et al. [3].

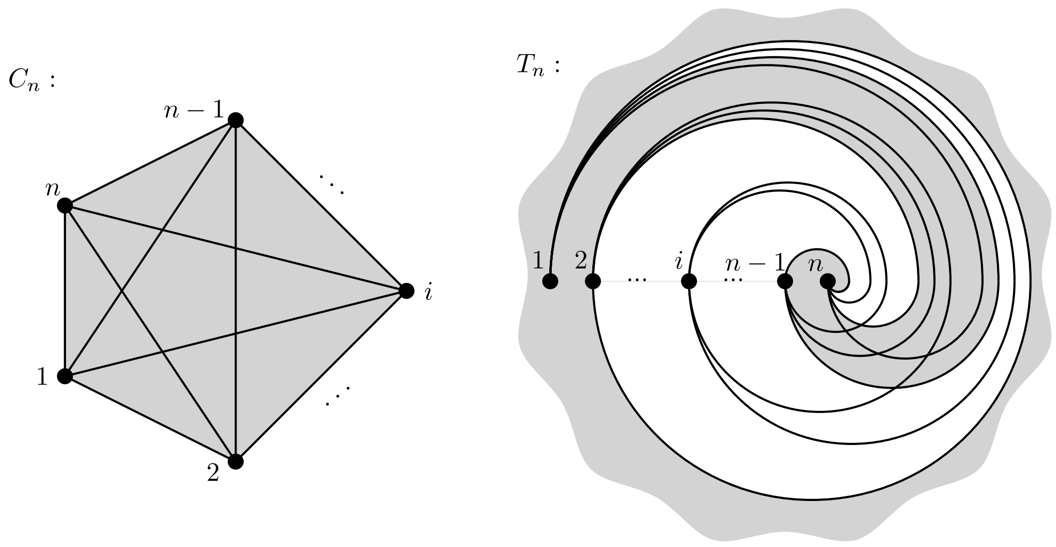


Figure 2 A drawing of C_n (left) and T_n (right) for $n \geq 4$.

In this article, we go beyond empty triangles and investigate the existence of k-holes in simple drawings for $k \geq 4$. In the subdrawing induced by a k-gon with $k \geq 4$, all triangles have exactly one empty side which is the convex side. We define the *convex side* of a k-gon as the union of all convex sides of its triangles and call a vertex which lies in the interior of its convex side an *interior vertex*. The convex side of C_n in Figure 2 is highlighted grey. Note that the triangles of a k-gon for $k \geq 4$ have exactly one convex side, which is the one not containing the other vertices of the k-gon and hence the convex side of a k-gon is well-defined. A *k-hole* is a k-gon which has no interior vertices. For example the vertices $1, 2, n-1, n$ form a 4-hole in T_n which is highlighted grey in Figure 2. For a k-gon C_k in a convex drawing, Arroyo et al. [4] showed that the edges from an interior vertex to a vertex of C_k and edges between two interior vertices are fully contained in the convex side of C_k.

In the geometric setting it is known that for $k \leq 6$ every sufficiently large point set contains a k-hole [15, 14, 19] and that there are arbitrarily large point sets without 7-holes [17]. Since the latter applies to simple drawings, the remaining questions in simple drawings are about the existence of 4-, 5- and 6-holes.

Our Results: For $n \geq 5$ we present a non-convex simple drawing of K_n without 4-holes (Section 2). Furthermore, we show that – as in the geometric setting – the number of 4-holes in convex drawings of K_n is at least $\Omega(n^2)$ (Theorem 3.1), generalizing a result by Bárány and Füredi [6], and that every sufficiently large convex drawing contains a 5-hole and a 6-hole (Theorem 3.2), generalizing the Empty Hexagon theorem by Gerken [14] and Nicolás [19]. In order to show the latter, we prove that if a subdrawing of a convex drawing is induced by a minimal k-gon with $k \geq 5$ together with its interior vertices, then it is f-convex

(Theorem 3.3). This result might be of independent interest as it allows to transfer results from the straight-line, pseudolinear, and f-convex setting to convex drawings.

2 Holes in simple drawings

The perfect twisted drawing T_n depicted in Figure 2(right) has exactly $2n-4$ empty triangles, which are spanned by the vertices $\{1,2,i\}$ for $3 \le i \le n$ and $\{i,n-1,n\}$ for $1 \le i \le n-2$ [16]. For $n \ge 4$, T_n has exactly one 4-hole, which is spanned by $\{1,2,n-1,n\}$.

For $n \ge 5$, let $\widetilde{T_n}$ denote the non-convex drawing of K_n that is obtained by starting with the drawing of T_n and rerouting the edge $\{1,n\}$ as illustrated in Figure 3. More precisely, while in T_n the edge $\{1,n\}$ crosses every edge $\{i,j\}$ with $2 \le i < j \le n-1$, in $\widetilde{T_n}$ it only crosses the edges $\{i,j\}$ with $2 \le i < j \le n-2$. Recall that the pairs of crossing edges determine isomorphism class.

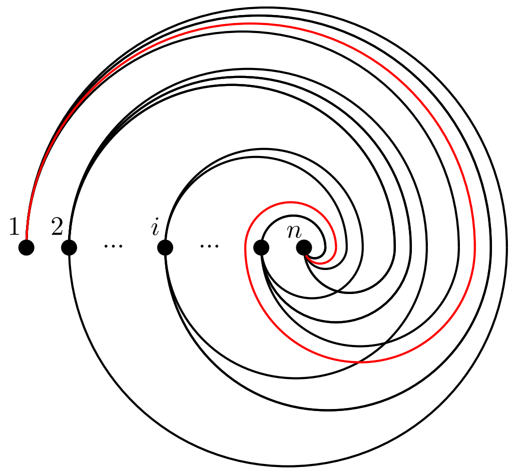


Figure 3 An illustration of the drawing $\widetilde{T_n}$ without 4-holes. The edge $\{1,n\}$ is highlighted red.

▶ Proposition 2.1. For $n \ge 5$ the drawing $\widetilde{T_n}$ does not contain a 4-hole.

Proof. Rerouting the edge $\{1,n\}$ only affects the emptiness of triangles incident to both vertices 1 and n. In particular it only affects the vertex $n-1$ which changes the side of every triangles incident to $\{1,n\}$. In $\widetilde{T_n}$, the only empty triangles incident to $\{1,n\}$ are $\{1,n-1,n\}$ and $\{1,n-2,n\}$. Note that the triangle $\{1,2,n\}$ is not empty in $\widetilde{T_n}$. Hence, the empty triangles in $\widetilde{T_n}$ are $\{1,2,i\}$ for $3 \le i \le n-1$, and $\{i,n-1,n\}$ for $1 \le i \le n-2$, and $\{1,n-2,n\}$. Since no four vertices span four empty triangles, $\widetilde{T_n}$ does not contain a 4-hole. ◀

3 Holes in convex drawings

In this section, we show that convex drawings of the complete graph behave similarly to geometric point sets when it comes to the existence of holes.

▶ **Theorem 3.1.** *Every convex drawing of K_n contains at least $\Omega(n^2)$ 4-holes.*

The proof generalizes the idea of Bárány and Füredi [6] and is deferred to the full version. The bound is asymptotically best possible as there are point sets (squared Horton sets [7] and random point sets [5]) which have only quadratically many 3-holes, 4-holes, 5-holes, and 6-holes.

Furthermore, we investigate larger holes. We show that every sufficiently large convex drawing contains a 6-hole (and hence a 5-hole).

▶ **Theorem 3.2.** *Every convex drawing of K_n with n sufficiently large contains a 6-hole.*

For the proof below we use the existence of a k-gon in sufficiently large simple drawings [21, 23]. Even though the existence of 6-holes directly implies the existence of 5-holes, when adapting the proof to 5-holes one can obtain a better bound on the required number of vertices.

An important part of the proof is that the subdrawing induced by a minimal k-gon together with its interior vertices is f-convex, which then can be transformed into a pseudolinear drawing. A k-gon is *minimal* if its convex side does not contain the convex side of another k-gon.

▶ **Theorem 3.3.** *Let C_k be a minimal k-gon with $n \geq k \geq 5$ in a convex drawing of K_n. Then the subdrawing induced by the vertices on the convex side of C_k is f-convex.*

Since the proof for the existence of 6-holes in point sets [14] also applies to the setting of pseudolinear drawings [22], we can now use Theorem 3.3 to derive Theorem 3.2. Similarly, the text-book proof for the existence of 5-holes in every 6-gon of a point set (see e.g. Section 3.2 in [18]) applies to pseudolinear drawings as it only uses triple orientations. However, proving the existence of 6-holes via 9-gons[1] is far more technical. Hence we refer the interested reader to [22] for a computer-assisted proof and [24] for a simplified proof of the Empty Hexagon theorem with worse bounds.

Proof of Theorem 3.2. By the result of Suk and Zeng [23] every convex drawing of K_n with $n > 2^{225 \log_2(5) \cdot k^2 \log_2(k)}$ contains a k-gon. In order to find a 6-hole, we apply this result for $k = 9$. (To find a 5-hole, we can use $k = 6$.) Consider a minimal k-gon. By Theorem 3.3, the subdrawing induced by the vertices from the convex side of the k-gon is f-convex. Since the existence of holes is invariant under the choice of the outer cell, we can choose the cell containing f as the unbounded cell to make the subdrawing pseudolinear. Next we apply the results concerning the existence of a 6-hole (resp. 5-hole) in pseudolinear drawings and conclude that the subdrawing induced by the k-gon and the interior vertices contains a 6-hole (resp. 5-hole). This 6-hole (resp. 5-hole) in the subdrawing does not contain vertices of the original drawing of K_n since those vertices would be interior vertices of the k-gon. Therefore it is also a 6-hole (resp. 5-hole) in the original drawing. This completes the argument. ◀

4 Discussion

We have shown that every convex drawing of K_n with $n \geq 5$ contain a quadratic number of 4-holes and that sufficiently large drawings contain 5- and 6-holes, while 7-holes do not exist in general. However, it remains to determine the precise values of $h^{\text{conv}}(5)$ and $h^{\text{conv}}(6)$, where $h^{\text{conv}}(k)$ (resp. $h^{\text{geom}}(k)$) denotes the smallest integer such that every convex (resp. geometric) drawing of size $n \geq h^{\text{conv}}(k)$ contains a k-hole. In the geometric setting it is known that $h^{\text{geom}}(5) = 10$ [15] and $30 \leq h^{\text{geom}}(6) \leq 1717$ [14, 20]. In this article we showed

[1] Gerken [14] showed that every 9-gon in a point set yields a 6-hole and Nicolás [19] showed that a 25-gon yields a 6-hole. Both articles involve very long case distinctions.

$h^{\text{conv}}(k) \leq 2^{225\cdot k^2 \cdot \log_2 5 \cdot \log_2 k} + 1$ for $k = 5$ and $k = 6$ (Theorem 3.2). Moreover, we used the SAT framework from [8] to find configurations for $n \leq 10$ and $n = 12$ without 5-holes and to prove that every drawing for $n = 11, 13, 14, 15, 16$ contains a 5-hole. Based on our computational data, we conjecture that $h^{\text{conv}}(5) = 13$. Note that unlike in the geometric setting, the existence of 5-holes in convex drawings is not monotone as the existence of a 5-hole in all convex drawings of K_{11} does not imply the existence for $n = 12$.

It would be interesting to obtain better bounds on the size of a largest k-gon and on the size of a largest f-convex subdrawing in a convex drawing of K_n. The currently best estimate for a k-gon is by Suk and Zeng [23], which yields $\Omega((\log n)^{1/2-o(1)})$, and combining this with Theorem 3.3 yields an f-convex drawing of the same size.

References

1 Oswin Aichholzer, Martin Balko, Thomas Hackl, Jan Kynčl, Irene Parada, Manfred Scheucher, Pavel Valtr, and Birgit Vogtenhuber. A superlinear lower bound on the number of 5-holes. *Journal of Combinatorial Theory, Series A*, 173:105236, 2020. doi:10.1016/j.jcta.2020.105236.

2 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*, 31:335–345, 2015. doi:10.1007/s00373-015-1550-5.

3 Alan Arroyo, Dan McQuillan, R. Bruce Richter, and Gelasio Salazar. Levi's Lemma, pseudolinear drawings of K_n, and empty triangles. *Journal of Graph Theory*, 87(4):443–459, 2018. doi:10.1002/jgt.22167.

4 Alan Arroyo, Dan McQuillan, R. Bruce Richter, and Gelasio Salazar. Convex drawings of the complete graph: topology meets geometry. *Ars Mathematica Contemporanea*, 22(3), 2022. doi:10.26493/1855-3974.2134.ac9.

5 Martin Balko, Manfred Scheucher, and Pavel Valtr. Tight bounds on the expected number of holes in random point sets. *Random Structures & Algorithms*, 62(1):29–51, 2023. doi:10.1002/rsa.21088.

6 Imre Bárány and Zoltán Füredi. Empty simplices in Euclidean space. *Canadian Mathematical Bulletin*, 30(4):436–445, 1987. doi:10.4153/cmb-1987-064-1.

7 Imre Bárány and Pavel Valtr. Planar point sets with a small number of empty convex polygons. *Studia Scientiarum Mathematicarum Hungarica*, 41(2):243–266, 2004. doi:10.1556/sscmath.41.2004.2.4.

8 Helena Bergold, Stefan Felsner, Meghana M. Reddy, and Manfred Scheucher. Using SAT to study plane Hamiltonian substructures in simple drawings. In Proc. of EuroCG'23, 2023.

9 Helena Bergold, Stefan Felsner, Manfred Scheucher, Felix Schröder, and Raphael Steiner. Topological Drawings meet Classical Theorems from Convex Geometry. *Discrete & Computational Geometry*, 2022. doi:10.1007/s00454-022-00408-6.

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

11 Paul Erdős. Some problems on elementary geometry. *Australian Mathematical Society*, 2:2–3, 1978. URL: https://users.renyi.hu/~p_erdos/1978-44.pdf.

12 Paul Erdős and George Szekeres. A combinatorial problem in geometry. *Compositio Mathematica*, 2:463–470, 1935. URL: http://www.renyi.hu/~p_erdos/1935-01.pdf.

13 Alfredo García, Javier Tejel, Birgit Vogtenhuber, and Alexandra Weinberger. Empty triangles in generalized twisted drawings of K_n. In *Graph Drawing and Network Visualization*, volume 13764 of *LNCS*, pages 40–48. Springer, 2022. doi:10.1007/978-3-031-22203-0_4.

14 Tobias Gerken. Empty Convex Hexagons in Planar Point Sets. *Discrete & Computational Geometry*, 39(1):239–272, 2008. doi:10.1007/s00454-007-9018-x.

15 Heiko Harborth. Konvexe Fünfecke in ebenen Punktmengen. *Elemente der Mathematik*, 33:116–118, 1978. URL: `http://www.digizeitschriften.de/dms/img/?PID=GDZPPN002079801`.

16 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`.

17 Joseph D. Horton. Sets with no empty convex 7-gons. *Canadian Mathematical Bulletin*, 26:482–484, 1983. `doi:10.4153/CMB-1983-077-8`.

18 Jiří Matoušek. *Lectures on Discrete Geometry*. Springer, 2002. `doi:10.1007/978-1-4613-0039-7`.

19 Carlos M. Nicolás. The Empty Hexagon Theorem. *Discrete & Computational Geometry*, 38(2):389–397, 2007. `doi:10.1007/s00454-007-1343-6`.

20 Mark H. Overmars. Finding sets of points without empty convex 6-gons. *Discrete & Computational Geometry*, 29(1):153–158, 2003. `doi:10.1007/s00454-002-2829-x`.

21 János Pach, József Solymosi, and Géza Tóth. Unavoidable configurations in complete topological graphs. *Discrete & Computational Geometry*, 30(2):311–320, 2003. `doi:10.1007/s00454-003-0012-9`.

22 Manfred Scheucher. A SAT attack on higher dimensional Erdős–Szekeres numbers, 2022. `doi:10.48550/arxiv.2105.08406`.

23 Andrew Suk and Ji Zeng. Unavoidable patterns in complete simple topological graphs. In *Graph Drawing and Network Visualization*, volume 13764 of *LNCS*, pages 3–15. Springer, 2022. `doi:10.1007/978-3-031-22203-0_1`.

24 Pavel Valtr. On Empty Hexagons. In *Surveys on Discrete and Computational Geometry: Twenty Years Later*, volume 453 of *Contemporary Mathematics*, pages 433–441. AMS, 2008. `doi:10.1090/conm/453/08809`.

Maximum overlap area of a convex polyhedron and a convex polygon under translation

Honglin Zhu[1] **and Hyuk Jun Kweon**[2]

1 **Massachusetts Institute of Technology**
`honglinz@mit.edu`

2 **Massachusetts Institute of Technology**
`kweon@mit.edu`

Abstract

Let P be a convex polyhedron and Q be a convex polygon with n vertices in total in three-dimensional space. We present a deterministic algorithm that finds a translation vector $v \in \mathbb{R}^3$ maximizing the overlap area $|P \cap (Q+v)|$ in $O(n \log^2 n)$ time. We then apply our algorithm to solve two related problems. We give an $O(n \log^3 n)$ time algorithm that finds the maximum overlap area of three convex polygons with n vertices in total. We also give an $O(n \log^2 n)$ time algorithm that minimizes the symmetric difference of two convex polygons under scaling and translation.

Related Version A full version of the paper is available at `https://arxiv.org/abs/2301.02949`.

1 Introduction

Shape matching is an important topic in computational geometry, with useful applications in areas such as computer graphics. In a typical problem of shape matching, we are supplied two or more shapes, and we want to determine how much the shapes resemble each other. More precisely, given a similarity measure and a set of allowed transformations, we want to transform the shapes to maximize their similarity measure.

While there are many candidates for the similarity measure, we focus on the area/volume of overlap or of symmetric difference. The advantage to these is that they are robust against noise on the boundary of the images [6]. For previous results in this area, see [6, 1, 3, 4, 2, 12].

While many have studied the matching problem for two convex polytopes of the same dimension, few have examined the problem for polytopes of different dimensions or matching more than two polytopes. In this paper, we present a deterministic algorithm for finding the maximum overlap area of a convex polyhedron and a convex polygon under translation in three-dimensional space. Using this algorithm, we also solve the problems of maximizing the overlap of three convex polygons and of minimizing the symmetric difference of two convex polygons under homothety.

2 Preliminaries

Let $P \subset \mathbb{R}^3$ be a convex polyhedron and $Q \subset \mathbb{R}^2$ be a convex polygon with n vertices in total. Throughout the paper, we assume that Q is in the xy-plane, and that the lowest point of P is on the xy-plane. We want to find a translation vector $v = (x, y, z) \in \mathbb{R}^3$ that maximizes the overlap area $f(v) = |P \cap (Q+v)|$.

It is easy to observe that $f(v)$ is continuous and piecewise quadratic on the interior of its support. As noted in [6, 1, 3], f is smooth on a region R if $P \cap (Q+v)$ is combinatorially equivalent for all $v \in R$, that is, if we have the same set of face-edge incidences between P and Q. Following the convention of [1], we call the polygons that form the boundaries of these regions the *event polygons*, and as in [6], we call the space of translations of Q the *configuration space*. The arrangement of the event polygons partition the configuration space into cells with disjoint interiors. The overlap function $f(v)$ is quadratic on each cell. Thus, to locate a translation maximizing f, we need to characterize the event polygons.

For two sets $A, B \subset \mathbb{R}^d$, we write the *Minkowski sum* of A and B as $A + B := \{a + b | a \in A, b \in B\}$. We also write $A - B$ for the Minkowski sum of A with $-B = \{-b | b \in B\}$. We categorize the event polygons into three types and describe them in terms of Minkowski sums:

(I) When $Q + v$ contains a vertex of P. For each vertex u of P, we have an event polygon $u - Q$. There are $O(n)$ event polygons of this type.
(II) When a vertex of $Q + v$ is contained in a face of P. For each face F of P and each vertex v of Q, we have an event polygon $F - v$. There are $O(n^2)$ event polygons of this type.
(III) When an edge of $Q + v$ intersects an edge of P. For each edge e of P and each edge e' of Q, we have an event polygon $e - e'$. There are $O(n^2)$ event polygons of this type.

The reason that convexity is fundamental is due to the following standard fact, as noted and proved in [6, 12].

▶ **Proposition 2.1.** *Let P be a d'-dimensional convex polytope and let Q be a d-dimensional convex polytope. Suppose $d' \geq d$. Let $f(v) = \mathrm{Vol}(P \cap (Q + v))$ be the volume of the overlap function. Then, $f(v)^{1/d}$ is concave on its support* $\mathrm{supp}(f) = \{v | f(v) > 0\}$.

As in [5], we say a function $f : \mathbb{R} \to \mathbb{R}$ is *unimodal* (resp. *strictly unimodal*) if it increases (resp. strictly increases) to a maximum value, possibly stays there for some interval, and then decreases (resp. strictly decreases). Furthermore, we say a function $f : \mathbb{R}^d \to \mathbb{R}$ is unimodal (resp. strictly unimodal) if its restriction to any line is unimodal (resp. strictly unimodal).

▶ **Corollary 2.2** ([5]). *For any line l parameterized by $l = p + vt$ in $\mathbb{R}^{d'}$, the function $f_l(t) = f(p + vt)$ is strictly unimodal.*

We introduce a divide-and-conquer technique that we apply in our algorithm.

▶ **Lemma 2.3** ([8]). *Given n hyperplanes in $\mathbb{R}^d$ and a region $R \subset \mathbb{R}^d$, a $(1/r)$-cutting is a collection of simplices with disjoint interiors, which together cover R and such that the interior of each simplex intersects at most n/r hyperplanes. A $(1/r)$-cutting of size $O(r^d)$ can be computed deterministically in $O(nr^{d-1})$ time. In addition, the set of hyperplanes intersecting each simplex of the cutting is reported in the same time.*

Another technique that we use in our algorithm is a generalization of Megiddo's prune-and-search [11]. This technique is of independent interest and can likely be applied to other problems.

▶ **Theorem 2.4.** *Let $S = \bigcup_{i=1}^{n} S_i$ be a union of n sets of $O(m)$ parallel lines in the plane, none of which are parallel to the x-axis, and suppose the lines in each S_i are indexed from left to right.*

Suppose there is an unknown point $p^* \in \mathbb{R}^2$ and we are given an oracle that decides in time T the relative position of p^* to any line in the plane. Then we can find the relative position of p^* to every line in S in $O(n \log^2 m + (T + n) \log(mn))$ time.

3 Maximum overlap of convex polyhedron and convex polygon

In this section, we prove our main result:

▶ **Theorem 3.1.** *Let P be a convex polyhedron and Q a convex polygon with n vertices in total. We can find a vector $v \in \mathbb{R}^3$ that maximizes the overlap area $|P \cap (Q+v)|$ in $O(n \log^2 n)$ time.*

Following the convention in [6], we call a translation that maximizes the overlap function f a *goal placement.* In the algorithm, we keep track of a closed *target region* R which we know contains a goal placement and decrease its size until for each event polygon F, either $F \cap \text{interior}(R) = \varnothing$ or $F \supset R$. Then, f is quadratic on R and we can find the maximum of f on R using standard calculus. Thus, the goal of our algorithm is to efficiently trim R to eliminate event polygons that intersect it.

In the beginning of the algorithm, the target region is the interior of the Minkowski sum $P - Q$, where the overlap function is positive. By the unimodality of the overlap function, the set of goal placements is convex. Thus, for a plane in the configuration space, either it contains a goal placement, or all goal placements lie on one of the two open half spaces separated by the plane. If we have a way of knowing which case it is for any plane, we can decrease the size of our target region by cutting it with planes and finding the piece to recurse. More precisely, we need a subroutine **PlaneDecision** that decides the relative position of the set of goal placements to a plane S.

Whenever **PlaneDecision** reports that a goal placement is found on a plane, we can let the algorithm terminate. Thus, we can assume it always reports a half-space containing a goal placement.

As in Algorithm 1, we break down our algorithm into three stages.

Algorithm 1: Pseudocode for Theorem 3.1

```
input  : A convex polyhedron $P \in \mathbb{R}^3$ and a convex polygon $Q \in \mathbb{R}^3$ with $n$ vertices
         in total
output : A translation $v \in \mathbb{R}^3$ maximizing the area $|P \cap (Q + v)|$
1 Locate a horizontal slice containing a goal placement that does not contain any
    vertices of $P$ and replace $P$ by this slice of $P$
2 Find a "tube" $D + l_y$ whose interior contains a goal placement and intersects $O(n)$
    event polygons, where $D$ is a triangle in the $xz$-plane and $l_y$ is the $y$-axis
3 Recursively construct a $(1/2)$-cutting of the target region $D + l_y$ to find a simplex
    containing a goal placement that does not intersect any event polygon
```

3.1 Stage 1

In the first stage of our algorithm, we make use of [6] to simplify our problem so that P can be taken as a convex polyhedron with all of its vertices on two horizontal planes.

We sort the vertices of P by z-coordinate in increasing order and sort the vertices of Q in counterclockwise order. Next, we trim the target region with horizontal planes (planes parallel to the xy-plane) to get to a slice that does not contain any vertices of P.

▶ **Lemma 3.2.** *In $O(n \log^2 n)$ time, we can locate a strip $R = \{(x, y, z) | z \in [z_0, z_1]\}$ whose interior contains a goal placement and P has no vertices with $z \in [z_0, z_1]$.*

By Chazelle's algorithm [7], the convex polyhedron $P' = \{(x, y, z) \in P | z \in [z_0, z_1]\}$ can be computed in $O(n)$ time. From now on, we replace P with P' (see Figure 1). Without loss of generality, assume $z_0 = 0$ and $z_1 = 1$.

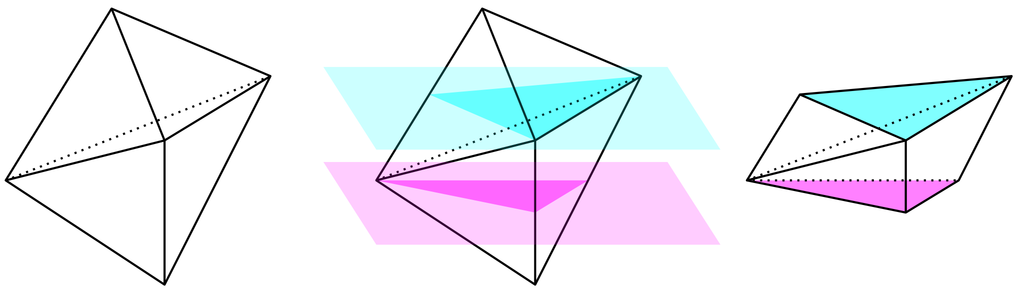


Figure 1 The slice of P with $z \in [z_0, z_1]$.

The region in the configuration space where $|P \cap (Q+v)| > 0$ is the Minkowski sum $P-Q$. Since P only has two levels $P_0 = \{(x,y,z) \in P | z = 0\}$ and $P_1 = \{(x,y,z) \in P | z = 1\}$ that contain vertices, the Minkowski sum $P-Q$ is simply the convex hull of $(P_0 - Q) \cup (P_1 - Q)$, which has $O(n)$ vertices. We can compute $P_0 - Q$ and $P_1 - Q$ in $O(n)$ time and compute their convex hull in $O(n \log n)$ time by Chazelle's algorithm [9].

3.2 PlaneDecision

Due to space constraints, we will not present the algorithm **PlaneDecision**. For details, see the full version of the paper [10].

We present a perturbation method to reduce the problem of deciding the relative position of the goal placement to an arbitrary plane to finding the maximum of the overlap over an arbitrary plane.

▶ **Lemma 3.3.** *Suppose we can compute $\max_{v \in S} f(v)$ for any plane $S \subset \mathbb{R}^3$ in time T, then we can perform **PlaneDecision** for any plane in time $O(T)$.*

Proof. The idea is to compute $\max_{v \in S'} f(v)$ for certain S' that are perturbed slightly from S to see in which direction relative to S does f increase.

We compute over an extension of the reals $\mathbb{R}[\omega]/(\omega^3)$, where $\omega > 0$ is smaller than any real number. Let $A > 0$ be the maximum of f over a plane S. Let S_+ and S_- be the two planes parallel to S that have distance ω from S. We compute $A_+ = \max_{v \in S_+} f(v)$ and $A_- = \max_{v \in S_-} f(v)$ in $O(T)$ time. Since f is piecewise quadratic, A_+ and A_- as symbolic expression will only involve quadratic terms in ω. Since f is strictly unimodal on $P-Q$, there are three possibilities:

1. If $A_+ > A$, then halfspace on the side of S_+ contains the set of goal placements.
2. If $A_- > A$, then halfspace on the side of S_- contains the set of goal placements.
3. If $A \geq A_+$ and $A \geq A_-$, then A is the global maximum of f.

Thus, in $O(T)$ time, we can finish **PlaneDecision**. ◀

With Theorem 3.3, it suffices for us to give an algorithm finding the maximum of the overlap over any plane. Just like in our main algorithm, we want to prune the configuration space (now restricted to a plane), until we locate a region that contains the maximum and that does not intersect any event polygon. In Algorithm 2, we give an outline for **PlaneDecision**:

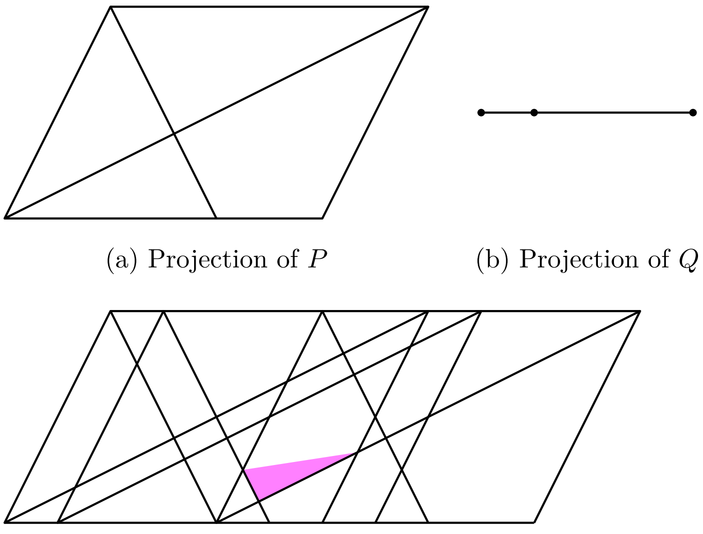
▶ **Proposition 3.4.** *For a plane S, we can perform **PlaneDecision** in $O(n \log n)$ time.*

Algorithm 2: Pseudocode for **PlaneDecision**

```
  input : A plane $S \subset \mathbb{R}^3$
  output: A translation $v \in S$ maximizing the area $|P \cap (Q+v)|$
1 Compute $S \cap (P-Q)$ and set it to be our initial target region.
2 Locate a strip on $S$ containing a good placement whose interior intersects $O(n)$ event
    polygons.
3 Recursively construct a (1/2)-cutting of the strip to find a triangle containing a good
    placement that does not intersect any event polygon
```

3.3 Stage 2

With the general **PlaneDecision** at our disposal, we now move on to Stage 2, the main component of our algorithm. We project the entire configuration space and the event polygons onto the xz-plane in order to find a target region D whose preimage $D + l_y$ intersects few event polygons, where l_y is the y-axis (see Figure 2).


(a) Projection of P (b) Projection of Q

(c) Projection of the configuration space, and the target region D

Figure 2 Projecting onto the xz-plane.

The non-horizontal edges of the event polygons project to segments on the strip $0 < z < 1$ on the xz-plane. We characterize our desired region D in the following lemma.

▶ **Lemma 3.5.** *For a region D that does not intersect any of the segments that are the projections of the non-horizontal edges of the event polygons, the preimage $D + l_y$ intersects $O(n)$ event polygons.*

Now it remains to efficiently find such a region D with $D + l_y$ containing a goal placement and compute the $O(n)$ event polygons that intersect its interior.

▶ **Lemma 3.6.** *In $O(n \log^2 n)$ time, we can find a triangle D in the xz-plane such that the interior of $D + l_y$ contains a goal placement and intersects $O(n)$ event polygons. We can compute these $O(n)$ event polygons in the same time bound.*

3.4 Stage 3

Now, we have a target region $R = D + l_y$ whose interior contains a goal placement, and we have the $O(n)$ event polygons that intersect it.

▶ **Lemma 3.7.** *In $O(n \log^2 n)$ time, we can find a region $R' \subset R$ containing a goal placement that does not intersect any of the $O(n)$ event polygons.*

Finally, since the overlap function is quadratic on our final region R', we can solve for the maximum using standard calculus. This concludes the proof of Theorem 3.1.

4 Applications

We present two applications of Theorem 3.1 to other problems in computational geometry. First, we give a deterministic algorithm for maximizing the overlap of three convex polygons.

▶ **Theorem 4.1.** *Let P, Q, R be three convex polygons with n vertices in total in the plane. We can find a pair of translations $(v_Q, v_R) \in \mathbb{R}^4$ that maximizes the overlap area $|P \cap (Q + v_Q) \cap (R + v_R)|$ in $O(n \log^3 n)$ time.*

In this problem, the configuration space is four-dimensional. An easy extension of Proposition 2.1 and Theorem 2.2 shows that the function of overlap area is again unimodal. This time, we have four-dimensional *event polyhedra* instead of event polygons that divide the configuration space into four-dimensional cells on which $g(v_Q, v_R)$ is quadratic. We call a hyperplane containing an event polyhedron an *event hyperplane*, and they are defined by two types of events:

(I) When one vertex of P, $Q + v_Q$ or $R + v_R$ lies on an edge of another polygon. There are $O(n)$ groups of $O(n)$ parallel event hyperplanes of this type.
(II) When an edge from each of the three polygons intersect at one point. There are $O(n^3)$ event hyperplanes of this type.

To overcome the difficulty of dealing with the $O(n^3)$ event hyperplanes of type (II), we first prune the configuration space to a region intersecting no event hyperplanes of type (I). We then show that the resulting region only intersects $O(n)$ event hyperplanes of type (II), at which point we can use Theorem 2.3 iteratively to finish.

Observe that maximizing the overlap over a hyperplane in which two polygons move relatively in a line and the other polygon moves freely corresponds precisely to the problem of overlapping a convex polyhedron (whose cross-sections are the intersections of the two polygons moving in a line) and a convex polygon (the third polygon). Thus, Theorem 3.1 gives us a kind of "**HyperplaneDecision**" which we can use to prune the event hyperplanes of type (I).

We also give a deterministic $O(n \log^2 n)$ time algorithm for minimizing the symmetric difference of two convex polygons under homothety (a scaling and a translation), which is an improvement to Yon et al.'s $O(n \log^3 n)$ time algorithm [12].

▶ **Theorem 4.2.** *Let P and Q be convex polygons with n vertices in total. Then we can find a homothety φ that minimizes the area of symmetric difference $|P \setminus \varphi(Q)| + |\varphi(Q) \setminus P|$ in $O(n \log^2 n)$ time.*

We want to minimize the function $h(\varphi) = h(x, y, \lambda) = |P \setminus \varphi(Q)| + |\varphi(Q) \setminus P|$, where $\varphi(Q) = \lambda Q + (x, y)$. We can rewrite this as $h(\varphi) = |P| + |Q|\lambda^2 - 2|P \cap \varphi(Q)|$. Thus, minimizing h is the same as maximizing the function $f(\varphi) = |P \cap \varphi(Q)| - c\lambda^2$, where $c = \frac{1}{2}|Q|$.

Consider the cone $C = \{(x, y, \lambda)|\lambda \in [0, M], (x, y) \in \lambda Q\}$, where $M = \sqrt{|P|/c}$ (see Figure 3). Then f is negative for $\lambda > M$ so it is never maximized. We also put P into $\mathbb{R}^3$ by $P = \{(x, y, 0)|(x, y) \in P\}$. Since $f(x, y, \lambda) = |C \cap (P + (-x, -y, \lambda))| - c\lambda^2$, the problem reduces to maximizing the overlap area of the cone C and P under translation subtracted by a quadratic function. Some modification of Theorem 3.1 gives Theorem 4.2.

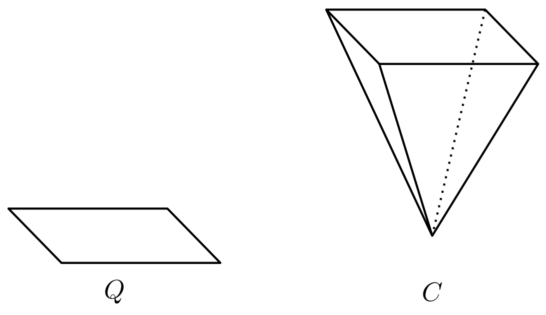


Figure 3 Formation of the cone C.

References

1 Hee-Kap Ahn, Peter Brass, and Chan-Su Shin. Maximum overlap and minimum convex hull of two convex polyhedra under translations. *Comput. Geom.*, 40(2):171–177, 2008. doi:10.1016/j.comgeo.2007.08.001.

2 Hee-Kap Ahn, Siu-Wing Cheng, Hyuk Jun Kweon, and Juyoung Yon. Overlap of convex polytopes under rigid motion. *Comput. Geom.*, 47(1):15–24, 2014. doi:10.1016/j.comgeo.2013.08.001.

3 Hee-Kap Ahn, Siu-Wing Cheng, and Iris Reinbacher. Maximum overlap of convex polytopes under translation. *Comput. Geom.*, 46(5):552–565, 2013. doi:10.1016/j.comgeo.2011.11.003.

4 Hee-Kap Ahn, Otfried Cheong, Chong-Dae Park, Chan-Su Shin, and Antoine Vigneron. Maximizing the overlap of two planar convex sets under rigid motions. *Comput. Geom.*, 37(1):3–15, 2007. doi:10.1016/j.comgeo.2006.01.005.

5 David Avis, Prosenjit Bose, Thomas C. Shermer, Jack Snoeyink, Godfried Toussaint, and Binhai Zhu. On the sectional area of convex polytopes. In *Communication at the 12th Annu. ACM Sympos. Comput. Geom.*, page C. Association for Computing Machinery, New York, NY, 1996.

6 Mark de Berg, Olivier Devillers, Marc van Kreveld, Otfried Schwarzkopf, and Monique Teillaud. Computing the maximum overlap of two convex polygons under translations. In *International Symposium on Algorithms and Computation*, pages 126–135. Springer, 1996.

7 Bernard Chazelle. An optimal algorithm for intersecting three-dimensional convex polyhedra. *SIAM J. Comput.*, 21(4):671–696, 1992. doi:10.1137/0221041.

8 Bernard Chazelle. Cutting hyperplanes for divide-and-conquer. *Discrete Comput. Geom.*, 9(2):145–158, 1993. doi:10.1007/BF02189314.

9 Bernard Chazelle. An optimal convex hull algorithm in any fixed dimension. *Discrete Comput. Geom.*, 10(4):377–409, 1993. doi:10.1007/BF02573985.

10 Hyuk Jun Kweon and Honglin Zhu. Maximum overlap area of a convex polyhedron and a convex polygon under translation, 2023. URL: https://arxiv.org/abs/2301.02949, doi:10.48550/ARXIV.2301.02949.

11 Nimrod Megiddo. Linear programming in linear time when the dimension is fixed. *J. Assoc. Comput. Mach.*, 31(1):114–127, 1984. `doi:10.1145/2422.322418`.

12 Juyoung Yon, Sang Won Bae, Siu-Wing Cheng, Otfried Cheong, and Bryan T. Wilkinson. Approximating convex shapes with respect to symmetric difference under homotheties. In *32nd International Symposium on Computational Geometry*, volume 51 of *LIPIcs. Leibniz Int. Proc. Inform.*, pages Art. No. 63, 15. Schloss Dagstuhl. Leibniz-Zent. Inform., Wadern, 2016.

Extending Orthogonal Planar Graph Drawings is Fixed-Parameter Tractable

Sujoy Bhore[1], Robert Ganian[2], Liana Khazaliya[2], Fabrizio Montecchiani[3], and Martin Nöllenburg[2]

1 **Indian Institute of Technology Bombay, India**
`sujoy.bhore@gmail.com`

2 **Technische Universität Wien, Austria**
`{rganian,lkhazaliya,noellenburg}@ac.tuwien.ac.at`

3 **University of Perugia**
`fabrizio.montecchiani@unipg.it`

Abstract

We consider the extension problem for bend-minimal orthogonal drawings of planar graphs, which is among the most fundamental geometric graph drawing representations. While the problem was known to be NP-hard, it is natural to consider the case where the drawn part is connected and only a small part of the graph is still to be drawn. Here, we prove the problem is in FPT when parameterized by the size of the missing subgraph.

Related Version `http://arxiv.org/abs/2302.10046`

1 Introduction

Drawing extension problems are motivated, for instance, by visualizing networks, in which certain subgraphs represent important motifs that require a specific drawing, or by visualizing dynamic networks, in which new edges and vertices must be integrated in an existing, stable drawing. Generally speaking, we are given a graph G and a (typically connected) subgraph H of G with a drawing $\Gamma(H)$, which is called a *partial* drawing of G. The drawing $\Gamma(H)$ satisfies certain topological or geometric properties, e.g., planarity, upward planarity, or 1-planarity, and the goal of the corresponding extension problem is to extend $\Gamma(H)$ to a drawing $\Gamma(G)$ of the whole graph G (if possible) by inserting the missing vertices and edges into $\Gamma(H)$ while maintaining the required drawing properties.

In this paper, we study the geometric drawing extension problem arising in the context of one of the most fundamental graph drawing styles: *orthogonal drawings* [3, 4, 6, 10]. In a planar orthogonal drawing, edges are represented as polylines comprised of (one or more) horizontal and vertical segments, ideally with as few overall bends as possible, where edges are not allowed to intersect except at common endpoints. Orthogonal drawings find applications in various domains from VLSI and printed circuit board (PCB) design, to schematic network visualizations, e.g., UML diagrams in software engineering, argument maps, or flow charts.

Given the above, a key optimization goal in orthogonal drawings is bend minimization. This task is known to be NP-hard [8] when optimizing over all possible combinatorial embeddings of a given graph, but can be solved in polynomial time for a fixed combinatorial embedding using the network flow model of Tamassia [11].

Despite the general popularity of planar orthogonal graph drawings, the corresponding extension problem has only been considered recently by Angelini et al. [1]. While they showed that the existence of a planar orthogonal extension can be decided in linear time,

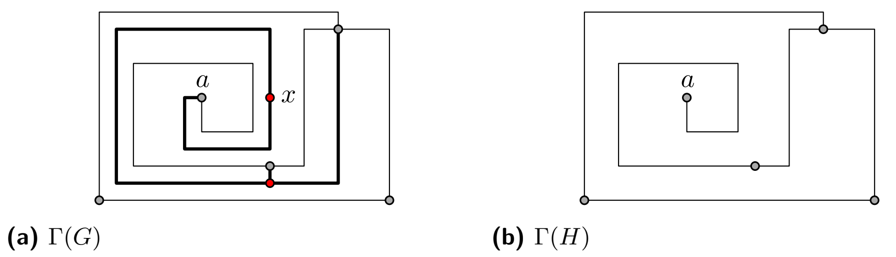


Figure 1 An orthogonal drawing of (a) a graph G and (b) a subgraph H of G.

the orthogonal bend-minimal drawing extension problem in general is easily seen to be NP-hard as it generalizes the case in which the pre-drawn part of the graph is empty [8]. Our paper addresses the parameterized complexity of the bend-minimal extension problem for planar orthogonal graph drawings under the most natural parameterization of the problem, which is the size of the subgraph that is still missing from the drawing.

Problem Statement. Let G be a planar graph and H be a connected subgraph of G. We call the complement $X = V(G) \setminus V(H)$ the *missing vertex set* of G, and $E_X = E(G) \setminus E(H)$ the *missing edge set*. Let $\Gamma(H)$ be a planar orthogonal drawing of H. A planar orthogonal drawing $\Gamma(G)$ *extends* $\Gamma(H)$ if its restriction to the vertices and edges of H coincides with $\Gamma(H)$. Moreover, $\Gamma(G)$ is a β-extension of $\Gamma(H)$ if it extends $\Gamma(H)$ and the total number of bends along the edges of E_X is at most β, for some $\beta \in \mathbb{N}$. For example, Figure 1a shows a 7-extension $\Gamma(G)$ of the drawing $\Gamma(H)$ in Figure 1b, with the missing vertices drawn in red.

BEND-MINIMAL ORTHOGONAL EXTENSION (BMOE)
Input: $(G, H, \Gamma(H))$, integer β
Problem: Is there a β-extension $\Gamma(G)$ of $\Gamma(H)$?

Our parameter of interest, denoted by κ, is the number of vertices and edges missing from H, i.e., $\kappa = |V(G) \setminus V(H)| + |E(G) \setminus E(H)|$.

Contributions and overview. We establish the fixed-parameter tractability of BMOE when parameterized by κ. While there have been numerous recent advances in the parameterized study of drawing extension problems [5,7,9], the specific drawing styles considered in those papers were primarily topological in nature, while for bend minimization the geometry of the instance is crucial. In order to overcome this difficulty, we develop a new set of tools summarized below. We first apply an initial branching step to simplify the problem (Section 2). This step allows us to reduce our target problem to BEND-MINIMAL ORTHOGONAL EXTENSION ON A FACE (F-BMOE), where the missing edges and vertices are drawn only in a marked face f and we have some additional information about how the edges are geometrically connected. Next, we focus on solving an instance of F-BMOE (Section 3). We show that certain parts of the marked face f are irrelevant and can be pruned away, and also use an involved argument to reduce the case of f being the outer face to the case of f being an inner face. Once that is done, we enter the centerpiece of our approach (Section 4), where the aim is to obtain a suitable discretization of our instance. To this end, we split the face f into so-called *sectors*, which group together points that have the same "bend distances" to all of the connecting points on the boundary of f. Furthermore, we construct a *sector-grid*—a point-set such that each sector contains a bounded number of points from this set, and every bend-minimal extension can be modified to only use points from this set for all vertices and bends. While this latter result would make it easy to handle each individual sector by brute force, the issue is that the number of sectors can be very large, hindering tractability. To deal with this obstacle, we capture the connections between sectors via a *sector graph* whose vertices are the sectors and whose edges represent geometric adjacencies between sectors. Crucially, we show that the sector graph has treewidth bounded by a function of κ. Having obtained this bound on the treewidth, the last step simply combines the already constructed sector grid with dynamic programming to solve F-BMOE (and hence also BMOE).

Many technicalities and proofs have been omitted; see [2] for the full paper.

2 Initial Branching

Let $\langle(G, H, \Gamma(H)), \beta\rangle$ be an instance of BMOE. A vertex $w \in V(H)$ is called an *anchor* if it is incident to an edge in E_X. For a missing edge $vw \in E_X$ incident to a vertex $v \in V(H)$, we will use "ports" to specify a direction that vw could potentially use to reach v in an extension of $\Gamma(H)$; we denote these directions as d, which is an element from $\{\downarrow$ (north), $\uparrow$ (south), $\leftarrow$ (east), $\rightarrow$ (west)$\}$. Formally, a *port candidate* for $vw \in E_X$ and $v \in V(H)$ is a pair (v, d). A *port-function* is an ordered set of port candidates which contains precisely one port candidate for each $vw \in E_X, v \in V(H)$, ordered lexicographically by v and then by w.

Bend-Minimal Orthogonal Extension on a Face (F-BMOE)
Input: Planar graph G_f; induced subgraph H_f of G_f with $k = |X_f|$, where $X_f = V(G_f) \setminus V(H_f)$; drawing $\Gamma(H_f)$ of H_f consisting of a single inner face f; port-function $\mathcal{P}$.
Task: Compute the minimum β for which a β-extension of $\Gamma(H_f)$ exists s.t. (1) all missing edges and vertices are drawn in face f, (2) each edge $xa \in E_X$ where $a \in V(H)$ connects to a via its port candidate defined by $\mathcal{P}$, or determine that no such extension exists.

▶ **Lemma 2.1.** *There is an algorithm that solves an instance $\mathcal{I}$ of* BMOE *in time $3^{\mathcal{O}(\kappa)} \cdot T(|\mathcal{I}|, k)$, where $T(a, b)$ is the time required to solve an instance of* F-BMOE *with instance size a and parameter value b.*

The algorithm in [1] can be used to test whether an instance of F-BMOE admits some β-extension. Hence, we will assume to be dealing with instances where such an extension exists. We will call a β-extension minimizing the value of β a *solution*.

3 Preprocessing

The first two steps that will allow us to solve F-BMOE include pruning out certain parts of the face which are provably irrelevant, and reducing the case of f being the outer face to the case of f being an inner face.

Let $\Gamma(G)$ be an orthogonal drawing of a graph G and let f be a face of $\Gamma(G)$. A *feature point* of $\Gamma(G)$ is a point representing either a vertex or a bend of an edge. A *reflex corner* p of f is a feature point that makes an angle larger than π inside f. Also, if p is an anchor, then it is called an *essential* reflex corner. A projection ℓ of a reflex corner p is a horizontal or vertical line-segment in the interior of f that starts at p and ends at its first intersection with the boundary of f. Figure 2 (left) shows two projections ℓ_1 and ℓ_2 of a reflex corner p.

Observe that each projection ℓ of a reflex corner p divides the face f into two connected regions. If p is not essential and one of the two regions contains no reflex corners of its own and no anchors, we call the region *redundant*. Our aim will be to show that such regions can be safely removed from the instance. Namely, we can prove the following, where a *clean*

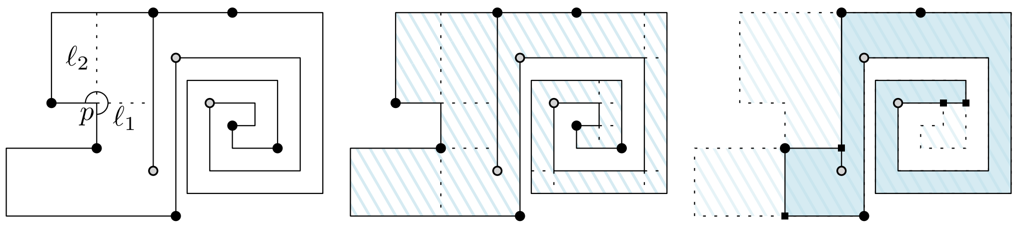


Figure 2 Left: A reflex corner p and its projections ℓ_1 and ℓ_2. Middle: A face (striped) with all its non-essential reflex corners and projections (anchor vertices have a gray filling while non-anchors are solid). Right: The corresponding clean instance (dummy vertices are drawn as small squares).

instance is such that each projection of each non-essential reflex corner in f splits f into two faces, each of which has at least one port on its boundary; see Figure 2 (right).

▶ **Lemma 3.1.** *There is a polynomial-time algorithm that takes as input an arbitrary instance of* F-BMOE *and outputs an equivalent instance which is clean.*

Given Lemma 3.1, we will hereinafter assume that our instances of F-BMOE are clean. Next, consider an instance of F-BMOE where the marked face is the outer face of $\Gamma(H_f)$, and let us begin by constructing a rectangle that bounds $\Gamma(H_f)$ and will serve as a "frame" for any solution. More formally, given an instance $\mathcal{I}$ of F-BMOE and a rectangle R that contains $\Gamma(H_f)$ in its interior, one easily sees that $\mathcal{I}$ admits a solution that lies in the interior of R. Based on this fact, we shall assume that any instance $\mathcal{I}$ is modified such that the outer face of $\Gamma(H_f)$ is a rectangle R containing no anchors (e.g., with four dummy vertices at its corners connected in a cycle). Notice that, while this ensures that f is no longer the outer face, f now contains a hole (that is, H_f is not connected anymore). The goal is now to remove this hole by connecting it to the boundary of R. To do so, let us consider an arbitrary horizontal or vertical line-segment ζ that connects the boundary of R with an edge-segment in the drawing $\Gamma(H_f)$ and intersects no other edge-segment of $\Gamma(H_f)$. Observe that, w.l.o.g., we can assume that each edge-segment in a solution $\Gamma(G_f)$ only intersects ζ in single points (and not in a line-segment); otherwise, one may shift ζ by a sufficiently small ϵ to avoid such intersections. Roughly speaking, we can show that the instance $\mathcal{I}$ can be "cut open" along ζ to construct an equivalent instance where the boundary of the polygon includes R, and to branch in order to determine how the edges in a hypothetical solution cross through ζ. However, to do so we need to ensure that there is a solution, in which the number of such crossings through ζ is bounded. To summarize, we can prove the following.

▶ **Lemma 3.2.** *There is an algorithm that takes as input an instance $\mathcal{I}$ of* F-BMOE *where f is the outer face and solves it in time $2^{\mathcal{O}(k^2 \log k)} \cdot Q(|\mathcal{I}|, k)$, where $Q(a, b)$ is the time to solve an instance of* F-BMOE *of size a and parameter value b such that f is the inner face.*

4 The Sector Graph

For a point $p \in f$, the *bend distance* $\mathrm{bd}(p, (a, d))$ to a port candidate (a, d) is the minimum integer q such that there exists an orthogonal polyline with q bends connecting p and a in the interior of f which arrives to a from direction d.

▶ **Definition 4.1.** Let $\mathcal{P} = ((a_1, d_1), \ldots, (a_q, d_q))$ be an ordered set of port candidates. For each point $p \in f$, we define its bend-vector as the tuple $\mathrm{vect}(p) = (\mathrm{bd}(p, (a_1, d_1)), \ldots, \mathrm{bd}(p, (a_q, d_q)))$.

▶ **Definition 4.2.** Given an ordered set of port candidates $\mathcal{P}$, a *sector* F is a maximal connected set of points with the same bend-vector w.r.t. $\mathcal{P}$.

When $\mathcal{P}$ is not specified explicitly, we will assume it to be the set of port candidates provided by the considered instance of F-BMOE. The face f is now partitioned into a set $\mathcal{F}$ of sectors. It is worth noting that sectors are connected regions in the face f, they do not overlap, and they cover the whole interior of f. We further notice that a sector can be degenerate, it may be a single point or a line-segment, and that pairs of (non-adjacent) sectors may have the same bend-vectors. At this point, we can define a graph representation capturing the adjacencies between the sectors in our instance; see Figure 3.

▶ **Definition 4.3.** Sectors A and B are *adjacent* if there exists a point p in A and a direction $d \in \{\uparrow, \downarrow, \leftarrow, \rightarrow\}$ such that the first point outside of A hit by the ray starting from p in direction d is in B.

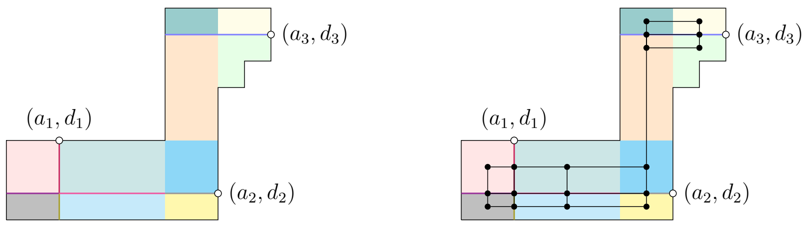


Figure 3 Left: partioning a face f into a set $\mathcal{F}$ of sectors, with three anchors marked using white circles. Right: the graph representation of $\mathcal{F}$.

The *sector graph* $\mathcal{G}$ is the graph whose vertex set is the set of sectors $\mathcal{F}$, and adjacencies of vertices are defined via the adjacency of sectors. It is not difficult to observe that the sector graph is a connected planar graph. Concerning its size, we observe that each sector contains at least one intersection point between two projections and that any such intersection point can be shared by at most nine sectors (four non-degenerate sectors plus five degenerate sectors). Hence the number of vertices in $\mathcal{G}$ is upper-bounded by $9x^2$, where x is the number of feature points in $\Gamma(H_f)$.

We now construct a "universal" point-set with the property that there exists a solution which places feature points only on these points, and where the intersection of the point-set with each sector is upper-bounded by a function of the parameter. Namely, let $\texttt{gridsize}(k) = c \cdot k^8$ (for some constant $c \approx 10^6$). Then we can prove the following:

▶ **Lemma 4.4.** *Given an instance $\mathcal{I}$ of* F-BMOE *we can construct a point-set (called a* sector grid*) in time $\mathcal{O}(|\mathcal{I}|)$ with the following properties:* (1) *$\mathcal{I}$ admits a solution whose feature points all lie on the sector grid, and* (2) *each sector contains at most $\texttt{gridsize}(k)$ points of the sector grid.*

To complete the proof of our fixed-parameter tractability result we proceed by first showing that the sector graphs in fact have treewidth bounded by a function of the parameter k, and then by using this fact to design a dynamic programming algorithm solving F-BMOE.

▶ **Theorem 4.5.** *Let $\mathcal{G}$ be a sector graph of a face f of the drawing $\Gamma(G)$. Then $\mathrm{tw}(\mathcal{G}) \leq (4+4k)^{4k}$. Based on this, there is an algorithm that solves* F-BMOE *in time $2^{k^{\mathcal{O}(1)}} \cdot |V(G_f)|$.*

By combining Theorem 4.5 with Lemma 2.1, we conclude:

▶ **Corollary 4.6.** BMOE *can be solved in time $2^{\kappa^{\mathcal{O}(1)}} \cdot n$, where n is the number of feature points of $\Gamma(H)$.*

References

1 Patrizio Angelini, Ignaz Rutter, and T. P. Sandhya. Extending partial orthogonal drawings. *J. Graph Algorithms Appl.*, 25(1):581–602, 2021. doi:10.7155/jgaa.00573.

2 Sujoy Bhore, Robert Ganian, Liana Khazaliya, Fabrizio Montecchiani, and Martin Nöllenburg. Extending orthogonal planar graph drawings is fixed-parameter tractable, 2023. URL: https://arxiv.org/abs/2302.10046, doi:10.48550/ARXIV.2302.10046.

3 Giuseppe Di Battista, Peter Eades, Roberto Tamassia, and Ioannis G. Tollis. *Graph Drawing: Algorithms for the Visualization of Graphs*. Prentice-Hall, 1999.

4 Christian A. Duncan and Michael T. Goodrich. Planar orthogonal and polyline drawing algorithms. In Roberto Tamassia, editor, *Handbook of Graph Drawing and Visualization*, chapter 7, pages 223–246. CRC Press, 2013.

5 Eduard Eiben, Robert Ganian, Thekla Hamm, Fabian Klute, and Martin Nöllenburg. Extending partial 1-planar drawings. In Artur Czumaj, Anuj Dawar, and Emanuela Merelli, editors, *Automata, Languages, and Programming (ICALP'20)*, volume 168 of *LIPIcs*, pages 43:1–43:19. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPIcs.ICALP.2020.43.

6 Markus Eiglsperger, Sándor P. Fekete, and Gunnar W. Klau. Orthogonal graph drawing. In Michael Kaufmann and Dorothea Wagner, editors, *Drawing Graphs: Methods and Models*, volume 2025 of *LNCS*, chapter 6, pages 121–171. Springer-Verlag, 2001. doi:10.1007/3-540-44969-8_6.

7 Robert Ganian, Thekla Hamm, Fabian Klute, Irene Parada, and Birgit Vogtenhuber. Crossing-optimal extension of simple drawings. In Nikhil Bansal, Emanuela Merelli, and James Worrell, editors, *48th International Colloquium on Automata, Languages, and Programming, ICALP 2021*, volume 198 of *LIPIcs*, pages 72:1–72:17. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPIcs.ICALP.2021.72.

8 Ashim Garg and Roberto Tamassia. On the computational complexity of upward and rectilinear planarity testing. *SIAM J. Comput.*, 31(2):601–625, 2001. doi:10.1137/S0097539794277123.

9 Thekla Hamm and Petr Hlinený. Parameterised partially-predrawn crossing number. In Xavier Goaoc and Michael Kerber, editors, *38th International Symposium on Computational Geometry, 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.

10 Takao Nishizeki and Md. Saidur Rahman. *Planar Graph Drawing*, volume 12 of *Lecture Notes Series on Computing*. World Scientific, 2004. doi:10.1142/5648.

11 Roberto Tamassia. On embedding a graph in the grid with the minimum number of bends. *SIAM J. Comput.*, 16(3):421–444, 1987. doi:10.1137/0216030.

$(1+\varepsilon)$-ANN data structure for curves via subspaces of bounded doubling dimension

Jacobus Conradi[1], **Anne Driemel**[2], **and Benedikt Kolbe**[2]

1 **Department of Computer Science, University of Bonn, Germany**

2 **Hausdorff Center for Mathematics, University of Bonn, Germany**

Abstract

We consider the $(1+\varepsilon)$-Approximate Nearest Neighbour (ANN) Problem for polygonal curves in d-dimensional space under the Fréchet distance and ask to what extent known data structures for doubling spaces can be applied to this problem. For this we identify a subspace of curves which has bounded doubling dimension and is close (in a Gromov-Hausdorff sense) to the target space, in which we solve the $(1+\varepsilon)$-ANN problem, transfering the results to the original space of all curves. Our results imply that there is a data structure for the $(1+\varepsilon)$-ANN problem for any set of parametrized polygonal curves in $\mathbb{R}^d$ with expected preprocessing time in $2^{O(\lambda)}n\log n$ with space used in $2^{O(\lambda)}n$, with a query time of $2^{O(\lambda)}\log n + \varepsilon^{-O(\lambda)}$, where $\lambda = O(k(d+\log(k\Phi(S)\varepsilon^{-1})))$ and $\Phi(S)$ denotes the spread of the set of vertices and edges of the curves in S.

1 Introduction

Given a set S of n points, the Nearest Neighbour Problem is the problem of constructing a data-structure on S which can efficiently identify the point in S that minimizes the distance to a given query point q. The Nearest Neighbour Problem is a fundamental problem. Its variants have long been studied and applied in different areas, such as RNA sequencing [2], disease diagnosing [13], motion pattern detection [4], shape indexing [1] or handwritten digit recognition [9]. The problem has been studied as early as the 1960s [10], and classical results such as the one by Shamos via point location in a Voronoi diagram achieve a query time of $O(\log n)$ while using $O(n\log n)$ space in $\mathbb{R}^2$ [11], which was later improved upon by Kirkpatrick to only require linear space and preprocessing time [8].

One complexity measure often used to generalize many different results to complicated metric spaces is the notion of doubling dimension [5, 7, 14]. The Approximate Nearest Neighbour (ANN) Problem in spaces with low doubling dimension has been studied extensively [7] and results are known that roughly match the bounds known for $\mathbb{R}^d$ [6]. The metric space we are interested in is the space of polygonal curves in $\mathbb{R}^d$ under the Fréchet distance. Polygonal curves naturally arise from any sort of motion tracking, such as GPS data or motion capture data, and are therefore of much interest. The metric space of curves under the Fréchet distance has been shown to have unbounded doubling dimension [3], which suggests that data structures designed for doubling spaces would perform poorly. Our work was inspired by the work of Sheehy and Sheth on the bottleneck distance for persistence diagrams [12], who showed that Clarkson's Algorithm for computing r-nets can be extended to spaces that are close to spaces of bounded doubling dimension. The primary motivation in this paper is to find a workaround to the unbounded doubling dimension, and to leverage the rich background of ANN results for doubling spaces to the Fréchet distance after all.

39th European Workshop on Computational Geometry, Barcelona, Spain, March 29–31, 2023.
This is an extended abstract of a presentation given at EuroCG'23. 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 Basic definitions

An **edge** in $\mathbb{R}^d$ is the continuous map obtained from the linear interpolation of two points a and b in $\mathbb{R}^d$ parametrized over $[0,1]$. We may write $\overline{a\,b}$ to denote this unique edge. A **polygonal curve** T of complexity n parametrized over $[0,1]$ is defined by a set of n points in $\mathbb{R}^d$ and is the result of $(n-2)$ concatenations of the $(n-1)$ edges in $\mathbb{R}^d$ defined by any two consecutive points. We call the underlying points of a polygonal curve of complexity n its **vertices**. For $0 \leq s \leq t \leq 1$ we denote the subcurve of T from $T(s)$ to $T(t)$ by $T[s,t]$.

▶ **Definition 2.1** (Fréchet distance)**.** Define the Fréchet distance for two curves X, Y in $\mathbb{R}^d$ as

$$\mathrm{d}_{\mathcal{F}}(X,Y) = \inf_{f,g:[0,1]\to[0,1]} \max_{t\in[0,1]} \|X(f(t)) - Y(g(t))\|$$

where f and g are continuous, non-decreasing and surjective.

The Fréchet distance is generally not a metric, but rather a pseudo-metric, as there are curves $X \neq Y$, such that $\mathrm{d}_{\mathcal{F}}(X,Y) = 0$. However this is easily remedied by considering the quotient space induced by the equivalence relation $X \sim Y \iff \mathrm{d}_{\mathcal{F}}(X,Y) = 0$.

Problem definition ($(1+\varepsilon)$-ANN) Let $(\mathcal{M}, \mathrm{d}_{\mathcal{M}})$ be a metric space. Let $P \subset \mathcal{M}$ be a set of points in $\mathcal{M}$ and a parameter $\varepsilon > 0$ be given. For a given point $q \in \mathcal{M}$, the problem of finding the $(1+\varepsilon)$-Approximate Nearest Neighbour ($(1+\varepsilon)$-ANN) asks to identify some point $\hat{x} \in P$ whose distance to q approximates the distance to the nearest neighbour in P. That is for every $x \in P$ it holds that $\mathrm{d}_{\mathcal{M}}(q,\hat{x}) \leq (1+\varepsilon)\,\mathrm{d}_{\mathcal{M}}(q,x)$.

To answer such queries when $\mathcal{M}$ is the space of polygonal curves with the Fréchet distance, we construct a subspace of curves that has bounded doubling dimension and is 'arbitrarily close' to the space of all curves. This then allows us to transfer results from spaces with bounded doubling dimension to these close spaces.

We denote by $\mathbb{X}^{d,k}$ the set of polygonal curves in $\mathbb{R}^d$ with complexity k. We further write $\mathbb{X}^{d,k}_{\Lambda}$ for the subset of polygonal curves in $\mathbb{X}^{d,k}$ where the length of each edge is bounded by Λ. The **doubling constant** of any metric space $(\mathcal{M}, \mathrm{d}_{\mathcal{M}})$ is defined as the minimal number ν, such that for any $x \in \mathcal{M}$ and any $r > 0$ the disk of radius r centered at x, denoted by $\mathrm{D}^{\mathcal{M}}_r(x)$, is contained in the union of at most ν disks with radius $r/2$. The **doubling dimension** of $(\mathcal{M}, \mathrm{d}_{\mathcal{M}})$ is defined as $\log_2(\nu)$. We may omit the metric space in the notation of a disk, whenever the metric space is clear, writing $\mathrm{D}_r(x)$ instead of $\mathrm{D}^{\mathcal{M}}_r(x)$ for $x \in \mathcal{M}$.

It turns out that for $k \geq 3$ the doubling dimension of $(\mathbb{X}^{d,k}, \mathrm{d}_{\mathcal{F}})$ is unbounded [3]. A slight modification to this construction yields an unbounded doubling dimension of $\mathbb{X}^{d,k}_{\Lambda}$ for any $\Lambda > 0$. We thus turn to the following subspace of $(\mathbb{X}^{d,k}, \mathrm{d}_{\mathcal{F}})$ which we want to analyze.

▶ **Definition 2.2** ((μ,ε)-curves)**.** For any $\varepsilon > 0$ and $\mu \in \mathbb{N}$ define the space of (μ,ε)-curves in $\mathbb{X}^{d,k}$ as the subspace of $(\mathbb{X}^{d,k}, \mathrm{d}_{\mathcal{F}})$ induced by the set of polygonal curves in $\mathbb{X}^{d,k}$ whose edge lengths are all exact multiples of ε. Further we require the edge lengths to be bounded by $\mu\varepsilon$. The space of (μ,ε)-curves in $\mathbb{X}^{d,k}$ is a natural subspace of $(\mathbb{X}^{d,k}_{\mu\varepsilon}, \mathrm{d}_{\mathcal{F}})$.

▶ **Lemma 2.3.** *Let $P \in \mathbb{X}^{d,k}_{\Lambda}$ be a polygonal curve and $\varepsilon > 0$. We can construct a $(\lceil \Lambda/\varepsilon \rceil + 1, \varepsilon)$-curve P' in $\mathbb{X}^{d,k}$ such that $\mathrm{d}_{\mathcal{F}}(P,P') \leq \varepsilon/2$ in $O(k\log(\Lambda/\varepsilon))$ time.*

Omitted proofs throughout this paper can be found in the full version.

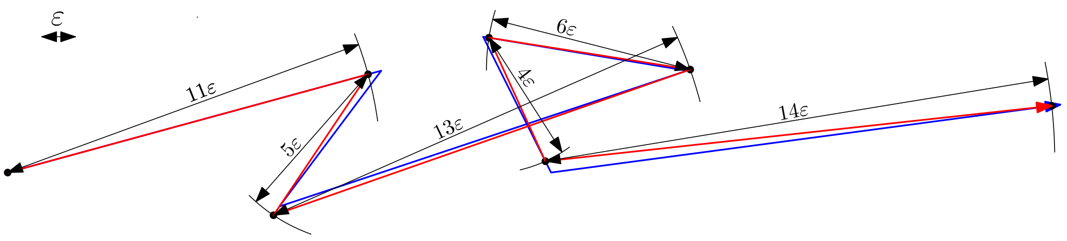


Figure 1 A curve $P \in \mathbb{X}^{d,k}_{\Lambda}$ in blue, and an ε-curve close to P resulting from Lemma 2.3 in red.

3 Bounding the doubling dimension of the space of (μ, ε)-curves

We now study the doubling dimension of the space of (μ, ε)-curves in $\mathbb{X}^{d,k}$. Unfortunately, our bound is non-constructive. As such, it does not provide a doubling oracle that, for a given disk of radius r, outputs a set of disks of radius $r/2$ which cover the disk of radius r.

Before diving into the analysis of the doubling dimension of the space of (μ, ε)-curves, we begin by analysing properties of the ambient space $\mathbb{R}^d$. For this we often inspect so called Δ-neighbourhoods of subsets of $\mathbb{R}^d$. For any subset A the **Δ-neighbourhood** of A is defined by $N_\Delta(A) = \{x \in \mathbb{R}^d \mid \exists a \in A : d(x, a) \leq \Delta\}$. Note that the Δ-neighbourhood of a single point x coincides with a disk of radius Δ centered at x.

It is well-known that the doubling dimension of $\mathbb{R}^d$ is in $\Theta(d)$. The following lemma is a generalization of the fact, that the doubling dimension of $\mathbb{R}^d$ is in $O(d)$.

▶ **Lemma 3.1.** *For any $r > 0$ and $c > 1$, any disk $\mathrm{D}_r(p) \subset \mathbb{R}^d$ can be covered by $O((2c+1)^d)$ disks of radius r/c.*

The main result we now want to prove is the following theorem.

▶ **Theorem 3.2.** *Let $k, \mu, d \in \mathbb{N}$ and $\varepsilon > 0$. The doubling dimension of the space of (μ, ε)-curves is bounded by $O(k(d + \log(k\mu)))$.*

Let P be a (μ, ε)-curve in $\mathbb{X}^{d,k}$. Our objective is to cover the Δ-neighbourhood of P with respect to $\mathrm{d}_\mathcal{F}$ with disks of radius $\Delta/2$. We encounter the question of where, given $p \in \mathbb{R}^d$, we may place a second point q such that there is a subcurve of P that is close to $\overline{pq}$. Indeed, for any curve Q with $\mathrm{d}_\mathcal{F}(P, Q) \leq \Delta$, the endpoint q of any edge $\overline{pq}$ of Q is such a point, depending only on the starting point p of the edge in question.

▶ **Definition 3.3.** Let P be a polygonal curve and $\lambda \geq 0$ and $\Delta \geq 0$. For $s \in [0, 1]$ define the locus of edge endpoints of edges close to subcurves of P starting at the parameter s as

$$L_{\lambda,\Delta}(P, s) = \left\{q \in \mathbb{R}^d \middle| \exists p \in \mathbb{R}^d \text{ and } \exists t \in [s, 1] \text{ with } \|p - q\| = \lambda \text{ and } \mathrm{d}_\mathcal{F}(P[s, t], \overline{pq}) \leq \Delta\right\}.$$

It turns out that the set $L_{\lambda,\Delta}(P, s)$ is contained in a disk of constant size.

▶ **Lemma 3.4.** *Let P be a polygonal curve. Let $\lambda \geq 0$ and $\Delta \geq 0$ be given. Then for every $s \in [0, 1]$ there is a point $p^* \in \mathbb{R}^d$, such that $L_{\lambda,\Delta}(P, s) \subset \mathrm{D}_{5\Delta}(p^*)$.*

▶ **Definition 3.5.** Let P be a polygonal curve and $\lambda \geq 0$ and $\Delta \geq 0$. For $p \in \mathbb{R}^d$ define the locus of edge endpoints of edges starting at p which are close to subcurves of P as the set

$$\mathscr{L}_{\lambda,\Delta}(P, p) = \left\{q \in \mathbb{R}^d \middle| \|p - q\| = \lambda \text{ and } \exists s, t \text{ with } 0 \leq s \leq t \leq 1 \text{ and } \mathrm{d}_\mathcal{F}(P[s, t], \overline{pq}) \leq \Delta\right\}.$$

Similarly to Lemma 3.4 we can identify k disks that cover $\mathscr{L}_{\lambda,\Delta}(P, p)$ for given P, λ, Δ and p.

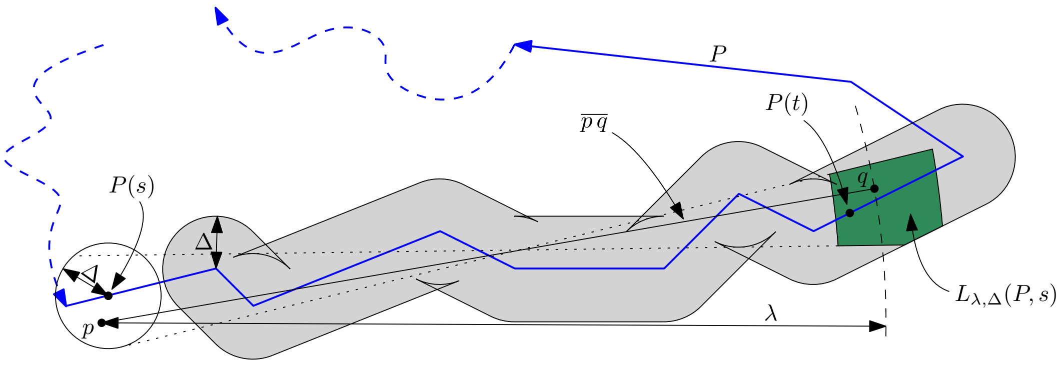


Figure 2 Illustration of the set $L_{\lambda,\Delta}(P,s)$ in dark green, together with the points p and $P(t)$ realizing a point q in $L_{\lambda,\Delta}(P,s)$, that is $\|p-q\|=\lambda$ and $\mathrm{d}_{\mathcal{F}}(P[s,t],\overline{pq})\leq\Delta$.

▶ **Lemma 3.6.** *Let $P\in\mathbb{X}^{d,k}$ be a polygonal curve. Let $\lambda\geq 0$ and $\Delta\geq 0$ be given. Then for every $p\in\mathbb{R}^d$ there are k points $p_1^*,\ldots,p_k^*\in\mathbb{R}^d$ such that $\mathscr{L}_{\lambda,\Delta}(P,p)\subset\bigcup_{i=1}^k \mathrm{D}_{5\Delta}(p_i^*)$.*

Proof. The set $I=\{s\in[0,1]\mid P(s)\in\mathrm{D}_\Delta(p)\}$ can be described as a disjoint union of at most k closed intervals, as the complexity of P is bounded by k. Assume that it is described by exactly k such intervals, that is, $I=\bigcup_{i=1}^k[l_i,r_i]$.

It suffices to show that $\mathscr{L}_{\lambda,\Delta}(P,p)\subset\bigcup_{i=1}^k L_{\lambda,\Delta}(P,l_i)$, as Lemma 3.4 then implies the claim. Assume that an arbitrary $q\in\mathscr{L}_{\lambda,\Delta}(P,p)$ is given. Then by definition there are values $0\leq s\leq t\leq 1$ such that $\mathrm{d}_{\mathcal{F}}(P[s,t],\overline{pq})\leq\Delta$. This implies that $\|p-P(s)\|\leq\Delta$, and hence $s\in I$ and in turn $s\in[l_i,r_i]$ for some $1\leq i\leq k$. But then the subcurve $P[l_i,s]$ is contained in $\mathrm{D}_\Delta(p)$, and thus $\mathrm{d}_{\mathcal{F}}(P[l_i,t],\overline{pq})\leq\Delta$ implying that $q\in L_{\lambda,\Delta}(P,l_i)$ and thus the claim. ◀

▶ **Corollary 3.7.** *For every polygonal curve P in $\mathbb{R}^d$, $\Delta>0$, $\lambda>0$, $c>1$ and point $p\in\mathbb{R}^d$ the set $N_{\Delta/c}\left(\mathscr{L}_{\lambda,(1+c^{-1})\Delta}(P,p)\right)$ can be covered by $O(k(10c+3)^d)$ many disks of radius Δ/c.*

▶ **Lemma 3.8.** *Let P be a polygonal curve. Let $\lambda\geq 0$, $\Delta\geq 0$ and $c\geq 1$ be given. Then for every $p,p'\in\mathbb{R}^d$ with $\|p-p'\|\leq\Delta/c$ we have that $\mathscr{L}_{\lambda,\Delta}(P,p)\subset N_{\Delta/c}\left(\mathscr{L}_{\lambda,(1+c^{-1})\Delta}(P,p')\right)$.*

We now prove a stronger version of Theorem 3.2, which allows us to analyse both the subspace doubling constant as well as the doubling constant of (μ,ε)-curves in $\mathbb{X}^{d,k}$.

▶ **Lemma 3.9.** *Let $k,\mu,d\in\mathbb{N}$ and $\varepsilon>0$. Let a (μ,ε)-curve P in $\mathbb{X}^{d,k}$ be given, as well as $\Delta>0$ and $c\geq 1$. There is a family of curves $\mathcal{C}_P\subset\mathbb{X}^{d,k}_{\mu\varepsilon+\Delta/c}$ of size $O(k\mu(10c+3)^d)^k$, such that for any (μ,ε)-curve Q with $\mathrm{d}_{\mathcal{F}}(P,Q)\leq\Delta$ there is a $Q^*\in\mathcal{C}_P$ with $\mathrm{d}_{\mathcal{F}}(Q,Q^*)\leq\Delta/c$.*

Proof. We construct the set $\mathcal{C}_P$ as follows. First, choose an element $(m_1,\ldots,m_{k-1})\in\{1,\ldots,\mu\}^{k-1}$. Next, choose one circle center of a cover of $\mathrm{D}_\Delta(P(0))$ consisting of $O((2c+1)^d)$ many disks of radius r/c, which exists by Lemma 3.1. Iteratively choose one point among the circle centers of a cover of $N_{\Delta/c}\left(\mathscr{L}_{m_{i-1}\varepsilon,(1+c^{-1})\Delta}(P,q_{i-1}^*)\right)$ of Corollary 3.7, consisting of $O(k(10c+3)^d)$ many disks of radius r/c as the vertex q_i^* of Q^* for $i\leq k$. Then $Q^*\in\mathbb{X}^{d,k}_{\mu\varepsilon+\Delta/c}$, as for any i the fact that q_i^* lies in $N_{\Delta/c}\left(\mathscr{L}_{m_{i-1}\varepsilon,(1+c^{-1})\Delta}(P,q_{i-1}^*)\right)$ implies that there is a point $q\in\mathscr{L}_{m_{i-1}\varepsilon,(1+c^{-1})\Delta}(P,q_{i-1}^*)$, with $\|q_{i-1}^*-q\|=m_{i-1}\varepsilon$ and $\|q-q_i^*\|\leq\Delta/c$. Hence, $\|q_i^*-q_{i-1}^*\|\leq m_1\varepsilon+\Delta/c\leq\mu\varepsilon+\Delta/c$. To account for all the choices, we have that $|\mathcal{C}_P|=O(k\mu(10c+3)^d)^k$.

Let Q be a given (μ,ε)-curve, with $\mathrm{d}_{\mathcal{F}}(P,Q)\leq\Delta$. The curve Q consists of $k-1$ edges and induces an ordered set $(m_1,\ldots,m_{k-1})\in\{0,\ldots,\mu\}^{k-1}$ representing the lengths of the edges in order. Let $q_1, \ldots, q_k$ be the vertices of Q. For all $1 \leq i \leq k$ it holds that $q_i \in \mathscr{L}_{m_{i-1}\varepsilon,\Delta}(P, q_{i-1})$, by construction.

As $\mathrm{d}_{\mathcal{F}}(P,Q) \leq \Delta$, the first vertex q_1 lies in $\mathrm{D}_\Delta(P(0))$, and thus there is a point q_1^* of the cover of $\mathrm{D}_\Delta(P(0))$ consisting of disks of radius r/c, that lies at distance at most Δ/c to q_1. For every subsequent q_i, by Lemma 3.8 and because $q_i \in \mathscr{L}_{m_{i-1}\varepsilon,\Delta}(P, q_{i-1})$, $q_i \in N_{\Delta/c}\left(\mathscr{L}_{m_{i-1}\varepsilon,(1+c^{-1})\Delta}(P, q_{i-1}^*)\right)$ and thus there is a point q_i^* of the Δ/c-cover of $N_{\Delta/c}\left(\mathscr{L}_{m_{i-1}\varepsilon,(1+c^{-1})\Delta}(P, q_{i-1}^*)\right)$ that is at distance at most Δ/c to q_i. This implies that there is an element Q^* (defined by exactly this choice of points) in $\mathcal{C}_P$ that has distance $\mathrm{d}_{\mathcal{F}}(Q, Q^*) \leq \Delta/c$. ◀

Proof of Theorem 3.2. Let P be a (μ,ε)-curve in $\mathbb{X}^{d,k}$ and a value Δ be given. By Lemma 3.9, there is a family $\mathcal{C}_P$ of curves of size $O(k\mu(43)^d)^k$ in $\mathbb{X}^{d,k}_{\mu\varepsilon+\Delta/4} \subset \mathbb{X}^{d,k}$, such that for any (μ,ε)-curve Q with $\mathrm{d}_{\mathcal{F}}(P,Q) \leq \Delta$ there is curve Q^* in $\mathcal{C}_P$ with $\mathrm{d}_{\mathcal{F}}(Q,Q^*) \leq \Delta/4$. For any $Q^* \in \mathcal{C}_P$ identify some (μ,ε)-curve $\widehat{Q^*}$ such that $\mathrm{d}_{\mathcal{F}}(Q^*,\widehat{Q^*}) \leq \Delta/4$. If no such element exists, ignore Q^*. Otherwise for any (μ,ε)-curve Q with $\mathrm{d}_{\mathcal{F}}(P,Q) \leq \Delta$ there is a curve Q^* in $\mathcal{C}_P$ with $\mathrm{d}_{\mathcal{F}}(Q,Q^*) \leq \Delta/4$, and thus by the triangle inequality there is a (μ,ε)-curve $\widehat{Q^*}$ with $\mathrm{d}_{\mathcal{F}}(Q,\widehat{Q^*}) \leq \Delta/2$, proving the bounded doubling dimension. ◀

4 Approximate Nearest Neighbour

Har-Peled et al. [6] showed the following result for the $(1+\varepsilon)$-ANN problem in metric spaces of bounded doubling dimension.

▶ **Theorem 4.1** ([6]). *Given a set S of n points in a metric space $\mathcal{M}$ of bounded doubling dimension ν, one can construct a data-structure for answering $(1+\varepsilon)$-approximate nearest neighbour queries. The query time is $2^{O(\nu)}\log n + \varepsilon^{-O(\nu)}$, the expected preprocessing time is $2^{O(\nu)} n \log n$ and the space used is $2^{O(\nu)} n$.*

A careful reading reveals an important specification for our purposes, namely, that the doubling dimension is that of the n-point metric space defined by S induced by the metric space $\mathcal{M}$ and not of the ambient metric space $\mathcal{M}$.

▶ **Lemma 4.2.** *Let $(\mathcal{M}, \mathrm{d}_{\mathcal{M}})$ be a metric space, and let S be some subset of $\mathcal{M}$. Then the doubling dimension of $(S, \mathrm{d}_{\mathcal{M}})$ is at most twice the doubling dimension of $(\mathcal{M}, \mathrm{d}_{\mathcal{M}})$.*

▶ **Lemma 4.3.** *Given a set S of n polygonal curves in $\mathbb{X}^{d,k}_\Lambda$ and parameters $0 < \varepsilon < 1$ and $\varepsilon' > 0$, one can construct a data-structure such that for given $q \in \mathbb{X}^{d,k}$ it outputs an element $s^* \in S$ such that for all $s \in S$ it holds that $\mathrm{d}_{\mathcal{F}}(s^*, q) \leq (1+\varepsilon)\mathrm{d}_{\mathcal{F}}(s,q) + \varepsilon'$. The query time is $2^{O(\nu)}\log n + \varepsilon^{-O(\nu)}$, the expected preprocessing time is $2^{O(\nu)} n \log n$ and the space used is $2^{O(\nu)} n$, where the doubling dimension ν is given by $O(k(d + \log(k(1+\Lambda/\varepsilon'))))$.*

Proof. Define $\hat{\varepsilon} = \varepsilon'/2$. Let $\mu = \lceil \Lambda/\hat{\varepsilon} \rceil + 1 = \Theta(1 + \Lambda/\varepsilon')$. We begin by simplifying every polygonal curve $s \in S$ via Lemma 2.3, resulting in a set S' of $(\mu,\hat{\varepsilon})$-curves. This takes $O(\log(\mu)nk)$ time, which is in $O(2^\nu n)$. As S' lies in the space of $(\mu,\hat{\varepsilon})$-curves, the doubling dimension of the set S' with the Fréchet distance is bounded by $\nu = O(k(d+\log(k(1+\Lambda/\varepsilon'))))$ via Theorem 3.2 and Lemma 4.2. Note that for every $s \in S$ and its simplification $s' \in S'$ it holds that $\mathrm{d}_{\mathcal{F}}(s, s') \leq \hat{\varepsilon}/2$. We apply Theorem 4.1 to the set S' and ε. Note that Theorem 4.1 assumes that the distance between any two points in the metric space of (μ,ε)-curves can be computed in $O(1)$ time. However, the continuous Fréchet distance takes polynomial time in k. On the other hand, both $2^{k\log k}$ and $\varepsilon^{-k\log k}$ dominate $\mathrm{poly}(k)$ for $\varepsilon < 1$. Thus the running time is indeed as claimed. We then query the data structure with q, returning an element $\widehat{s'}$ such that for every $s' \in S'$ it holds that $\mathrm{d}_{\mathcal{F}}(q, \widehat{s'}) \leq (1+\varepsilon)\mathrm{d}_{\mathcal{F}}(q, s')$. Lastly, the element of S returned by the data structure will be the element $\widehat{s} \in S$ which corresponds to $\widehat{s'}$. We then get for every $s \in S$ that

$$\begin{aligned}\mathrm{d}_{\mathcal{F}}(q,\widehat{s}) &\leq \mathrm{d}_{\mathcal{F}}(q,\widehat{s'}) + \hat{\varepsilon}/2 \leq (1+\varepsilon)\mathrm{d}_{\mathcal{F}}(q,s') + \hat{\varepsilon}/2 \leq (1+\varepsilon)(\mathrm{d}_{\mathcal{F}}(q,s) + \hat{\varepsilon}/2) + \hat{\varepsilon}/2 \\ &\leq (1+\varepsilon)\mathrm{d}_{\mathcal{F}}(q,s) + \hat{\varepsilon} + \varepsilon\hat{\varepsilon}/2 = (1+\varepsilon)\mathrm{d}_{\mathcal{F}}(q,s) + \hat{\varepsilon}(1+\varepsilon/2) \\ &\leq (1+\varepsilon)\mathrm{d}_{\mathcal{F}}(q,s) + \varepsilon'. \end{aligned}$$ ◀

▶ **Definition 4.4** (spread)**.** For a point set P in some metric space $(\mathcal{M}, d_{\mathcal{M}})$ we define the spread $\Phi(P)$ as the ratio between the maximal and minimal pairwise distance of points in P. Similarly, define the spread $\Phi(S)$ of a collection of sets as the ratio between the maximal and minimal non-zero pairwise distances of sets in S where for two sets $A, B \subset \mathcal{M}$ their distance is defined as $\mathrm{d}_{\mathcal{M}}(A,B) = \min_{a\in A}\min_{b\in B}\mathrm{d}_{\mathcal{M}}(a,b)$.

▶ **Theorem 4.5.** *Given a set S of n polygonal curves in $\mathbb{X}^{d,k}$ and $0 < \varepsilon \leq 1$ one can construct a data-structure answering $(1+\varepsilon)$-approximate nearest neighbour queries. The query time is $2^{O(\nu)}\log n + \varepsilon^{-O(\nu)}$, the expected preprocessing time is $2^{O(\nu)}n\log n$ and the space used is $2^{O(\nu)}n$, where $\nu = O(k(d + \log(k\Phi(S)\varepsilon^{-1})))$, where $\Phi(S)$ denotes the spread of set of vertices and edges of the curves in S.*

Proof Sketch. Let $\varepsilon' = \varepsilon/4$ and $\varepsilon'' = \varepsilon'\left(\min_{s\neq s'\in S}\mathrm{d}_{\mathcal{F}}(s,s')\right)$. Let $E(S)$ be the set of edges of curves in S and let further $\Lambda = \max_{e\in E(S)}\|e\|$, thus clearly $S \subset \mathbb{X}_{\Lambda}^{d,k}$. We then apply Lemma 4.3 with ε' and ε'' resulting in the described data structure. The correctness follows directly, with the running time following as $\Lambda/\min_{s\neq s'\in S}\mathrm{d}_{\mathcal{F}}(s,s') + 1 = O(\Phi(S))$. ◀

References

1 Jeffrey S Beis and David G Lowe. Shape indexing using approximate nearest-neighbour search in high-dimensional spaces. In Proceedings of IEEE computer society conference on computer vision and pattern recognition, pages 1000–1006. IEEE, 1997.

2 Eckart Bindewald and Bruce A Shapiro. Rna secondary structure prediction from sequence alignments using a network of k-nearest neighbor classifiers. RNA, 12(3):342–352, 2006. doi:10.1261/rna.2164906.

3 Anne Driemel, Amer Krivošija, and Christian Sohler. Clustering time series under the fréchet distance. In Proceedings of the twenty-seventh annual ACM-SIAM symposium on Discrete algorithms, pages 766–785. SIAM, 2016. doi:1.9781611974331.ch55.

4 Joachim Gudmundsson, Marc van Kreveld, and Bettina Speckmann. Efficient detection of motion patterns in spatio-temporal data sets. In Proceedings of the 12th Annual ACM International Workshop on Geographic Information Systems, page 250–257. ACM, 2004. doi:10.1145/1032222.1032259.

5 Anupam Gupta, Robert Krauthgamer, and James R Lee. Bounded geometries, fractals, and low-distortion embeddings. In 44th Annual IEEE Symposium on Foundations of Computer Science, 2003. Proceedings., pages 534–543. IEEE, 2003. doi:10.1109/SFCS.2003.1238226.

6 Sariel Har-Peled and Manor Mendel. Fast construction of nets in low-dimensional metrics and their applications. SIAM Journal on Computing, 35(5):1148–1184, 2006. doi:10.1137/S0097539704446281.

7 David R. Karger and Matthias Ruhl. Finding nearest neighbors in growth-restricted metrics. In Proceedings of the Thiry-Fourth Annual ACM Symposium on Theory of Computing, page 741–750. Association for Computing Machinery, 2002. doi:10.1145/509907.510013.

8 David Kirkpatrick. Optimal search in planar subdivisions. SIAM Journal on Computing, 12(1):28–35, 1983. doi:10.1137/0212002.

9 Yuchun Lee. Handwritten digit recognition using k nearest-neighbor, radial-basis function, and backpropagation neural networks. Neural computation, 3(3):440–449, 1991. doi:10.1162/neco.1991.3.3.440.

10 Marvin Minsky and Seymour Papert. Perceptrons: An Introduction to Computational Geometry. The MIT Press, 1969.

11 Michael Ian Shamos and Dan Hoey. Closest-point problems. In 16th Annual Symposium on Foundations of Computer Science (sfcs 1975), pages 151–162, 1975. doi:10.1109/SFCS.1975.8.

12 Donald R. Sheehy and Siddharth S. Sheth. Nearly-Doubling Spaces of Persistence Diagrams. In 38th International Symposium on Computational Geometry (SoCG 2022), volume 224, pages 60:1–60:15. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPIcs.SoCG.2022.60.

13 Mai Shouman, Tim Turner, and Rob Stocker. Applying k-nearest neighbour in diagnosing heart disease patients. International Journal of Information and Education Technology, 2(3):220–223, 2012. doi:10.7763/IJIET.2012.V2.114.

14 Kunal Talwar. Bypassing the embedding: algorithms for low dimensional metrics. In Proceedings of the thirty-sixth annual ACM symposium on Theory of computing, pages 281–290, 2004. doi:10.1145/1007352.1007399.

Improved Bounds for Discrete Voronoi Games

Mark de Berg[1] **and Geert van Wordragen**[2]

1 **Department of Mathematics and Computer Science, TU Eindhoven, the Netherlands**
`M.T.d.Berg@tue.nl`

2 **Department of Computer Science, Aalto University, Finland**
`Geert.vanWordragen@aalto.fi`

Abstract

In the planar one-round discrete Voronoi game, two players $\mathcal{P}$ and $\mathcal{Q}$ compete over a set V of n voters represented by points in $\mathbb{R}^2$. First, $\mathcal{P}$ places k points, then $\mathcal{Q}$ places ℓ points, and then each voter $v \in V$ is won by the player who has placed a point closest to v. We present lower bounds on the number of voters that $\mathcal{P}$ can always win, which improve the existing bounds for all $k \geqslant 4$. As a by-product, we obtain improved bounds on small ε-nets for convex ranges.

1 Introduction

In the *discrete Voronoi game*, two players compete over a set V of n voters in $\mathbb{R}^d$. Player $\mathcal{P}$ places a set P of k points, player $\mathcal{Q}$ places a set Q of ℓ points that are disjoint from the points in P, and then each voter $v \in V$ is won by the player who has placed a point closest to v. In other words, each player wins the voters located in its Voronoi cell in the Voronoi diagram $\mathrm{Vor}(P \cup Q)$. In case of ties, when a voter v lies on the boundary between a Voronoi cell owned by $\mathcal{P}$ and a Voronoi cell owned by $\mathcal{Q}$, then v is won by player $\mathcal{P}$. There are two variants of the discrete Voronoi game. In the *multiple-round* game, $k = \ell$ and the two players alternate placing points. In the *one-round* game, which is the variant we are interested in, $\mathcal{P}$ first places k points and then $\mathcal{Q}$ places ℓ points; here k and ℓ need not be equal. The one-round discrete Voronoi game was introduced by Banik *et al.* [2].

The discrete one-round Voronoi game for $k = \ell = 1$ is closely related to the concept of plurality points in spatial voting theory [9]. In this theory, there is a d-dimensional policy space, and voters are modelled as points indicating their preferred policies. A *plurality point* is then a proposed policy that would win at least $\lceil n/2 \rceil$ voters against any competing policy. Phrased in terms of Voronoi games, this means that $\mathcal{P}$ can place a single point that wins at least $\lceil n/2 \rceil$ voters against any single point placed by $\mathcal{Q}$. The discrete Voronoi game with $k > 1$ and $\ell = 1$ can be thought of as an election where a coalition of k parties is colluding against a single other party. Another way to interpret Voronoi games is as a *competitive facility-location problem*, which has also been studied in a graph-theoretic setting (see e.g. [1, 8, 11]).

Previous work. The one-round discrete Voronoi game leads to interesting algorithmic as well as combinatorial problems. The algorithmic problem is to compute an optimal set of locations for a given set V of voters, for player $\mathcal{P}$ (assuming $\mathcal{Q}$ responds optimally) and for player $\mathcal{Q}$ (for a given set of locations placed by $\mathcal{P}$) [2, 5]. We are interested in the combinatorial problem, which is to prove bounds on the number of voters that player $\mathcal{P}$ can win, assuming player $\mathcal{Q}$ responds optimally to the points played by $\mathcal{P}$. Tight bounds are only known for the case $k = \ell = 1$, where Chawla *et al.* [7] showed the following: for any set V of n voters in $\mathbb{R}^d$, player $\mathcal{P}$ can win at least $\lceil n/(d+1) \rceil$ voters and at most $\lceil n/2 \rceil$ voters, and

39th European Workshop on Computational Geometry, Barcelona, Spain, March 29–31, 2023.
This is an extended abstract of a presentation given at EuroCG'23. 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.

reference	$k = 1$	$k = 2$	$k = 3$	$k = 4$	$k = 5$	arbitrary k
Banik *et al.* [3]	1/3	3/7	7/15	15/31	21/41	$1 - \frac{42}{k}$
this paper				1/2	11/21	$1 - \frac{20}{k}$

Table 1 Lower bounds on the fraction of voters that $\mathcal{P}$ can win on any voter set in $\mathbb{R}^2$, when $\mathcal{P}$ has k points and $\mathcal{Q}$ has a single point, and the L_2-metric is used.

these bounds are tight. Situations where $\mathcal{P}$ can win $\lceil n/2 \rceil$ voters are particularly interesting, as these correspond to the existence of a plurality point in voting theory. The bounds just mentioned imply that a plurality point does not always exist. In fact, a plurality point only exists for certain very symmetric point sets, as shown by Wu *et al.* [12]. De Berg *et al.* [4] showed how to test in $O(n \log n)$ time if a voter set admits a plurality point.

The combinatorial problem for $k > 1$ and $\ell = 1$ was studied by Banik *et al.* [3]. Here player $\mathcal{P}$ will never be able to win more than $\left(1 - \frac{1}{2k}\right) n$ voters, because player $\mathcal{Q}$ can always win at least half of the voters of the most crowded Voronoi cell in $\mathrm{Vor}(P)$. Banik *et al.* [3] present two methods to derive lower bounds on the number of voters that $\mathcal{P}$ can always win. Below we discuss their results in $\mathbb{R}^2$, but we note that they generalize their methods to $\mathbb{R}^3$.

The first method uses a (weak) ε-net for convex ranges on the voter set V, that is, a point set N such that any convex range R containing more than εn voters, will also contain a point from N. Now, if $\ell = 1$ then the voters won by $\mathcal{Q}$ lie in a single Voronoi cell in $\mathrm{Vor}(P \cup Q)$. Since Voronoi cells are convex, this means that if we set $P := N$ then $\mathcal{P}$ wins at least $(1-\varepsilon)n$ voters. Banik *et al.* use the ε-net construction for convex ranges by Mustafa and Ray [10]. There is no closed-form expression for the size of their ε-net, but the method can give a $(4/7)$-net of size 2, for instance, and an $(8/15)$-net of size 3. The smallest size for which they obtain an ε-net for some $\varepsilon \leqslant 1/2$, is $k = 5$. The second method of Banik *et al.* uses an ε-net for disks, instead of convex sets. This is possible because one can show that a point $q \in Q$ that wins α voters, must have a disk around it that covers at least $\lfloor \alpha/6 \rfloor$ voters without containing a point from P. Banik *et al.* also present a $(7/k)$-net for disks of size k. Hence, $\mathcal{P}$ wins at least $\left(1 - \frac{42}{k}\right) n$ voters, which is better than the first method when $k \geqslant 137$.

Our results. We study the combinatorial question—how many voters can player $\mathcal{P}$ win from any voter set V of size n, under optimal play from $\mathcal{Q}$—in the planar setting, for $k > 1$ and $\ell = 1$. We obtain the following results.

- We improve[1] over the ε-net bounds by Mustafa and Ray [10] for convex ranges. This gives an improvement over the results of Banik *et al.* [3] on the fraction of voters that $\mathcal{P}$ can win when $k \geqslant 4$ and k is relatively small. We do not have a closed-form expression for the size of our ε-net, but Theorem 2.3 gives a formula based on the quality of smaller ε-nets, and Table 1 shows how our bounds compare to those of Banik *et al.* for $k = 4, 5$. Note that our bounds improve the smallest k for which $\mathcal{P}$ can win at least half the voters, from $k = 5$ to $k = 4$.
- We present a new strategy for player $\mathcal{P}$ for large k. Unlike the strategies by Banik *et al.*, it is not based on ε-nets. Instead, it uses a quadtree-based approach. By combining this approach with several other ideas and using our ε-net method as a subroutine, we

[1] Our definition of ε-net is slightly weaker than usual, since a range missing the ε-net may contain up to $\lceil \varepsilon n \rceil$ points, instead of $\lfloor \varepsilon n \rfloor$ points.

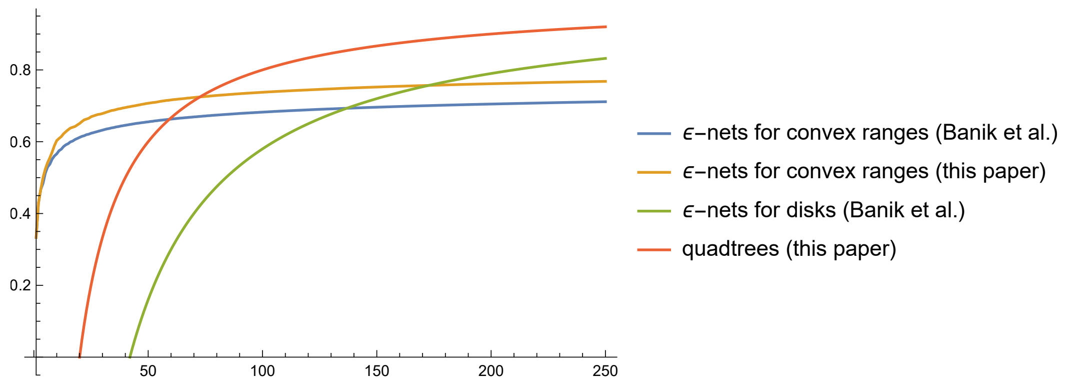


Figure 1 Lower bounds on the fraction of voters that $\mathcal{P}$ can win on any voter set in $\mathbb{R}^2$ as a function of k (the number of points of $\mathcal{P}$) when $\mathcal{Q}$ has a single point. The red and green graphs do not intersect, so for large k the quadtree method gives the best solution.

are able to show that there is a set P of k points that guarantees that $\mathcal{P}$ wins at least $\left(1 - \frac{20}{k}\right) n - 6$ voters, which significantly improves the $\left(1 - \frac{42}{k}\right) n$ bound of Banik *et al.* Fig. 1 shows the bounds obtained by the various methods in a graphical way.

2 Better ε-nets for convex ranges

Below we present a new method to construct an ε-net for convex ranges in the plane, which improves the results of Mustafa and Ray [10]. As mentioned in the introduction, this implies improved bounds on the number of voters $\mathcal{P}$ can win with k points when $\mathcal{Q}$ has a single point, for relatively small values of k.

Let L be a set of three concurrent lines and consider the six wedges defined by the lines. Bukh [6] proved that for any continuous measure there is a choice of L where each of the wedges has equal measure. Instead of a measure, we have V, a set of points in the plane where no three points are collinear (it is in *general position*), thus we need a generalisation where the wedges contain some specified number of points. In our generalisation, the *weight* of a wedge is given by the number of points from V assigned to it. If a point $v \in V$ lies in the interior of a wedge then we assign v to that wedge, and if v lies on the boundary of two or more wedges we assign v to one of them. (This assignment is not arbitrary, but we will do it is such a way as to obtain the desired number of points in each wedge.) We call this a *wedge assignment*. The weight of a half-plane is defined analogously.

▶ **Theorem 2.1.** *Let V be a set of n points in general position in the plane, where $n \geqslant 8$ is even. For any given $\alpha, \beta, \gamma \in \mathbb{N}$ such that $2\alpha + 2\beta + 2\gamma = n$, we can find a set of three concurrent lines that partitions the plane into six wedges such that there is a wedge assignment resulting in wedges whose weights are $\alpha, \beta, \gamma, \alpha, \beta, \gamma$ in counterclockwise order.*

Proof. Let $\ell(\theta)$ be the directed line making an angle θ with the positive x-axis that has exactly weight $n/2$ on either side of it, for a suitable assignment of points to the half-planes on either side of $\ell(\theta)$. Consider the line $\ell(\theta)$ for $\theta = 0$. For some point $z = (x, 0) \in \ell(\theta)$, consider the rays $\rho_1, \ldots, \rho_4$ emanating from z such that the six wedges defined by these rays and $\ell(0)$ have the desired number of voters; see Fig. 2. By varying θ and the point z, we can ensure that the rays $\rho_1, \ldots, \rho_4$ line up in such a way that, together with $\ell(\theta)$, they form three concurrent lines; see the full version for a complete proof. ◀

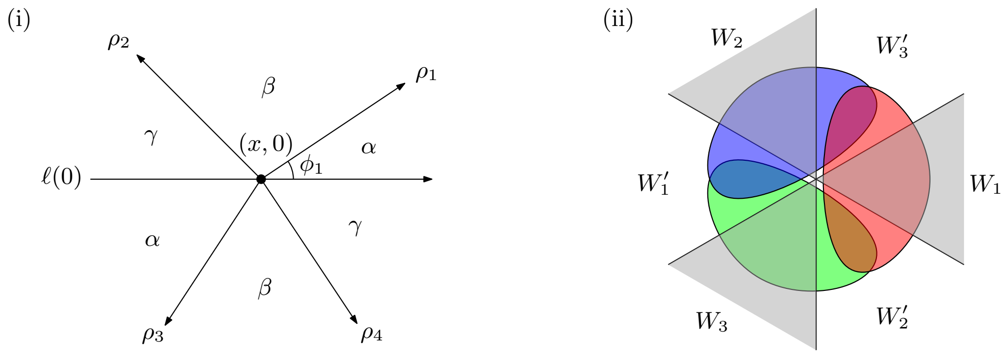


Figure 2 (i) Illustration for the proof of Theorem 2.1. (ii) Illustration for the proof of Theorem 2.3.

We also need the following easy-to-prove observation.

▶ **Observation 2.2.** *Let L be a set of three lines intersecting in a common point p^*, and consider the six closed wedges defined by L. Any convex set S not containing p^* intersects at most four wedges, and the wedges intersected by S are consecutive in the clockwise order.*

We now have all the tools to prove our new bounds on ε-nets for convex ranges. The guarantee they give is slightly weaker: where ordinarily placing an ε-net for n points means that a range not intersecting the ε-net can contain at most εn (and thus at most $\lfloor \varepsilon n \rfloor$) points, our *ceiling-based* ε-nets only guarantee that such a range contains at most $\lceil \varepsilon n \rceil$ points.

▶ **Theorem 2.3.** *Let ε_k be the smallest value such that any finite point set in $\mathbb{R}^2$ admits a weak ε_k-net of size k for convex ranges. Then for any set V of $n \geqslant 8$ points in general position, with n even, and any $r_1, r_2, s \in \mathbb{N}_0$, we can make a ceiling-based ε-net for V with*

$$\varepsilon = \frac{1}{2}\left(\frac{1}{\varepsilon_{r_1}} + \frac{2}{\varepsilon_{r_2}}\right)^{-1} + \frac{1}{2}\varepsilon_s.$$

Proof. Let $\mu := \frac{1}{2}\left(\frac{1}{\varepsilon_{r_1}} + \frac{2}{\varepsilon_{r_2}}\right)^{-1}$. We apply Theorem 2.1 with $\beta = \gamma = \left\lceil \frac{\mu}{\varepsilon_{r_2}} n \right\rceil$ and $\alpha = \frac{n}{2} - 2\beta$, which means $\alpha \leqslant \left\lfloor \frac{\mu}{\varepsilon_{r_1}} n \right\rfloor$, giving us a set L of three concurrent lines. To show that there exists a weak ceiling-based $(\varepsilon_{r_1+2r_2+3s+1})$-net N for V, number the wedges defined by L as $W_1, W_3', W_2, W_1', W_3, W_2'$ in clockwise order, as in Fig. 2(ii). Let $V_i \subset V$ and $V_i' \subset V$ be the subsets of points assigned to W_i and W_i', respectively. We can assume without loss of generality that $|V_1| = |V_1'| = \alpha$, and $|V_2| = |V_2'| = \beta$, and $|V_3| = |V_3'| = \gamma$. We add the following points to our net N: (i) the common intersection of the lines in L, denoted by p^*; (ii) an ε_{r_1}-net for V_1, an ε_{r_2}-net for V_2, and an ε_{r_2}-net for V_3; (iii) for each of the three collections of three consecutive wedges—these collections are indicated in red, green, and blue in Fig. 2(ii)—an ε_s-net. By construction, the size of our net N is $1 + r_1 + 2r_2 + 3s$. It can be shown that N is a ceiling-based $(\mu + \frac{1}{2}\varepsilon_s)$-net; see the full version. ◀

Note that $\varepsilon_0 = 1$, since if the net is empty, a range can contain all n points from V. Moreover, $\varepsilon_1 = 2/3$, and $\varepsilon_2 = 4/7$, and $\varepsilon_3 \leqslant 8/15$ by the results of Mustafa and Ray [10]. Using Theorem 2.3 we can obtain ceiling-based ε-nets with $k \geqslant 4$ points, by finding the best choice of r_1, r_2, s such that $k = r_1 + 2r_2 + 3s + 1$. This gives $\varepsilon_4 \leqslant \frac{1}{2}$, by setting $r_1, r_2 = 0$ and $s = 1$. Hence, for even n, player $\mathcal{P}$ can always place four points to win at least as many voters as player $\mathcal{Q}$, as opposed to the five that were proven in earlier work. Note that this also holds for $n \leqslant 8$, since then player $\mathcal{P}$ can simply pick four points coinciding with four of the at most eight voters. A similar statement holds for larger k when $n \leqslant 8$.

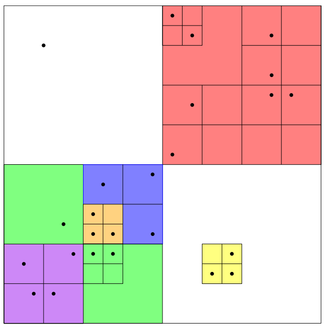


Figure 3 A compressed quadtree subdivision. The six regions generated for $m = 2$ are indicated in colour. Each region covers between three and eight voters, and one 'free' voter is left uncovered.

3 A quadtree-based strategy for player $\mathcal{P}$

In this section we sketch our quadtree-based strategy for $\mathcal{P}$, which gives good results when k is relatively large. A detailed description can be found in the full version.

The algorithm. First, we construct a *compressed quadtree* $\mathcal{T}$ on the voter set V. This gives a tree structure where each node ν is associated with a square or a donut region, which we denote by $\sigma(\nu)$. Donut regions in a compressed quadtree do not contain voters. We use the compressed quadtree to generate a set P of k points for player $\mathcal{P}$, as follows.

We pick a parameter m based on the maximum number of points we want to place and then we traverse the tree $\mathcal{T}$ in a bottom-up manner to generate a set $\mathcal{R}$ of regions containing between $m+1$ and $4m$ voters. Essentially, when we have collected the right number of voters in our bottom-up traversal, we add a region to $\mathcal{R}$ and otherwise we pass the (at most m) voters on to the parent node. Thus each region in $\mathcal{R}$ will be a quadtree cell $\sigma(\nu)$ minus the quadtree cells $\sigma(\nu')$ of certain nodes ν' in the subtree rooted at ν. An example of this is shown in Fig. 3. For each square $\sigma(\nu)$ corresponding to a region $R(\nu) \in \mathcal{R}$, we place the following 13 points for player $\mathcal{P}$: a grid of 3×3 points inside $\sigma(\nu)$, plus four points outside $\sigma(\nu)$ as in Fig. 4(i).

▶ **Lemma 3.1.** *The quadtree-based strategy places fewer than $13n/m$ points for player $\mathcal{P}$.*

An analysis of the number of voters player $\mathcal{Q}$ can win. To give an upper bound for the number of voters player $\mathcal{Q}$ can win, we consider the child regions of the regions in $\mathcal{R}$, which are themselves not in $\mathcal{R}$, as they have at most m voters. In the full version, we first show that player $\mathcal{Q}$ can win voters from at most five such child regions: the child region containing the point q played by $\mathcal{Q}$, and the child regions immediately above, below, to the

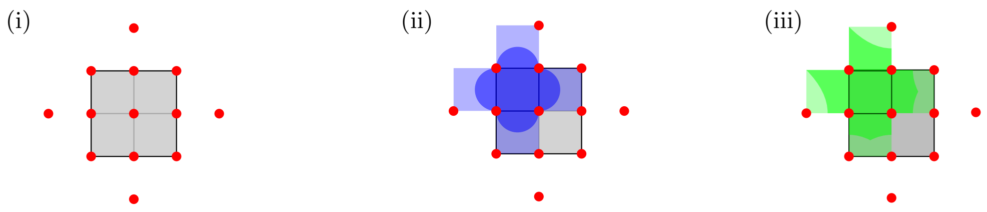


Figure 4 (i) The 13 points $P(R)$ placed around a region $R \in \mathcal{R}$. (ii) The set $Q(R')$ of the north-west child region R'. (iii) The set $V(R')$ of the north-west child region R'.

right, and to the left of q. We then continue to prove that $\mathcal{Q}$ can never win voters from all five regions simultaneously, but, in fact, from only three of them. The proofs are based on the following fact: Let R' be a child region of a region $R(\nu) \in \mathcal{R}$. Then, due to the 13 points we placed for each region in $\mathcal{R}$, the area $Q(R')$ from which a point $q \in R'$ might win voters has the shape shown in Fig. 4(ii). Moreover, the region $V(R')$ where q must be located to win voters inside R' has the shape shown in Fig. 4(iii). Since each child region contains at most $m < n/|\mathcal{R}|$ voters and $\mathcal{P}$ places $k \leqslant 13|\mathcal{R}|$ points, and $\mathcal{Q}$ can win voters from at most three child regions, we can conclude the following lemma.

▶ **Lemma 3.2.** *The quadtree-based strategy can place at most k points such that $\mathcal{P}$ wins at least $\left(1 - \frac{39}{k}\right) n$ voters against any single point placed by player $\mathcal{Q}$.*

A more refined strategy for player $\mathcal{P}$. It can be shown that the analysis presented above is tight. Hence, to get a better bound we need a better strategy. Recall that each region $R \in \mathcal{R}$ contains between $m + 1$ and $4m$ voters. In the strategy above, we use the same 13 points for any R, regardless of the exact number of voters it contains and how they are distributed over the child regions of R. Our refined strategy takes this into account, and also incorporates the ε-nets developed in the previous section; see the full version for details. This eventually gives the following theorem.

▶ **Theorem 3.3.** *Let V be a set of n voters in $\mathbb{R}^2$ in general position. For any given k, player $\mathcal{P}$ can win at least $\left(1 - \frac{20}{k}\right) n - 6$ voters by placing at most k points, against any single point placed by player $\mathcal{Q}$.*

4 Conclusion

We studied the discrete one-round Voronoi game where player $\mathcal{P}$ can place $k > 1$ points and player $\mathcal{Q}$ can place a single point. We improved the existing bounds on the number of voters player $\mathcal{P}$ can win in the L_2-metric. A challenging open problem is: Is it always possible for player $\mathcal{P}$ to win at least half the voters in the L_2-metric by placing fewer than four points?

References

1 Sayan Bandyapadhyay, Aritra Banik, Sandip Das, and Hirak Sarkar. Voronoi game on graphs. *Theoretical Computer Science*, 562:270–282, 2015. doi:https://doi.org/10.1016/j.tcs.2014.10.003.

2 Aritra Banik, Bhaswar B. Bhattacharya, and Sandip Das. Optimal strategies for the one-round discrete Voronoi game on a line. *J. Comb. Optim.*, 26(4):655–669, 2013. doi:10.1007/s10878-011-9447-6.

3 Aritra Banik, Jean-Lou De Carufel, Anil Maheshwari, and Michiel H. M. Smid. Discrete Voronoi games and ε-nets, in two and three dimensions. *Comput. Geom.*, 55:41–58, 2016. doi:10.1016/j.comgeo.2016.02.002.

4 Mark de Berg, Joachim Gudmundsson, and Mehran Mehr. Faster algorithms for computing plurality points. *ACM Trans. Algorithms*, 14(3):36:1–36:23, 2018. doi:10.1145/3186990.

5 Mark de Berg, Sándor Kisfaludi-Bak, and Mehran Mehr. On one-round discrete Voronoi games. In *Proc. 30th International Symposium on Algorithms and Computation (ISAAC 2019)*, volume 149 of *LIPIcs*, pages 37:1–37:17, 2019. doi:10.4230/LIPIcs.ISAAC.2019.37.

6 Boris Bukh. A point in many triangles. *Electron. J. Comb.*, 13(1), 2006. URL: http://www.combinatorics.org/Volume_13/Abstracts/v13i1n10.html.

7 Shuchi Chawla, Uday Rajan, R. Ravi, and Amitabh Sinha. Min-max payoffs in a two-player location game. *Oper. Res. Lett.*, 34(5):499–507, 2006. doi:10.1016/j.orl.2005.10.002.

8 Christoph Dürr and Nguyen Kim Thang. Nash equilibria in Voronoi games on graphs. In *Algorithms – ESA 2007*, pages 17–28, Berlin, Heidelberg, 2007. Springer Berlin Heidelberg.

9 Richard D. McKelvey and Richard E. Wendell. Voting equilibria in multidimensional choice spaces. *Mathematics of Operations Research*, 1(2):144–158, 1976.

10 Nabil H. Mustafa and Saurabh Ray. An optimal extension of the centerpoint theorem. *Comput. Geom.*, 42(6-7):505–510, 2009. `doi:10.1016/j.comgeo.2007.10.004`.

11 Sachio Teramoto, Erik D. Demaine, and Ryuhei Uehara. Voronoi game on graphs and its complexity. In *2006 IEEE Symposium on Computational Intelligence and Games*, pages 265–271, 2006. `doi:10.1109/CIG.2006.311711`.

12 Yen-Wei Wu, Wei-Yin Lin, Hung-Lung Wang, and Kun-Mao Chao. Computing plurality points and condorcet points in euclidean space. In *Proc. 24th International Symposium on Algorithms and Computation (ISAAC 2013)*, volume 8283 of *Lecture Notes in Computer Science*, pages 688–698. Springer, 2013. `doi:10.1007/978-3-642-45030-3\_64`.

On polynomials associated to Voronoi diagrams of point sets and crossing numbers

Mercè Claverol[1], Andrea de las Heras-Parrilla[1], David Flores-Peñaloza[2], Clemens Huemer[1], and David Orden[3]

1 **Universitat Politècnica de Catalunya**
`merce.claverol@upc.edu`, `andrea.de.las.heras@estudiantat.upc.edu`, `clemens.huemer@upc.edu`

2 **Facultad de Ciencias, Universidad Nacional Autónoma de México**
`dflorespenaloza@ciencias.unam.mx`

3 **Universidad de Alcalá**
`david.orden@uah.es`

Abstract

Three polynomials are defined for sets S of n points in general position in the plane: The Voronoi polynomial with coefficients the numbers of vertices of the order-k Voronoi diagrams of S, the circle polynomial with coefficients the numbers of circles through three points of S enclosing k points, and the $E_{\leq k}$ polynomial with coefficients the numbers of (at most k)-edges of S. We present several formulas for the rectilinear crossing number of S in terms of these polynomials and their roots. We also prove that the roots of the Voronoi polynomial lie on the unit circle if and only if S is in convex position. Further, we present bounds on the location of the roots of these polynomials.

1 Introduction

Let S be a set of $n \geq 4$ points in general position in the plane, meaning that no three points of S are collinear and no four points of S are cocircular. The Voronoi diagram of order k of S, $V_k(S)$, is a subdivision of the plane into cells such that points in the same cell have the same k nearest points of S. Voronoi diagrams have found many applications in a wide range of disciplines, see e.g. [7, 27]. We define the *Voronoi polynomial* $p_V(z) = \sum_{k=1}^{n-1} v_k z^{k-1}$, where v_k is the number of vertices of $V_k(S)$. Proximity information among the points of S is also encoded by the *circle polynomial* of S, which we define as $p_C(z) = \sum_{k=0}^{n-3} c_k z^k$, where c_k is the number of circles passing through three points of S that enclose exactly k other points of S. The numbers v_k and c_k are related via the well-known relation

$$v_k = c_{k-1} + c_{k-2} \tag{1}$$

where $c_{-1} = 0$ and $c_{n-2} = 0$, see e.g. [21]. The two polynomials $p_V(z)$ and $p_C(z)$ are especially interesting due to their connection to the rectilinear crossing number problem.

The rectilinear crossing number of a point set S, $\overline{cr}(S)$, is the number of pairwise edge crossings of the complete graph K_n when drawn with straight-line segments on S, i.e. the vertices of K_n are the points of S. Equivalently, $\overline{cr}(S)$ is the number of convex quadrilaterals with vertices in S. We denote $\overline{cr}(S)$ as $\alpha\binom{n}{4}$, with $0 \leq \alpha \leq 1$. Note that for S in convex position, $\alpha = 1$. The rectilinear crossing number problem consists in, for each n, finding the minimum value of $\overline{cr}(S)$ among all sets S of n points, no three of them collinear, commonly denoted as $\overline{cr}(K_n)$. The limit of $\overline{cr}(K_n)/\binom{n}{4}$, when n tends towards infinity, is the so-called *rectilinear crossing number constant* α^*. This problem is solved only for $n \leq 27$ and $n = 30$, and the current best bound for the rectilinear crossing number constant is $\alpha^* > 0,37997$, see

39th European Workshop on Computational Geometry, Barcelona, Spain, March 29–31, 2023.

the survey [3] and the web page [5]. A fruitful approach to the rectilinear crossing number problem is proving bounds on the numbers of j-edges and of $(\leq k)$-edges of S [1, 2, 6, 11, 20]. An (oriented) j-edge of S is a directed straight line ℓ passing through two points of S such that the open half-plane bounded by ℓ and on the right of ℓ contains exactly j points of S. The number of j-edges of S is denoted by e_j, and $E_{\leq k} = \sum_{j=0}^{k} e_j$ is the number of $(\leq k)$-edges. We then also consider the $E_{\leq k}$ *polynomial* $p_E(z) = \sum_{k=0}^{n-3} E_{\leq k} z^k$, which also encodes information on higher order Voronoi diagrams, since the number of j-edges e_j is the number of unbounded cells of the order-$(j+1)$ Voronoi diagram of S (see, e.g., Proposition 30 in [13]). Note that $p_E(z)$ has no term $E_{\leq n-2}$. For an illustration of the defined polynomials for a particular point set, see Figure 1.

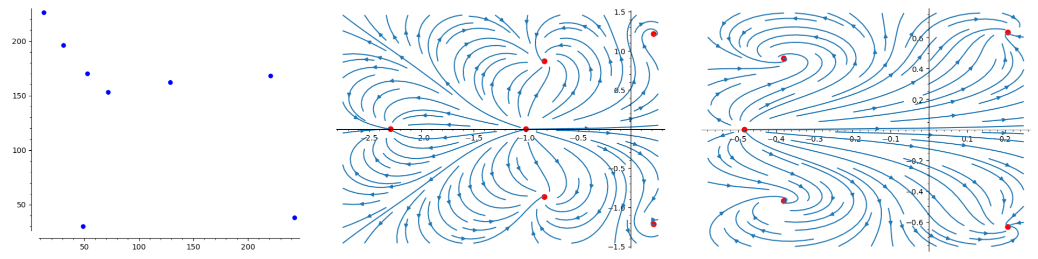


Figure 1 Left: 1591-th entry of the order type database for 8 points [4]. With complex stream plots of its Voronoi polynomial (center): $p_V(z) = 10 + 23z + 27z^2 + 24z^3 + 17z^4 + 9z^5 + 2z^6$, and its $E_{\leq k}$ polynomial (right): $p_E(z) = 4 + 13z + 22z^2 + 34z^3 + 43z^4 + 52z^5$; roots are red points.

For a point set S, we show that $\overline{cr}(S)$ appears in the first derivatives of these three polynomials when evaluated at $z = 1$ and, in addition, we obtain appealing formulas for $\overline{cr}(S)$ in terms of the roots of the polynomials. Motivated by this, we study the location of such roots, showing several bounds on their modulus. As a particular result, we also prove that the roots of the Voronoi polynomial lie on the unit circle if and only if S is in convex position. Furthermore, the circle polynomial comes into play when considering the random variable X that counts the number of points of S enclosed by the circle defined by three points chosen uniformly at random from S. The probability generating function of X is $p_C(z)/\binom{n}{3}$. In [24] a central limit theorem for random variables with values in $\{0, \ldots, n\}$ was shown, under the condition that the variance is large enough and that no root of the probability generating function is too close to $1 \in \mathbb{C}$. We show that the random variable X does not approximate a normal distribution, and use the result from [24] to derive that $p_C(z)$ has a root close to $1 \in \mathbb{C}$.

Throughout this work, points (a, b) in the plane are identified with complex numbers $z = a + ib$. To avoid cumbersome notation we omit indicating the point set S where it is clear from context; for example, each polynomial considered depends on a point set S but we write $p_C(z)$ instead of $p_C^S(z)$.

2 Known relations

A main source is the work by Lee [19], from where several of the following formulas can be obtained.

- For any point set S, and $0 \leq k \leq n-3$, it holds that, see [8, 12, 13, 19, 21],

$$c_k + c_{n-k-3} = 2(k+1)(n-k-2). \tag{2}$$

- From [15] we get the following two equations.

$$\sum_{k=0}^{n-3} k \cdot c_k = \binom{n}{4} + \overline{cr}(S) = (1+\alpha)\binom{n}{4}. \tag{3}$$

This was essentially also obtained in [29], though not stated in terms of $\overline{cr}(S)$.

$$\sum_{k=0}^{n-3} k^2 \cdot c_k = \binom{n}{5} + \binom{n}{4} + (n-3)\overline{cr}(S). \tag{4}$$

- For $k \leq \frac{n-3}{2}$ it holds that, see Lemma 3.1 in [14],

$$c_k \geq (k+1)(n-k-2), \text{ and } c_{n-k-3} \leq (k+1)(n-k-2). \tag{5}$$

Next Equations (6), (7) and (8) hold for a point set S in convex position.

$$2c_k = c_{k-1} + c_{k+1} + 2. \tag{6}$$

Then, the number of vertices of $V_k(S)$ fulfills, see e.g. Proposition 34, Equation (4) in [13],

$$v_k = c_{k-1} + c_{k-2} = (2k-1)n - 2k^2\ . \tag{7}$$

This implies that

$$v_k = v_{n-k}. \tag{8}$$

- For every set S of n points in general position, the relation between $E_{\leq k}$ and c_k is, see e.g. Proposition 33 in [13],

$$c_k + E_{\leq k} = (k+1)(2n-k-2). \tag{9}$$

3 Properties of the Voronoi, circle and $E_{\leq k}$ polynomials

For every set S of n points in general position:

▶ **Proposition 1.** *Polynomials $p_C(z) = \sum_{k=0}^{n-3} c_k z^k$, and $p_V(z) = \sum_{k=1}^{n-1} v_k z^{k-1}$ satisfy*

$$p_V(z) = (1+z)p_C(z). \tag{10}$$

▶ **Proposition 2.** *The circle polynomial $p_C(z) = \sum_{k=0}^{n-3} c_k z^k$ satisfies*

1. $p_C(1) = \binom{n}{3}$.
2. $p'_C(1) = \binom{n}{4} + \overline{cr}(S)$.
3. $p''_C(1) = \binom{n}{5} + (n-4)\overline{cr}(S)$.
4. $p_C(-1) = \frac{n-1}{2}$ *for n odd.*

▶ **Proposition 3.** *The Voronoi polynomial $p_V(z) = \sum_{k=1}^{n-1} v_k z^{k-1}$ satisfies*

1. $p_V(1) = 2\binom{n}{3}$.
2. $p'_V(1) = \binom{n}{3} + 2\binom{n}{4} + 2\overline{cr}(S)$.
3. $p''_V(1) = 2\binom{n}{4} + 2\binom{n}{5} + 2(n-3)\overline{cr}(S)$.
4. $p_V(-1) = 0$.
5. $p'_V(-1) = \frac{n-1}{2}$ *for n odd.*

▶ **Proposition 4.** *The $E_{\leq_k}$ polynomial $p_E(z) = \sum_{k=0}^{n-3} E_{\leq_k} z^k$ satisfies:*

1. $p_E(1) = 3\binom{n}{3}$.

2. $p'_E(1) = \sum_{k=0}^{n-3} kE_{\leq k} = 9\binom{n}{4} - \overline{cr}(S)$.
3. $p''_E(1) = \sum_{k=0}^{n-3} k(k-1)E_{\leq k} = 35\binom{n}{5} - (n-4)\overline{cr}(S)$.
4. $p_E(-1) = \frac{n(n-1)}{2}$ *for n odd.*

Using Lemma 1 of Aziz and Mohammad in [9], we get an intriguing family of formulas for the rectilinear crossing number.

▶ **Proposition 5.** *The coefficients of the polynomials $p_V(z)$, $p_C(z)$ and $p_E(z)$ satisfy*

$$\textbf{1)}\ \overline{cr}(S) = \frac{4}{3(n-3)} \sum_{k=1}^{n-3}\sum_{j=0}^{n-3} c_j \frac{z_k^{j+1}}{(z_k-1)^2} = \sum_{j=0}^{n-3} c_j \left(\frac{4}{3(n-3)} \sum_{k=1}^{n-3} \frac{z_k^{j+1}}{(z_k-1)^2} \right), \tag{11}$$

where the z_k are the $(n-3)$-th roots of -3.

$$\textbf{2)}\ \overline{cr}(S) = \frac{2}{3(n-1)} \sum_{k=1}^{n-1}\sum_{j=1}^{n-1} v_j \frac{z_k^{j}}{(z_k-1)^2} = \sum_{j=1}^{n-1} v_j \left(\frac{2}{3(n-1)} \sum_{k=1}^{n-1} \frac{z_k^{j}}{(z_k-1)^2} \right), \tag{12}$$

where the z_k are the $(n-1)$-th roots of -3.

$$\textbf{3)}\ \overline{cr}(S) = -\frac{4}{n-3} \sum_{k=1}^{n-3}\sum_{j=0}^{n-3} E_{\leq j} \frac{z_k^{j+1}}{(z_k-1)^2} = \sum_{j=0}^{n-3} E_{\leq j} \left(\frac{-4}{n-3} \sum_{k=1}^{n-3} \frac{z_k^{j+1}}{(z_k-1)^2} \right), \tag{13}$$

where the z_k are now the $(n-3)$-th roots of $-\frac{1}{3}$.

4 On the roots of the Voronoi, circle and $E_{\leq k}$ polynomials

In this section, we study properties for the roots of these polynomials. By Proposition 1, $p_V(z)$ has the same roots as $p_C(z)$ plus the root $z = -1$. A direct relation between roots of polynomials and the rectilinear crossing number can be derived from the well-known relation

$$\frac{P'(z)}{P(z)} = \sum_{i=1}^{n} \frac{1}{z - a_i}, \tag{14}$$

where $P(z)$ is a polynomial of degree n with roots $a_1, \ldots, a_n$, and z is any complex number such that $P(z) \neq 0$. For the circle polynomial $p_C(z)$ and $z = 1$, using Proposition 2 we get

$$\frac{\binom{n}{4} + \overline{cr}(S)}{\binom{n}{3}} = \sum_{i=1}^{n-3} \frac{1}{1 - a_i}, \tag{15}$$

where the a_i are the roots of $p_C(z) = \sum_{k=0}^{n-3} c_k z^k$. Note that 1 is never a root of a polynomial whose coefficients are all positive, as is the case in the polynomials introduced in this work. Using Equation (14) and the reciprocal polynomial $p_C^*(z) = \sum_{k=0}^{n-3} c_k z^{n-k-3}$ we obtain

▶ **Proposition 6.**

$$\sum_{k=0}^{n-3} (n-k-3)c_k = 3\binom{n}{4} - \overline{cr}(S). \tag{16}$$

▶ **Proposition 7.**

$$\frac{-2\binom{n}{4} + 2\overline{cr}(S)}{\binom{n}{3}} = \sum_{i=1}^{n-3} \frac{1 + a_i}{1 - a_i}, \tag{17}$$

where the a_i are the roots of $p_C(z) = \sum_{k=0}^{n-3} c_k z^k$ (also works for the roots of $p_V(z)$).

Of particular interest is $p_V(z) = \sum_{k=1}^{n-1} v_k z^{k-1}$ for a set of n points in convex position. By Equation (7), $v_k = (2k-1)n - 2k^2$. By Equation (8), $p_V(z)$ is a palindromic polynomial, so it has roots a_i and $1/a_i$. Then, for sets S of n points in convex position,

$$\sum_{i=1}^{n-2} \frac{1}{1-a_i} = \frac{n-2}{2}, \tag{18}$$

where the a_i are the roots of $p_V(z) = \sum_{k=1}^{n-1} v_k z^{k-1}$. For S in convex position, from Proposition 7 we also have,

$$\sum_{i=1}^{n-2} \frac{1+a_i}{1-a_i} = 0. \tag{19}$$

For our next result we use a theorem due to Malik [23], also see [28], Corollary 14.4.2.

▶ **Theorem 4.1.** *Let S be a set of points in general position. Then S is in convex position if and only if all the roots of the Voronoi polynomial of S, $p_V(z) = \sum_{k=1}^{n-1} v_k z^{k-1}$, lie on the unit circle.*

In order to find a lower bound on the largest modulus of the roots of $p_C(z)$ with S not in convex position, we use two theorems. The first one is due to Laguerre [18], Theorem 1, see also [25], and [28], Theorem 3.2.1b. The second theorem is due to Obrechkoff [26], also see [10] and [22], Chapter IX, 41, Exercise 5.

▶ **Theorem 4.2.** *For every set S of $n > 3$ points in general position with rectilinear crossing number $\overline{cr}(S) = \alpha \cdot \binom{n}{4}$, the Voronoi polynomial $p_V(z) = \sum_{k=1}^{n-1} v_k z^{k-1}$ has a root of modulus at least $1 + \frac{(1-\alpha)\pi^2}{16(n-3)^2} + O\left(\frac{1}{n^4}\right)$.*

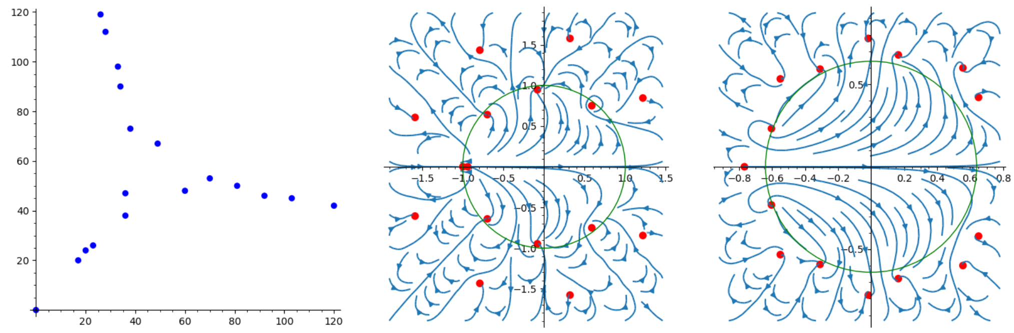


Figure 2 Left: Point set S minimizing the rectilinear crossing number for $n = 18$ [5]. With complex stream plots of its Voronoi (center), and $E_{\leq k}$ (right) polynomials; with roots as red points and circles illustrating, respectively, the bounds of Theorems 4.2 and 4.5.

For an illustration of Theorem 4.2 see Figure 2, center. We further show that the Voronoi polynomial $p_V(z)$ has a root close to point 1 in the complex plane. Thereto, we apply Theorem 1.2 from Michelen and Sahasrabudhe [24].

▶ **Theorem 4.3.** *Let α be a constant from $(0, 1]$ and let S be a set of n points in general position with $\frac{\overline{cr}(S)}{\binom{n}{4}} = \alpha$. Then the Voronoi polynomial of S, $p_V(z) = \sum_{k=1}^{n-1} v_k z^{k-1}$, has a root ζ such that $|1-\zeta| \in o\left(\frac{log(n)}{n}\right)$.*

In the following, we study the location of the roots of the $E_{\leq k}$ polynomial $p_E(z) = \sum_{k=0}^{n-3} E_{\leq k} z^k$ of a point set S. Note that its coefficients $E_{\leq k}$ form an increasing sequence of positive numbers. The well-known Eneström-Kakeya theorem [17] tells us that all the roots of $p_E(z)$ are contained in the unit disk, and more precisely, that they are contained in an annulus: The absolute values of the roots of $p_E(z)$ lie between the greatest and the least of

$$\frac{E_{\leq n-4}}{E_{\leq n-3}}, \frac{E_{\leq n-5}}{E_{\leq n-4}}, \ldots, \frac{E_{\leq 1}}{E_{\leq 2}}, \frac{E_{\leq 0}}{E_{\leq 1}}.$$

We give a lower bound on the largest modulus of the roots of the $E_{\leq k}$ polynomial.

▶ **Theorem 4.4.** *Let S be a set of $n > 3$ points in general position, with rectilinear crossing number $\overline{cr}(S) = \alpha \cdot \binom{n}{4}$. Then the $E_{\leq k}$ polynomial of S, $p_E(z) = \sum_{k=0}^{n-3} E_{\leq k} z^k$, has a root of modulus at least $\frac{3+\alpha}{9-\alpha}$.*

Next, we show a better lower bound on the largest modulus of $p_E(z)$ when n is large enough. Thereto, we apply a theorem of Titchmarsh ([31], p. 171), also see [16], Theorem A.

▶ **Theorem 4.5.** *Let S be a set of n points with h of them on the boundary of the convex hull of S, in general position. Then $p_E(z) = \sum_{k=0}^{n-3} E_{\leq k} z^k$ has a root of modulus at least*

$$\left(\frac{3\binom{n}{3}}{h} \right)^{-\frac{1}{n-3}}.$$

For an illustration of Theorem 4.5, see Figure 2, right.

5 Discussion

We have introduced three polynomials $p_V(z)$, $p_C(z)$, $p_E(z)$ for sets S of n points in general position in the plane, showing their connection to $\overline{cr}(S)$ and several bounds on the location of their roots. The obvious open problem is using bounds on such roots to improve upon the current best bound on the rectilinear crossing number problem. To the best of our knowledge, this approach has not been explored so far. Besides, we think that the presented polynomials are interesting objects of study on their own, given the many applications of Voronoi diagrams. For some of the formulas presented for one of the polynomials, like Equation (15), there are analogous statements for the other polynomials considered.

Further, several other polynomials on point sets can be considered. The reader interested in crossing numbers has probably in mind the j-edge polynomial $p_e(z) = \sum_{j=0}^{n-2} e_j z^j$ of a point set S. For this, the known formula for the rectilinear crossing number $\overline{cr}(S)$ in terms of the numbers of j-edges e_j of S, see [20], Lemma 5, translates into

$$2\overline{cr}(S) - 6\binom{n}{4} = p_e''(1) - (n-3)p_e'(1). \tag{20}$$

As is the case for the Voronoi polynomial and the circle polynomial, for sets S of n points in convex position, the j-edge polynomial $p_e(z)$ has all its roots on the unit circle. This is readily seen since $p_e(z)$ is then n times the all-ones polynomial, $p_e(z) = n\sum_{j=0}^{n-2} z^j$, as $e_j = n$ for all j, if S is in convex position. Its roots are the $(n-1)$-th roots of unity, except $z = 1$.

We finally propose to study the presented polynomials for random point sets. The expected rectilinear crossing number is known for sets of n points chosen uniformly at random from several convex shapes K, see e.g. [30], Section 1.4.5. pp. 63–64, and [3].

Acknowledgments. M. C., A. d.-P., C. H., and D. O. were supported by project PID2019-104129GB-I00/MCIN/AEI/10.13039/501100011033. C.H. was supported by project Gen. Cat. DGR 2021-SGR-00266. D. F.-P. was supported by grants PAPIIT IN120520 and PAPIIT IN115923 (UNAM, México).

References

1 B. M. Ábrego, M. Cetina, S. Fernández-Merchant, J. Leaños, G. Salazar, On $\leq k$-edges, crossings, and halving lines of geometric drawings of K_n. *Discrete and Computational Geometry*, 48, 192–215, 2012.

2 B. M. Ábrego, S. Fernández-Merchant, A lower bound on the rectilinear crossing number. *Graphs and Combinatorics*, 21, 293–300, 2005.

3 B. M. Abrego, S. Fernández-Merchant, G. Salazar, The rectilinear crossing number of K_n: Closing in (or are we?), *Thirty Essays on Geometric Graph Theory*, 5–18, Springer, 2012.

4 O. Aichholzer, `http://www.ist.tugraz.at/aichholzer/research/rp/triangulations/ordertypes/`, accessed 2022.

5 O. Aichholzer, `http://www.ist.tugraz.at/aichholzer/research/rp/triangulations/crossing/`, accessed 2022.

6 O. Aichholzer, J. García, D. Orden, P. Ramos, New lower bounds for the number of $(\leq k)$-edges and the rectilinear crossing number of K_n, *Discrete and Computational Geometry*, 38, 1–14, 2007.

7 F. Aurenhammer. Voronoi diagrams – a survey of a fundamental geometric data structure. *ACM Computing Surveys*, 23(3), 345–405, 1991.

8 F. Ardila, The number of halving circles, *The American Mathematical Monthly*, 111, 586–591, 2004.

9 A. Aziz, Q. G. Mohammad, Simple proof of a theorem of Erdős and Lax. *Proceedings of the American Mathematical Society*, 80, 119–122, 1980.

10 W. Bergweiler, A. Eremenko, Distribution of zeros of polynomials with positive coefficients, *Annales Academiae Scientiarum Fennicae Mathematica*, 40, 375–383, 2015.

11 J. Balogh, G. Salazar, k-sets, convex quadrilaterals, and the rectilinear crossing number of K_n, *Discrete and Computational Geometry* 35, 671–690, 2006.

12 K. L. Clarkson, P. W. Shor, Applications of random sampling in computational geometry, II, *Discrete and Computational Geometry*, 4, 387–421, 1989.

13 M. Claverol, A. de las Heras-Parrilla, C. Huemer, A. Martínez-Moraian, The edge labeling of higher order Voronoi diagrams. `https://arxiv.org/abs/2109.13002`, 2021.

14 M. Claverol, C. Huemer, A. Martínez-Moraian, On circles enclosing many points, *Discrete Mathematics*, 344(10), 112541, 2021.

15 R. Fabila-Monroy, C. Huemer, E. Tramuns, The expected number of points in circles, 28th European Workshop on Computational Geometry (EuroCG 2012), 69–72, 2012.

16 R. Gardner, B. Shields, The number of zeros of a polynomial in a disk, *Journal of Classical Analysis* 3, 167–176, 2013.

17 S. Kakeya, On the limits of the roots of an algebraic equation with positive coefficients, *Tôhoku Mathematical Journal* 2, 140–142, 1912.

18 M. Laguerre, Sur la résolution des équations numériques, *Nouvelles annales de mathématiques 2e série*, 17, 20–25, 1878.

19 D. T. Lee, On k-nearest neighbor Voronoi diagrams in the plane, *IEEE Transactions on Computers*, 31, 478–487, 1982.

20 L. Lovász, K. Vesztergombi, U. Wagner, E. Welzl, Convex quadrilaterals and k-sets. Towards a theory of geometric graphs, *Contemporary Mathematics*, 342, 139–148, 2004.

21 R. C. Lindenbergh, A Voronoi poset, *Journal for Geometry and Graphics*, 7, 41–52, 2003.

22 M. Marden, The geometry of polynomials, Mathematical Surveys and Monographs, Number 3, 1949. American Mathematical Society.

23 M. A. Malik, On the derivative of a polynomial, *Journal of the London Mathematical Society*, s2-1(1), 57–60, 1969.

24 M. Michelen, J. Sahasrabudhe, Central limit theorems and the geometry of polynomials. `https://arxiv.org/abs/1908.09020`, 2019.

25 J. von Sz. Nagy, Über einen Satz von Laguerre. Walter de Gruyter, Berlin/New York Berlin, New York, 1933.

26 N. Obrechkoff, Sur un problème de Laguerre, *Comptes Rendus* 177, 102–104, 1923.

27 A. Okabe, B. Boots, K. Sugihara, S. N. Chiu. Spatial Tessellations: Concepts and Applications of Voronoi diagrams, second edition, Wiley, 2000.

28 Q. I. Rahman, G. Schmeisser, Analytic theory of polynomials. London Mathematical Society Monographs New Series 26, Clarendon Press, Oxford, 2002.

29 J. Urrutia, A containment result on points and circles. Preprint, February 2004. `https://www.matem.unam.mx/~urrutia/online_papers/PointCirc2.pdf`

30 L. Santaló, Integral geometry and geometric probability, 2nd edition, Cambridge University Press, 2004.

31 E. C. Titchmarsh, The theory of functions, 2nd edition, Oxford University Press, London, 1939.

Density Approximation for Kinetic Groups

Max van Mulken[1], Bettina Speckmann[1], and Kevin Verbeek[1]

1 **Department of Mathematics and Computer Science, TU Eindhoven**
`[m.j.m.v.mulken|b.speckmann|k.a.b.verbeek]@tue.nl`

Related Version arXiv:2212.03685

1 Introduction

Sets of moving entities can form groups which travel together for significant amounts of time. Identifying and tracking groups is an important task in a variety of research areas, such as wildlife ecology, urban transport, or sports analysis. Consequently, in recent years various definitions and tracking algorithms have been proposed, such as herds [7], mobile groups [8], clusters [2, 9], and flocks [3, 13]. In computational geometry, there is a sequence of papers on variants of the *trajectory grouping structure* which allows a compact representation of all groups within a set of moving entities [4, 10, 14, 15, 16].

Not only the existence of one or more groups is an important fact to discover; in many application areas the actual shape of the group and the density distribution of the moving entities carries meaning as well. Consider, for example, the herd of wildebeest in Fig. 1. Both its global shape and the distribution of dense areas indicate that this herd is migrating. Research in wildlife ecology [6, 11] has established that animals often stay close together when not under threat and respond to immediate danger by spreading out. Hence from the density and the extent of a herd we can infer fear levels and external disturbances. The density distribution and general shape of a group are not only meaningful in wildlife ecology, but they can also provide useful insights when monitoring, for example, visitors of a festival.

In this paper we initiate the algorithmic study of the shape of a moving group. Specifically, we identify and track particularly dense areas which provide a meaningful first idea of the time-varying shape of the group. It is our goal to develop a solid theoretical foundation which will eventually form the basis for a software system that can track group shapes in real time. To develop an efficient algorithmic pipeline, we are making several simplifying assumptions on the trajectories of the moving objects (known ahead of time, piecewise linear, within a bounding box). In Section 5 we (briefly) explain how we intend to build further on our theoretical results to lift these restrictions, trading theoretical guarantees for efficiency.

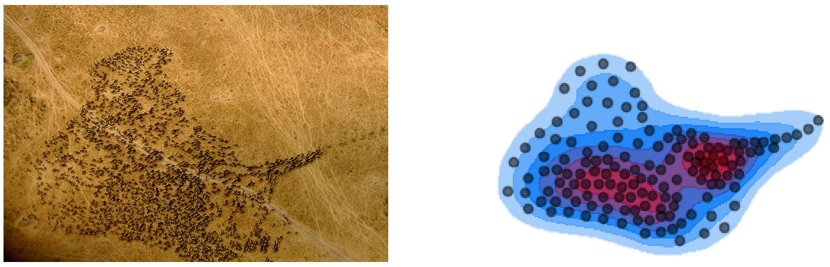


Figure 1 A herd of wildebeest, the shape of the group indicates migratory behavior.[1]

[1]https://commons.wikimedia.org/wiki/File:Wbeest_Mara.jpg on 25/11/2022.

Problem statement. Our input consists of a set P of n moving points in $\mathbb{R}^2$; we assume that the points follow linear motion and are contained in a bounding box $\mathcal{D} = [0, D] \times [0, D]$ with size parameter D. The position of a point $p \in P$ over time is described by a function $p(t)$, where $t \in [0, T]$ is the time parameter. We often omit the dependence on t, and simply denote the position of a point as p. We assume that the set P continuously forms a single group; we aim to monitor the density of P over time. We measure the density of P at position (x, y) using the well-known concept of *kernel density estimation* (KDE) [12]. KDE uses *kernels* of some width σ around each point $p \in P$ to construct a function $\text{KDE}_P \colon \mathbb{R}^2 \to \mathbb{R}^+$ such that $\text{KDE}_P(x, y)$ estimates the density of P at position (x, y). We are mainly interested in how the density peaks of P, i.e., the local maxima of the function KDE_P, change over time. Note that not all local maxima are equally relevant, since some are minor "bumps" caused by noise. Our approach is guaranteed to track the most significant local maxima through time, but may also track other maxima. We measure the significance of a local maximum using the concept of *topological persistence*.

Approach and organization. We could simply attempt to maintain the entire function KDE_P over time and additionally keep track of its local maxima. However, doing so would be computationally expensive. Furthermore, there are good reasons to approximate KDE_P: (1) KDE is itself also an approximation of the density, and (2) approximating KDE_P may eliminate local maxima that are not relevant. For a simple approach, consider building a quadtree on P. Observe that in areas where P is dense, the quadtree cells will be relatively small. Hence, we can associate the size of a quadtree cell with the density of P. This approach has the added advantage that the spatial resolution near the density peaks is higher. However, there are two main drawbacks: (1) the approximation does not depend on the chosen kernel size for the KDE, which is an important parameter for analysis, and (2) we cannot guarantee that all significant local maxima of KDE_P are preserved.

The approach we present in this paper builds on the simple approach described above, but eliminates its drawbacks. We first introduce how to approximate any function f using a quadtree based on the volume underneath f, which is described in more detail in Section 2. Next, in Section 3, we shift our focus to the function KDE_P, and describe how to construct a point set to estimate the volume underneath KDE_P. This will allow us to efficiently maintain the approximation as the underlying points move. In Section 4 we describe how to maintain the approximation of KDE_P, as well as its maxima, as the underlying points move using a kinetic data structure. Lastly, in Section 5 we reflect upon our approach and sketch how our theoretical results can serve as a guide towards a solution that is efficient in practice.

2 Volume-based quadtree

We first introduce the approximation of a function $f \colon [0, D]^2 \to \mathbb{R}^+$ using a volume-based quadtree T, where D is the size of the domain. Specifically, instead of subdividing a cell in the quadtree when it contains more than one point, we subdivide a cell in the quadtree if the volume under the function f contained within the cell exceeds some pre-specified threshold value ρ. Furthermore, we assign a single function value to each leaf cell of the quadtree corresponding to the average value of f within the cell. This results in a 2-dimensional step function f_T that approximates f (see Fig. 2). In the following we assume, w.l.o.g., that the total volume under f is 1.

Our goal is to pick threshold value ρ such that we can prove that f_T has small additive error with respect to f. This does not hold for all functions f, but only for functions that

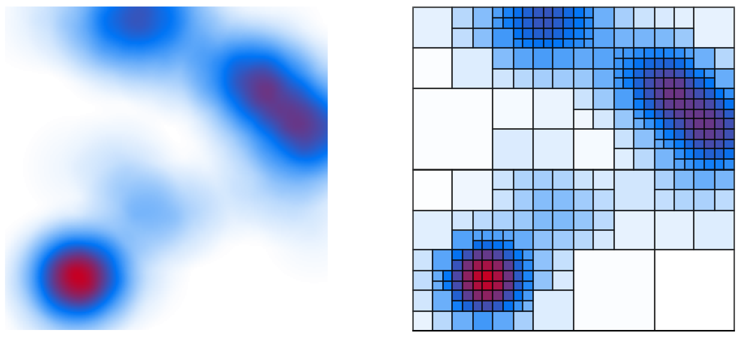


Figure 2 A 2-dimensional function (left) approximated as a step function (right).

are Lipschitz continuous[2]. Thus, we also use the Lipschitz constant λ to express our bounds. We summarize these results in the following theorem:

▶ **Theorem 2.1.** *Let T be the volume-based quadtree of $f : [0, D]^2 \to \mathbb{R}^+$ with threshold ρ. Then T has at most $O\left(\frac{1}{\rho}\log\left(\frac{D}{\sqrt{\rho/z^*}}\right)\right)$ nodes in total, where z^* is the maximum value of f. Additionally, for any cell $v \in T$ we have that $|f(x, y) - f_T(x, y)| \le \min\left(\frac{2\sqrt{2}}{3}\lambda s(v), \sqrt[3]{6\lambda^2\rho}\right)$ for all $(x, y) \in v$, where $s(v)$ is the side length of v and λ is the Lipschitz constant of f.*

Proof sketch. We first obtain the minimum side length of any cell v in T using ρ and z^*. By construction, the parent w of v in T must contain a total volume of at least ρ. Since the maximum function value of f is z^*, the side length of w must be at least $\sqrt{\rho/z^*}$, and hence the side length of v is at least $\frac{1}{2}\sqrt{\rho/z^*}$. This directly implies that the depth of T is as most $\log\left(\frac{2D}{\sqrt{\rho/z^*}}\right)$. Next, observe that removing all leaves from T results in a tree that has at most $\frac{1}{\rho}$ leaves (since they are interior disjoint). Combined with the maximum depth, this gives us the size bound on T in the theorem statement.

To obtain the error bound we observe that, given any $(x, y) \in [0, D]^2$ for which $f(x, y) = z$, the cone with its apex at (x, y, z) and with slope λ must lie fully underneath f. Given any cell $v \in T$, since the value of f_T in v is set to the average value of f in v, we must have $f(x_v, y_v) = f_T(x_v, y_v)$ for some $(x_v, y_v) \in v$. Therefore, we can use the cone centered at (x_v, y_v) to bound how much volume is in v, bounding the function value of f_T. ◀

3 From volume to points

We aim to maintain a volume-based quadtree for KDE_P over time. We scale KDE_P such that the volume underneath KDE_P is 1. Additionally, we assume the kernel width σ to be 1. This implies that, for most kernels, both the Lipschitz constant and the maximum value are also bounded by a small constant (see for example the kernel in Fig. 3 (right)).

We can now use the result from Section 2 to approximate KDE_P with a volume-based quadtree. However, doing so would require us to maintain the volume under KDE_P as the underlying points move, which is not very efficient. We therefore approximate this volume using a discrete set of points. We do so in two steps: (1) we approximate the

[2]A function is Lipschitz continuous if there is some Lipschitz constant λ such that λ is the maximum absolute slope of f in any direction.

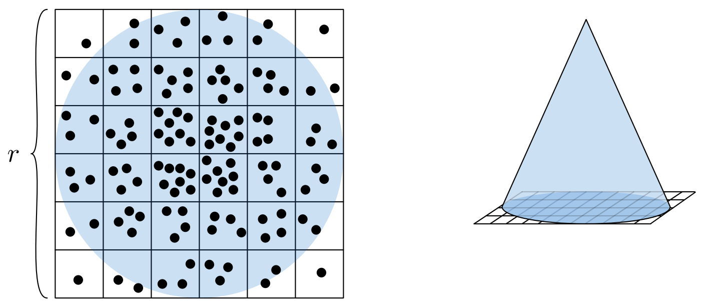


Figure 3 An example of how a cone kernel can be approximated by a point set. In each grid cell, $\lceil rz(c) \rceil$ points are arbitrarily placed.

volume under each kernel individually, and (2) we combine these kernel approximations to obtain an approximation for the volume under KDE_P. We use the concepts of *coresets* and *ε-approximations* to compute a small set of points Q that can accurately approximate the volume under KDE_P. We choose the points in Q such that we do not have to update Q as long as the original points in P do not change their trajectories. We can, however, update Q efficiently when a trajectory changes.

Let K denote the kernel function. We approximate the volume under K for each point $p \in P$ using a point set S_p. To obtain S_p, we consider a regular $r \times r$ grid G on the domain $[-1, 1]^2$, for some value r to be chosen later. We construct a *grid-based sampling* $S_p(r)$ of K by arbitrarily placing $\lceil rz(c) \rceil$ points in every cell $c \in G$, where $z(c)$ is the average value of K in the grid cell c. See Fig. 3 for an example. By carefully choosing r we can obtain, for a chosen error $\varepsilon_{\text{dsc}} > 0$, a point set S_p consisting of $O(\frac{1}{\varepsilon^3_{\text{dsc}}})$ points such that we have $\left| \frac{|S_p(r) \cap R|}{|S_p(r)|} - V_R(K) \right| \leq \varepsilon_{\text{dsc}}$ for any square region $R \subseteq [0, D]^2$, where $V_R(K)$ denotes the volume under K restricted to region R.

Now, we construct such grid-based samplings S_p around each point $s \in P$, and we assign points in S_p the same movement direction as p. The complete set of points $\mathcal{S} = \bigcup_{p \in P} S_p$ now provides an approximation for the volume under KDE_P with error at most ε_{dsc} for all times t. We can now use a result by Agarwal *et al.* [1] to construct an ε-approximation Q of $\mathcal{S}$. In the remainder of this paper we will simply refer to the set of linearly moving points Q as the coreset of KDE_P, where the additive error with respect to the volume is ε_{cor}. We summarize the result in the following theorem:

▶ **Theorem 3.1.** *Let KDE_P be a KDE function on a set of n linearly moving points P. For any $\varepsilon_{cor} > 0$, we can construct a coreset Q of linearly moving points such that, at any time and any square region $R \subseteq [0, D]^2$, we get that $\left| \frac{|Q \cap R|}{|Q|} - V_R(KDE_P) \right| < \varepsilon_{cor}$, where $V_R(KDE_P)$ is the volume under KDE_P restricted to R. Q consists of $O(\frac{1}{\varepsilon^2_{cor}} \log \left(\frac{1}{\varepsilon_{cor}} \right))$ points and can be constructed in $O(n \operatorname{poly} \left(\frac{\log n}{\varepsilon_{cor}} \right))$ time.*

We approximate the volume-based quadtree T of KDE_P using the *weight-based quadtree* $\widetilde{T}$ of Q, in which we subdivide a node $v \in \widetilde{T}$ when the fraction of points from Q that fall in the region corresponding to v exceeds ρ (instead of the actual volume). We can alter Theorem 2.1 slightly to give similar bounds on the size and error of the weight-based quadtree $\widetilde{T}$, based on λ, ρ, and the coreset error ε_{cor}. This way, for any error $\varepsilon > 0$, we can set $\rho = \Theta(\varepsilon^3)$ and $\varepsilon_{\text{cor}} = \Theta(\varepsilon^4)$ to obtain that $|\text{KDE}_P(x, y) - f_{\widetilde{T}}(x, y)| < \varepsilon$ for every time t.

4 KDS for density approximation

We now transformed the problem of maintaining local maxima of KDE_P to the problem of maintaining a quadtree on a set of moving points. We present a simple *kinetic data structure* (KDS) that maintains the weight-based quadtree and the function $f_{\widetilde{T}}$, as well as its local maxima. Note that the local maxima of $f_{\widetilde{T}}$ correspond at least to the local maxima of KDE_P that have persistence 2ε or more [5].

The kinetic data structure to store $\widetilde{T}$ and its local maxima is relatively simple. For each cell in T_{vol}, we store whether it is a local maximum or not. Note that a local maximum is defined as a cell that is higher than all of its spatial neighbors. We can show that any cell that has a spatial neighbor that is three or more levels deeper in $\widetilde{T}$ can never be a local maximum. Therefore, in order to be able to update efficiently whether a node is a local maximum, we store a set of pointers in each node $v \in \widetilde{T}$ that points to all spatial neighbors $w \in \widetilde{T}$ of v such that v and w differ in depth by at most 2.

We trigger an event only when a point from Q crosses a boundary of a cell in $\widetilde{T}$. Then, we must locally recompute $\widetilde{T}$, as well as which cells in $\widetilde{T}$ correspond to local maxima. Using the pointers described in the previous paragraph, this can be done efficiently. We summarize the result in the following theorem.

▶ **Theorem 4.1.** *Let $f = KDE_P$ be a KDE function on a set P of n linearly moving points in $[0, D]^2$. For any $\varepsilon > 0$, there exists a KDS that approximately maintains the local maxima of f with persistence at least 2ε. The KDS can be initialized in $O\left(n \operatorname{poly}\left(\frac{\log n}{\varepsilon}\right)\right)$ time, processes at most $O\left(D \operatorname{poly}\left(\frac{1}{\varepsilon}\right)\right)$ events, and can handle events and flight plan updates in $O\left(\log D + \operatorname{poly}\left(\frac{1}{\varepsilon}\right)\right)$ and $O\left(\operatorname{poly}\left(\frac{\log n}{\varepsilon}\right)\right)$ time, respectively.*

The coreset Q only needs to change when a point changes its trajectory, which we can also handle efficiently using the data structure by Agarwal *et al.* [1]. Although we only prove results on maintaining persistent local maxima, our approach actually maintains an approximation of the KDE function, and hence could also be used to track other shape features on the density of P.

5 Discussion

We believe that approximating the density surface we want to maintain via a suitable coreset of moving points is a promising direction also in practice. Below we briefly sketch how to handle our various input restrictions.

First of all, we consider the bounding box restriction. Various bounds on the quadtree complexity and the KDS quality measures depend on the size of the domain D. As we assume that our input points represent a single group, it makes sense to assume that the kernel functions of any point (its region of influence) must overlap with the kernel function of at least one other point. Since we scale the input such that the kernel width is $\sigma = 1$, this directly implies that $D = O(n)$ for a static set of points, although it is likely much smaller. However, when points move in a single direction for a long time (say, when a herd is migrating), they may easily leave a domain of that size. To address this problem without blowing up the size of the domain, we can move the domain itself along a piecewise-linear trajectory. A change of direction of the domain directly changes the trajectories of all points, and all events in the KDS must be recomputed. The coreset, however, does not need to change during such an event. We can limit the number of domain flight plan changes by using a slightly larger domain than needed at any point in time.

Second, we discuss the assumption that the trajectories are piecewise linear and known ahead of time. For the coreset to exist, the range space formed by the trajectories of the (samples around) the input objects and a set of square regions needs to have bounded VC-dimension. We proved an upper bound on this VC-dimension in the case that all trajectories are linear. Real-world animal trajectories are certainly not linear. However, since animals cannot move at arbitrary speeds and subgroups can often be observed to stay together, we still expect the corresponding range space to have bounded VC-dimension. A formal proof seems out of reach for more than very restrictive motion models, but bounds might be deduced from experimental data.

The actual computation of the coreset via the algorithm of Agarwal *et al.* [1] is impossible if the trajectories are not known ahead of time. However, random sampling (that is, sample a point $p \in P$, and then sample from its kernel) can be expected to result in a coreset of good size and quality in practice (the worst-case bounds on the size of the coreset which we proved are unlikely to be necessary in practice). Since we generally do not know the trajectories of the animals, but we do have bounds on their maximum speeds, a black-box KDS could be used to maintain such a random sampling coreset efficiently.

Our theoretical results inform the direction of our future engineering efforts in two ways. First of all, we now know that we can approximate well with a coreset whose size depends only on the desired approximation factor and not on the input size. Second, we know how to sample to find such a coreset, by constructing randomly shifted copies of the input points.

References

1 Pankaj Agarwal, Mark de Berg, Jie Gao, Leonidas Guibas, and Sariel Har-Peled. Staying in the Middle: Exact and Approximate Medians in R1 and R2 for Moving Points. In *Proceedings of the 17th Canadian Conference on Computational Geometry*, pages 43–46, 01 2005.

2 Gennady Andrienko and Natalia Andrienko. Interactive cluster analysis of diverse types of spatiotemporal data. *ACM SIGKDD Explorations*, 11(2):19–28, 2010.

3 Marc Benkert, Joachim Gudmundsson, Florian Hübner, and Thomas Wolle. Reporting flock patterns. *Computational Geometry*, 41(3):111–125, 2008.

4 Kevin Buchin, Maike Buchin, Marc van Kreveld, Bettina Speckmann, and Frank Staals. Trajectory grouping structure. *Journal of Computational Geometry*, 6(1):75–98, 2015.

5 Kevin A. Buchin, Max J.M. van Mulken, Bettina Speckmann, and Kevin A.B. Verbeek. Kinetic group density in 1d. In *Proceesings of the 38th European Workshop on Computational Geometry*, pages 47:1–47:6, 2022.

6 Jasper Eikelboom. *Sentinel animals: Enriching artificial intelligence with wildlife ecology to guard rhinos*. PhD thesis, Wageningen University, 2021.

7 Yan Huang, Cai Chen, and Pinliang Dong. Modeling herds and their evolvements from trajectory data. In *Proceedings of the 5th International Conference on Geographic Information Science*, pages 90–105, 2008.

8 San-Yih Hwang, Ying-Han Liu, Jeng-Kuen Chiu, and Ee-Peng Lim. Mining mobile group patterns: A trajectory-based approach. In *Proceedings of the 9th Conference on Advances in Knowledge Discovery and Data Mining*, pages 713–718, 2005.

9 Panos Kalnis, Nikos Mamoulis, and Spiridon Bakiras. On discovering moving clusters in spatio-temporal data. In *Proceedings of the 9th International Conference on Advances in Spatial and Temporal Databases*, pages 364–381, 2005.

10 Irina Kostitsyna, Marc van Kreveld, Maarten Löffler, Bettina Speckmann, and Frank Staals. Trajectory grouping structure under geodesic distance. In *Proceedings of the 31st International Symposium on Computational Geometry*, pages 674–688, 2015.

11 Anders Nilsson. Predator behaviour and prey density: evaluating density-dependent intraspecific interactions on predator functional responses. *Journal of Animal Ecology*, 70(1):14–19, 2001.

12 Emanuel Parzen. On Estimation of a Probability Density Function and Mode. *The Annals of Mathematical Statistics*, 33(3):1065 – 1076, 1962.

13 Craig Reynolds. Flocks, herds and schools: A distributed behavioral model. In *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques*, pages 25–34, 1987.

14 Arthur van Goethem, Marc van Kreveld, Maarten Löffler, Bettina Speckmann, and Frank Staals. Grouping time-varying data for interactive exploration. In *Proceedings of the 32nd International Symposium on Computational Geometry*, pages 61:1–61:16, 2016.

15 Marc van Kreveld, Maarten Löffler, Frank Staals, and Lionov Wiratma. A refined definition for groups of moving entities and its computation. *International Journal of Computational Geometry & Applications*, 28(02):181–196, 2018.

16 Lionov Wiratma, Marc van Kreveld, Maarten Löffler, and Frank Staals. An experimental evaluation of grouping definitions for moving entities. In *Proceedings of the 27th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 89–98, 2019.

On the Size of Fully Diverse Sets of Polygons using the Earth Movers Distance or Wasserstein Distance*

Fabian Klute[1,2] **and Marc van Kreveld**[2]

1 **Polytechnic University of Catalonia, Barcelona, Spain**
`fmklute@gmail.com`

2 **Department of Information and Computing Sciences, Utrecht University, Utrecht, The Netherlands**
`m.j.vankreveld@uu.nl`

Abstract

Diversity is a property of sets that shows how varied or different its elements are. We define full diversity in a metric space and study the maximum size of fully diverse sets. A set is *fully diverse* if each pair of elements is as distant as the maximum possible distance between any pair, up to a constant factor. In a previous paper (Klute and van Kreveld, On Fully Diverse Sets of Geometric Objects and Graphs, WG 2022), we showed how large a fully diverse set of simple polygons inside a unit-diameter region can be when we assume the Hausdorff distance, the Fréchet distance, or the area of symmetric difference to determine the distance between two simple polygons. In this paper we extend these results by considering the Earth Movers Distance and the Wasserstein Distance: we show that the maximum size of fully diverse sets is $O(1)$ in both cases. When we restrict ourselves to convex polygons, the Fréchet distance and area of symmetric difference also allow only $O(1)$ such polygons in a fully diverse set, unlike the simple polygon case for these measures.

Related Version

1 Introduction

The process of generating data, for example for benchmarks, may require that the resulting data is sufficiently diverse. Another occurrence is the creation of a layout or configuration from various options. For example, in graph drawing this observation has led to systems that present several drawings and then let the user choose a drawing or indicate preference, allowing for the creation of more drawings aligning with the preferred ones [2].

In a recent paper [10] we introduced a framework that allows us to study diversity of "objects", and analyze the maximum number of objects that are pairwise far apart. This framework is applicable in many contexts.

Diversity as a counting problem. We define full diversity as in [10]. Let $(\mathcal{S}, \mu)$ be a metric space where $\mathcal{S}$ is a base set or class of objects and μ is a distance measure that assigns a distance to any pair from $\mathcal{S}$. We consider the case where $\mu(a,b)$ is bounded for all $a, b \in \mathcal{S}$; let $M = \sup_{a,b\in\mathcal{S}} \mu(a,b)$ be the highest value that is attained (possibly in the limit) by μ on $\mathcal{S}$.

* Supported by the Netherlands Organisation for Scientific Research (NWO) under project no. 612.001.651 and the Austrian Science Foundation (FWF) grant J4510.

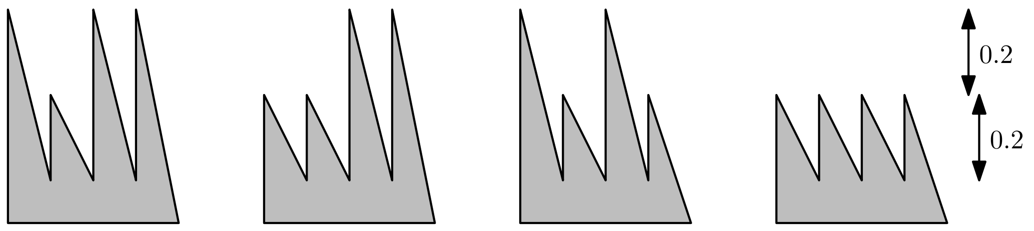


Figure 1 Fully diverse set of n-vertex simple polygons that each fit inside a unit-diameter region. Every pair gives a Fréchet distance of at least 0.2 if at any place, one polygon has a high peak where the other has a low peak. We can have $2^{\lfloor n/2 \rfloor}$ such simple polygons in a fully diverse set.

Table 1 Bounds on the maximum size of fully diverse sets in various metric spaces with geometric measures. The descriptive complexity of the objects is n and U denotes a unit-diameter disk.

Object	Metric	Space	Diameter	Lower bound	Upper bound	Status
Polygons	Hausdorff distance	U	$\Theta(1)$	$\Omega(1)$	$O(1)$	[10]
Polylines	Fréchet distance	U	$\Theta(1)$	$2^{\Omega(n)}$	$2^{O(n)}$	[10]
Polygons	Area symm. diff.	U	$\Theta(1)$	$2^{\Omega(n)}$	$2^{O(n \log n)}$	[10]
Point sets	Matching distance	U	$\Theta(n)$	$\Omega(1)$	$O(1)$	new
Polygons	Wasserstein distance	U	$\Theta(1)$	$\Omega(1)$	$O(1)$	new

▶ **Definition 1.** For a given $c \geq 1$, a subset $\hat{\mathcal{S}} \subseteq \mathcal{S}$ is called $\frac{1}{c}$-*diverse* if for all $x, y \in \hat{\mathcal{S}}$, we have $\mu(x,y) \geq \frac{1}{c} \cdot M$. If c can be chosen constant, independent of $|\hat{\mathcal{S}}|$, then $\hat{\mathcal{S}}$ is called *fully diverse.*

Intuitively, we relate the distance of all pairs of the subset to the maximum distance within the base set. Note that the metric space must be bounded to make this definition meaningful. We are interested in the question how large fully diverse subsets can be. As a simple example, consider all points in a unit square region in the plane and Euclidean distance as the metric. Then the maximum distance is $\sqrt{2}$, so a $\frac{1}{c}$-diverse (sub)set of points must have pairwise distances of at least $\sqrt{2}/c$. It is easy to see that any $\frac{1}{c}$-diverse set has size $O(c^2)$ by a packing argument, and any maximal $\frac{1}{c}$-diverse set has size $\Omega(c^2)$. The maximum size of a fully diverse set of points is $\Theta(c^2) = \Theta(1)$ if c is a constant.

In our recent paper [10] we studied more complex objects like simple polygons, embeddings of graphs, and graphs themselves, where we need to choose an appropriate distance measure. In this paper we extend upon the geometric results. We assume that the objects reside in a bounded space to make the metric space bounded. Let U be a unit-diameter disk in the plane in which our objects reside. Results from [10] and new results are shown together in Table 1. A construction for the lower bound for the Fréchet distance is shown in Figure 1.

The known results include the result that the metric space of n-vertex simple polygons has a large diversity if distance is measured by area of symmetric difference. Our main new result shows that this is not the case for Earth Movers Distance or Wasserstein distance. The situation that the distance over which weight or area is moved counts in the measure, "removes the large diversity" in the metric space.

Intuitively, convex polygons are much less diverse than simple polygons. However, for the Hausdorff distance and the Earth Movers Distance this does not show, because even simple polygons allow only constant-size fully diverse sets. This paper shows that the Fréchet distance and area of symmetric difference also give only $O(1)$ size fully diverse sets of convex polygons in a unit-diameter region.

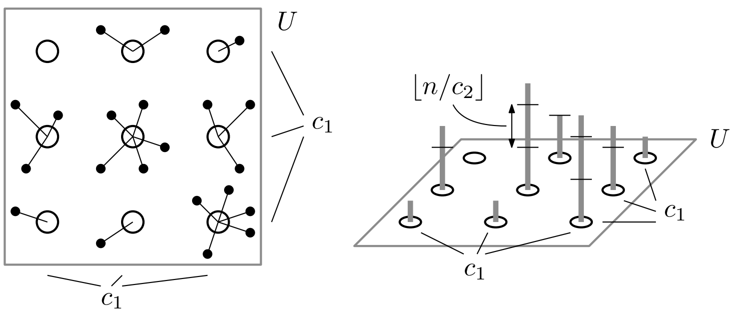


Figure 2 Snapping the n points to c_1^2 grid points (left), and making batches at the grid points to approximate the number of points snapped to the same grid point (right).

Relation to diversity and similar notions in science. Diversity has been studied in a variety of scientific contexts. One well-known example is in ecosystems, specifically, the diversity of species that are represented in a sample of animals or plants, see for instance [8, 13]. The Shannon index is commonly used, also known as Shannon entropy in information theory.

In computer science, diversity has been studied in a variety of areas. For example, the diversity of the output in selection tasks in big data [6] or recommender systems [12], of input data sets for machine learning [11], or of colored point sets in computational geometry [15].

Diversity without a priori assigned categories is of interest in the study of the diversity of a population in genetic algorithms, e.g. [16]. Following similar ideas, researchers later studied the diversity of sets of solutions in satisfiability problems [9], multicriteria optimization problems [14], and, recently, parameterized algorithms, e.g. [3]. Hebrard et al. [7] introduced the maximization problem to find the set of solutions that maximizes this sum or minimum distance over all sets of solutions.

2 Diversity of n-point Sets using Earth Movers Distance

We consider the matching distance for sets of n points inside a unit diameter disk U. This is also the Earth Movers Distance between two sets of n points with unit weights, but the Earth Movers Distance is more general. We first discuss the matching distance and then the trivial adaptation for the Earth Movers Distance.

Assume that a set $\mathcal{S}$ of n-point sets is fully diverse, giving us the existence of a constant $c \geq 1$. We will discretize any n-point set in such a way that any two n-point sets that have the same discretization have distance $< n/c$ (so they cannot be together in a fully diverse set), and the number of different discretizations is constant (a function of c but not of n).

Let $c_1, c_2 \geq 1$ be two constants. We generate a regular square grid with spacing $1/c_1$, so that we get at most c_1^2 grid points inside U, see Figure 2. Let P be an n-point set. We snap all points of P to their nearest grid point. Let n_g be the number of points of P snapped to a grid point g. We discretize this number as well, at every grid point. To this end we make batches of $\lfloor n/c_2 \rfloor$ points and count the number of batches that is full. The total discretization implies that there are $c_2^{c_1^2}$ different discretized sets, since there are at most c_2 choices for each of the c_1^2 grid points.

Let P and Q be two n-point sets with the same discretization. We analyze how we must choose c_1 and c_2 so that P and Q necessarily have distance less than n/c. The total snapping distance is at most $\sqrt{2}n/c_1$ for P and for Q. The total cost of matching the points of P and Q not in a full batch is upper-bounded by $c_1^2 n/c_2$, as there are at most this many points not in a full batch and the matching distance of any pair of points is at most 1. So an upper bound on the matching cost of P and Q is $2\sqrt{2}n/c_1 + c_1^2 n/c_2$. If we choose $c_1 > 4\sqrt{2}c$ and $c_2 > c_1^3/(2\sqrt{2})$, then both terms are less than $n/(2c)$, so the matching cost of P and Q is less than n/c. In other words, a fully diverse set of n-point sets must contain point sets with different discretizations, and there are only $O(1)$ discretizations. In particular, there are at most $c_2^{c_1^2} = (77c^3)^{36c^2} = c^{O(c^2)} = O(1)$ discretizations if we choose $c_1 = 6c$ and $c_2 = 77c^3$.

The reasoning above holds without any change for two point sets with total weight n each, regardless of the number of points. We can change n to any other value; only the relative weights of the points matters.

▶ **Theorem 2.** *The maximum size of a fully diverse set of point sets with a fixed total weight inside a unit-diameter region, assuming minimum matching distance or Earth Movers Distance, is* $\Theta(1)$.

3 Diversity of Distributions using the Wasserstein Distance

It is straightforward to extend this scheme to moving earth based on areas in simple polygons, or, more generally, moving mass using the Wasserstein distance. We extend the results of the previous section to distributions of unit mass inside U, after which we get the result for equal-area simple polygons using the Earth Movers Distance as a corollary.

The Wasserstein Distance measures the similarity of two probability density functions, and hence we assume the total mass of each to be 1. It is a metric that is essentially a continuous version of the Earth Movers Distance for points.

We simply snap volume to the same set of grid points as for the n-point set case, and we make batches as before as well. We replace n by 1 in the expressions, since this is the total mass and therefore the maximum Wasserstein distance in a unit-diameter region.

▶ **Theorem 3.** *The maximum size of a fully diverse set of distributions inside a unit-diameter region, assuming Wasserstein Distance, is* $\Theta(1)$.

▶ **Corollary 4.** *The maximum size of a fully diverse set of n-vertex, equal-area simple polygons inside a unit-diameter region, assuming Earth Movers Distance, is* $\Theta(1)$.

4 Diversity of Sets of Convex Polygons

Intuitively convex polygons have a lot less diversity than simple polygons, but does our concept of full diversity reflect this? For the Hausdorff distance and Earth Movers Distance the maximum size of a fully diverse set can be only $O(1)$, and the same upper bound holds when we restrict ourselves to convex polygons. So the question remains to be answered only for the Fréchet distance and the area of symmetric difference. In these cases we show that the maximum size of fully diverse sets becomes $O(1)$ as well.

The case of the Fréchet distance is easy, since the Fréchet distance and Hausdorff distance are the same for convex polygons (or their boundaries) [1, 4]. So the $O(1)$ upper bound directly transfers to this case.

The case of the area of symmetric difference requires a bit more work. We will show that for any convex shape, there is a convex polygon with constant complexity that has only little loss of area and no gain in area. Note that the maximum perimeter of a convex shape inside a unit diameter region is π and the maximum area is $\pi/4$ by the isoperimetric inequality.

Let $c > 1$ be a constant. If a convex shape s has perimeter less than $3/c^2$, then we define s' as a triangle using any three points on the boundary of s, and call s' a *simplification* of s.

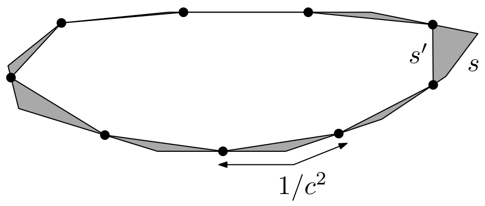


Figure 3 A convex shape s and its simplification s' by choosing a point on the boundary of s with interdistance $1/c^2$. The grey polygons are the symmetric difference; their area is the loss of area when s is simplified to s'.

Since the area of s is $< 1/c^4$, the triangle has lost at most that much area with respect to s. If s has perimeter at least $3/c^2$, then we choose points on the boundary of s at intervals $1/c^2$; the last interval may be shorter (see Figure 3). Let s' be the convex polygon with these points as its vertices; again we call s' the simplification of s. Clearly s' has $< \pi c^2 + 1$ vertices. The small convex polygons that are the symmetric difference of s and s' each have perimeter at most $2/c^2$ and therefore area $< 1/c^4$. There are $< \pi c^2 + 1$ of these small polygons, so their total area is $< 5/c^2$ since $c > 1$.

Let S be a set of convex shapes inside U and let $S' = \{s' \mid s \in S\}$ be the set of simplifications of the shapes in S. Let $c' > 1$ be a constant and let s'_1, s'_2 be two convex polygons that have area of symmetric difference at least $\pi/(4c')$, and hence they can be in a fully diverse set. We know that if s'_1 and s'_2 were derived as simplifications from s_1 and s_2, then the area of symmetric difference between s_1 and s_2 is at least $\pi/(4c') - 2 \cdot 5/c^2$, with c the constant used in the simplification. If we choose $c = c'$ and $c'' = \max(2c', 80/\pi)$, then c'' is a constant that witnesses that s_1 and s_2 can be together in a fully diverse set S. Hence, if S' is fully diverse for some constant c', then S is fully diverse for a different constant c''.

We already remarked that each polygon in S' has $O(c^2)$ vertices. Hence, the maximum size of a fully diverse set of simplified convex polygons S' is $2^{O(c^2 \log c)} = O(1)$, using the upper bound for the full diversity for simple polygons obtained in [10].

▶ **Theorem 5.** *The maximum size of a fully diverse set of convex polygons inside a unit-diameter region is* $\Theta(1)$ *for Hausdorff distance, Fréchet distance, area of symmetric difference, and Earth movers distance.*

5 Conclusions and Open Problems

We have continued the study of the maximum size of fully diverse sets of polygons by considering the Earth Movers Distance and the Wasserstein distance, and by considering convex polygons as a special case for all measures. It may be interesting to see how realistic polygons [5] behave, as a special case that lies in between the convex and the simple case. For example, it seems the $O(1)$ bound for convex polygons and area of symmetric difference does not use convexity, only the fact that the perimeter is bounded, and hence the $O(1)$ bound may hold for all constant perimeter simple polygons as well.

References

1 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.

2 Benjamin Bach, Andre Suslik Spritzer, Evelyne Lutton, and Jean-Daniel Fekete. Interactive random graph generation with evolutionary algorithms. In *Proc. 20th Int. Symp. on*

Graph Drawing (GD'12), volume 7704 of *LNCS*, pages 541–552, 2012. doi:10.1007/978-3-642-36763-2_48.

3 Julien Baste, Michael R. Fellows, Lars Jaffke, Tomás Masařík, Mateus de Oliveira Oliveira, Geevarghese Philip, and Frances A. Rosamond. Diversity of solutions: An exploration through the lens of fixed-parameter tractability theory. In *Proc. 29th International Joint Conference on Artificial Intelligence (IJCAI'20)*, pages 1119–1125, 2020. doi:10.24963/ijcai.2020/156.

4 Kevin Buchin, Maike Buchin, and Carola Wenk. Computing the Fréchet distance between simple polygons in polynomial time. In *Proc. 22nd Annual Symposium on Computational Geometry*, pages 80–87. ACM, 2006. doi:10.1145/1137856.1137870.

5 Mark de Berg, A. Frank van der Stappen, Jules Vleugels, and Matthew J. Katz. Realistic input models for geometric algorithms. *Algorithmica*, 34(1):81–97, 2002. doi:10.1007/s00453-002-0961-x.

6 Marina Drosou, H. V. Jagadish, Evaggelia Pitoura, and Julia Stoyanovich. Diversity in big data: A review. *Big Data*, 5(2):73–84, 2017. doi:10.1089/big.2016.0054.

7 Emmanuel Hebrard, Brahim Hnich, Barry O'Sullivan, and Toby Walsh. Finding diverse and similar solutions in constraint programming. In *Proc. 20th National Conference on Artificial Intelligence (AAAI'05)*, pages 372–377, 2005.

8 Mark O Hill. Diversity and evenness: a unifying notation and its consequences. *Ecology*, 54(2):427–432, 1973. doi:10.2307/1934352.

9 Linnea Ingmar, Maria Garcia de la Banda, Peter J. Stuckey, and Guido Tack. Modelling diversity of solutions. In *Proc. 34th AAAI Conference on Artificial Intelligence (AAAI'20)*, pages 1528–1535, 2020.

10 Fabian Klute and Marc van Kreveld. On fully diverse sets of geometric objects and graphs. In *International Workshop on Graph-Theoretic Concepts in Computer Science*, volume 13453 of *LNCS*, pages 328–341. Springer, 2022. doi:10.1007/978-3-031-15914-5_24.

11 Alex Kulesza and Ben Taskar. Determinantal point processes for machine learning. *Found. Trends Mach. Learn.*, 5(2-3):123–286, 2012. doi:10.1561/2200000044.

12 Matevz Kunaver and Tomaz Pozrl. Diversity in recommender systems - A survey. *Knowl.-Based Syst.*, 123:154–162, 2017. doi:10.1016/j.knosys.2017.02.009.

13 Hanna Tuomisto. A consistent terminology for quantifying species diversity? Yes, it does exist. *Oecologia*, 164(4):853–860, 2010. doi:10.1007/s00442-010-1812-0.

14 Tamara Ulrich, Johannes Bader, and Lothar Thiele. Defining and optimizing indicator-based diversity measures in multiobjective search. In *Proc. 11th Int. Conf. on Parallel Problem Solving from Nature (PPSN'10)*, volume 6238 of *LNCS*, pages 707–717, 2010. doi:10.1007/978-3-642-15844-5_71.

15 Marc van Kreveld, Bettina Speckmann, and Jérôme Urhausen. Diverse partitions of colored points. In *Proc. 17th Algorithms and Data Structures Symp. (WADS'2021)*, volume 12808 of *LNCS*, pages 641–654, 2021. doi:10.1007/978-3-030-83508-8_46.

16 Mark Wineberg and Franz Oppacher. The underlying similarity of diversity measures used in evolutionary computation. In *Proc. Annual Genetic and Evolutionary Computation Conference (GECCO'03)*, volume 2724 of *LNCS*, pages 1493–1504, 2003. doi:10.1007/3-540-45110-2_21.

A linear bound for the Colin de Verdière parameter μ for graphs embedded on surfaces

Camille Lanuel[*1], Francis Lazarus[†2], and Rudi Pendavingh[3]

1 Université de Lorraine, CNRS, Inria, LORIA, F-54000 Nancy, France
2 G-SCOP, CNRS, UGA, Grenoble, France
3 Department of Mathematics and Computer Science, Eindhoven University of Technology, 5600 MB Eindhoven, Netherlands

Abstract

The Colin de Verdière graph parameter $\mu(G)$ was introduced in 1990 by Y. Colin de Verdière. It is defined via spectral properties of a certain type of matrices, called Schrödinger operators, associated to a graph G. We provide a combinatorial and self-contained proof that for all graphs G embedded on a surface S, the Colin de Verdière parameter $\mu(G)$ is upper bounded by $7 - 2\chi(S)$, where $\chi(S)$ is the Euler characteristic of S.

Related Version https://arxiv.org/abs/2303.00556

1 Introduction

In this paper, we establish an upper bound for the Colin de Verdière's graph parameter μ for graphs that can be embedded on a fixed surface. This parameter was introduced by Colin de Verdière [4] in analogy with the multiplicity of the second eigenvalue of Schrödinger operators on a Riemannian surface. The exact definition of $\mu(G)$ resorts to a transversality condition between the space of so-called discrete Schrödinger operators on a graph $G = (V, E)$ and a certain stratification of the space of symmetric matrices of dimension $V \times V$. This *Strong Arnold Hypothesis* (SAH), as coined by Colin de Verdière [3], expresses a stability property and ensures that μ is minor-monotone. It can thus be applied to the graph minor theory of Robertson and Seymour. We refer to the survey by van der Holst, Lovász and Schrijver [13] for more properties on μ.

Here, we are interested in an upper bound for the parameter μ of the minor-closed family of graphs that can be embedded on a surface S. It is relatively easy to show that $\mu(K_n) = n - 1$ for K_n, the complete graph with n vertices [13]. On the other hand, the largest n such that K_n embeds on S is known as the Heawood number

$$\gamma(S) = \left\lfloor \frac{7 + \sqrt{49 - 24\chi(S)}}{2} \right\rfloor,$$

where $\chi(S)$ is the Euler characteristic of S. Colin de Verdière [2] conjectured that the maximum of μ for all graphs that can be embedded in S is attained at $K_{\gamma(S)}$. In other words, μ is upper bounded by $\gamma(S) - 1$. In practice, the known upper bounds have been proved in the realm of Riemannian surfaces where μ is defined for each Riemannian metric

[*] This work was conducted when this author was at the G-SCOP laboratory in Grenoble. This author is (partially) supported by the grant ANR-17-CE40-0033 of the French National Research Agency ANR (project SoS)
[†] This author is partially supported by LabEx PERSYVAL-Lab (ANR-11-LABX-0025-01) funded by the French program Investissement d'avenir.

as the maximum multiplicity of the second eigenvalue of Schrödinger differential operators (based on the Laplace-Beltrami operator associated to the given metric). In this framework, Besson [1] obtained the bound $7 - 2\chi(S)$. When the Euler characteristic is negative, this bound was further decreased by 2 by Nadirashvili [8], who thus showed the upper bound $5 - 2\chi(S)$. Sévennec [10] eventually divided by two the dependency in the characteristic to obtain the upper bound $5 - \chi(S)$. Note that all those bounds are linear in the genus of S and remain far from the square root bound in the conjecture of Colin de Verdière. It can be proved that any upper bound for μ in the Riemannian world holds for graphs, cf. [6, Th. 6.3] and [2, Th. 7.1]. However, the proof relies on the construction of Schrödinger differential operators from combinatorial ones and is not particularly illuminating from the combinatorial viewpoint. The goal of this paper is to propose a purely combinatorial and self-contained proof of the bound of Besson in the combinatorial framework of graphs. In the following, S is a compact surface, orientable or not.

▶ **Theorem 1.1.** *Let G be a graph that can be embedded on a surface S, then*

$$\mu(G) \leqslant 7 - 2\chi(S).$$

Our proof completes and slightly simplifies a proof of Pendavingh that appeared in his PhD thesis [9]. Before describing the general strategy of the proof, we provide some relevant definitions and basic facts.

2 Background

2.1 Schrödinger operators and μ

Let $G = (V, E)$ be a connected simplicial[1] graph with at least two vertices. A *Schrödinger operator* on G, sometimes called a generalized Laplacian, is a symmetric $V \times V$ matrix such that for $i \neq j \in V$, its ij coefficient is negative if $ij \in E$ and zero otherwise. (There is no condition on the diagonal coefficients.) We say that a Schrödinger operator L satisfies the *Strong Arnold Hypothesis (SAH)* if there is no nonzero symmetric matrix $X = (X_{ij})$ such that $LX = 0$ and $X_{ij} = 0$ whenever $i = j$ or $ij \notin E$, in other words, if the only symmetric matrix satisfying these conditions is zero. This is an algebraic translation of a transversality condition between the space of the discrete Schrödinger operators on G and a certain stratification of the space of symmetric matrices of dimension $V \times V$. It is not necessary to understand the SAH for the purpose of this paper.

It follows from Perron–Frobenius theorem that the first (smallest) eigenvalue of a Schrödinger operator L has multiplicity one [5]. Now, if λ_2 is the second eigenvalue of L, then the first eigenvalue of $L - \lambda_2\mathrm{Id}$, where Id is the identity matrix, is negative and its second eigenvalue is zero. This translation by $-\lambda_2\mathrm{Id}$ does not change the sequence of multiplicities of the eigenvalues of L nor the stability of L with respect to the SAH. Consequently, we can safely restrict to Schrödinger operators whose second eigenvalue is zero. We can now define the *Colin de Verdière graph parameter* $\mu(G)$ as the maximal corank (dimension of the kernel $\ker(L)$) of any Schrödinger operator L satisfying the Strong Arnold Hypothesis. In other words, $\mu(G)$ is the largest integer p such that there exists a Schrödinger operator L with $\dim\ker L = p$ and L satisfies the SAH.

Recall that a *minor* of G is a graph obtained from G by deleting edges, vertices or contracting egdes. As a fundamental property, μ is minor-monotone.

[1] A graph is *simplicial*, or *simple*, if it has no loops or multiple edges.

▶ **Theorem 2.1** ([4]). *If H is a minor of G, then $\mu(H) \leqslant \mu(G)$.*

It also characterizes planar graphs.

▶ **Theorem 2.2** ([4]). *A graph G is planar if and only if $\mu(G) \leqslant 3$.*

Colin de Verdière made the following conjecture in [4].

▶ **Conjecture 2.3.** *The chromatic number of a graph G satisfies* $\mathrm{chr}(G) \leqslant \mu(G) + 1$.

This conjecture contains the four colour theorem thanks to Theorem 2.2. It is implied by Hadwiger's conjecture (a graph which is not colourable with k colours has the complete graph K_{k+1} as a minor), using the fact that $\mu(K_n) = n - 1$ [13] and the minor-monotone property of μ.

We view a vector of $\mathbb{R}^V$ as a discrete map $V \to \mathbb{R}$, so that a Schrödinger operator acts linearly on the set of discrete maps. For $f : V \to \mathbb{R}$, we denote by V_f^+, V_f^0, V_f^- the subsets of vertices where f takes respectively positive, null and negative values. The *support* of f is the subset $V_f^+ \cup V_f^-$ of vertices with nonzero values. As a simple property of Schrödinger operators we have

▶ **Lemma 2.4** ([13]). *Let L be a Schrödinger operator of G and let $f \in \ker L$. Then, a vertex $v \in V_f^0$ is adjacent to a vertex of V_f^+ if and only if v is adjacent to a vertex of V_f^-.*

A discrete version of the nodal theorem of Courant reads as follows.

▶ **Theorem 2.5** ([5, 12]). *Let L be a Schrödinger operator of G and let $f \in \ker L$ be a nonzero map with minimal support*[2]*. Then, the subgraphs of G induced respectively by V_f^+ and V_f^- are nonempty and connected.*

2.2 Surfaces and Euler characteristic

By a *surface* of finite type we mean a topological space homeomorphic to a compact two dimensional manifold minus a finite number of points. A surface may have nonempty boundary and each of the finitely many boundary components is homeomorphic to a circle. We shall only consider surfaces of finite type and omit to specify this condition. A *closed* surface means a compact surface without boundary. By a *triangulation* of a surface, we mean a simplicial complex together with a homeomorphism between its underlying space and the surface.

The *Euler characteristic* $\chi(X)$ of a finite simplicial complex (and more generally a finite CW complex) is the alternating sum of the numbers of cells of each dimension. In particular, the Euler characteristic of a graph $\Gamma = (V, E)$ is $\chi(\Gamma) = |V| - |E|$. For a surface S we define $\chi(S)$ as the Euler characteristic of a polygonisation of S, that is, a description of S as a CW complex. The Euler characteristic is homotopy invariant: two spaces with the same homotopy type have the same Euler characteristic.

The following property will be needed in our proof. It is a corollary of the Inclusion-exclusion formula [11, p.205]. The proof is not completely trivial and can be found in [7, Section 2] .

[2] A discrete map $f \in \ker L$ has minimal support if it is nonzero and for every nonzero $g \in \ker L$, if $V_g^+ \cup V_g^- \subseteq V_f^+ \cup V_f^-$, then $V_g^+ \cup V_g^- = V_f^+ \cup V_f^-$.

▶ **Proposition 2.6.** *Let X be a triangulated compact surface and let Y be a subcomplex of X. Then*

$$\chi(X) = \chi(Y) + \chi(X \setminus Y).$$

Here, X and Y should be considered as topological spaces, forgetting about their triangulations. In general, $X \setminus Y$ is not a subcomplex of X.

3 Overview of the proof

In this section, we give the main ideas of the proof of Theorem 1.1. The detailed proof can be found in [7, Section 5].

Let G be a graph embedded on a surface S, which we can assume closed without loss of generality. We may also assume that S is not homeomorphic to a sphere as otherwise Theorem 1.1 follows directly from Theorem 2.2. In a first step, we remove an open disk $D \subset S \setminus G$ whose boundary avoids G, and build a graph[3] H embedded in $S \setminus D$ such that

C1. H triangulates $S \setminus D$ and subdivides ∂D into a cycle of $\mu(G) - 1$ edges,

C2. G is a minor of H, and

C3. the length of the shortest closed walk in H that is non-contractible in $S \setminus D$, i.e. the *edgewidth* of H in $S \setminus D$, is $\mu(G) - 1$.

We denote by W the set of vertices of H.

We next choose a Schrödinger operator L for H whose corank achieves $\mu(H)$. Condition C2 and the monotonicity of μ imply $\mu(H) \geqslant \mu(G)$, so that $\ker L$ has dimension at least $\mu(G)$. By C1, ∂D has $\mu(G) - 1$ vertices, so the vectors of the basis of $\ker L$ restricted to ∂D are linearly dependent. Thus there exists a nonzero vector $f \in \ker L$ such that f cancels on the vertices of ∂D. We pick such an f with minimal support so that by Theorem 2.5 the subsets of vertices W_f^+ and W_f^- induce connected subgraphs of H. We connect the vertices of ∂D by inserting $\mu(G) - 4$ edges in D to obtain a graph H' with the same vertices as H and that triangulates S. We can now extend f linearly on each face of H' to get a piecewise linear map $\bar{f} : S \to \mathbb{R}$. Let S_f^+, S_f^0, S_f^- denote the subspaces of S where $\bar{f}$ is respectively positive, null, and negative. By Theorem 2.5, S_f^+ and S_f^- are connected open subsurfaces of S, while S_f^0 is a closed subcomplex of some subdivision of the triangulation induced by H'. We can thus apply Proposition 2.6 to write

$$\chi(S) = \chi\left(S_f^0\right) + \chi\left(S_f^+ \cup S_f^-\right) = \chi\left(S_f^0\right) + \chi\left(S_f^+\right) + \chi\left(S_f^-\right).$$

The subsets S_f^+ and S_f^- cannot be homeomorphic to spheres because they are both proper subsets of S. We deduce from the classification of surfaces, that $\chi(S_f^+) \leqslant 1$ and $\chi(S_f^-) \leqslant 1$. It ensues that $\chi(S_f^0) \geqslant \chi(S) - 2$. The goal is now to provide an upper bound for $\chi(S_f^0)$ in terms of $\mu(G)$ in order to obtain the desired upper bound for $\mu(G)$.

Start by observing S_f^0: it is formed of plain triangles adjacent by an edge or a vertex, or connected together, via their vertices, by a piece of dimension 1. To provide the upper bound for $\chi(S_f^0)$, we build a graph Γ whose Euler characteristic is larger than S_f^0 by contracting its two dimensional parts, that are the parts formed by the plain triangles. To ensure that this operation results in the desired property on the Euler characteristic of Γ, precautions are taken before the contraction to remove the singularities of the two dimensional parts while keeping the same homotopy type. By definition of $\bar{f}$, $D \subset S_f^0$, so we can define K as the two

[3] See [7, Section 4] for a detailed construction of such a graph.

dimensional part containing D. Because of condition C3 on H, we can argue that K has a non-contractible boundary in $S \setminus D$. It follows that this boundary has length at least the edgewidth of H, hence at least $\mu(G) - 1$ by condition C3. Thanks to Lemma 2.4 we may infer that K contracts to a vertex of degree at least $\mu(G) - 1$ in Γ. We also argue thanks to Lemma 2.4 that Γ has no vertex of degree one. By the handshaking lemma applied to Γ, we deduce that $\chi(S_f^0) \leqslant \chi(\Gamma) \leqslant (3 - \mu(G))/2$. We finally conclude that $\chi(S) - 2 \leqslant (3 - \mu(G))/2$, hence $\mu(G) \leqslant 7 - 2\chi(S)$.

References

1 Gérard Besson. Sur la multiplicité de la première valeur propre des surfaces riemanniennes. *Annales de l'institut Fourier*, 30(1):109–128, 1980. URL: `http://www.numdam.org/item/10.5802/aif.777.pdf`.

2 Yves Colin de Verdière. Construction de laplaciens dont une partie finie du spectre est donnée. *Annales scientifiques de l'école normale supérieure*, 20(4):599–615, 1987.

3 Yves Colin de Verdière. Sur une hypothèse de transversalité d'Arnol'd. *Comment. Math. Helv.*, 63(2):184–193, 1988.

4 Yves Colin de Verdière. Sur un nouvel invariant des graphes et un critère de planarité. *Journal of Combinatorial Theory, Series B*, 50(1):11–21, 1990.

5 Yves Colin de Verdière. Théorèmes de Courant et de Cheng combinatoires. *Séminaire de théorie spectrale et géométrie*, 13:9–13, 1994.

6 Yves Colin de Verdière. *Spectres de graphes.* Number 4 in Cours Spécialiés. Société Mathématique de France, 1998.

7 Camille Lanuel, Francis Lazarus, and Rudi Pendavingh. A linear bound for the Colin de Verdière parameter μ for graphs embedded on surfaces, 2023. Preprint. URL: `https://arxiv.org/abs/2303.00556`, `doi:10.48550/ARXIV.2303.00556`.

8 Nikolai Semenovich Nadirashvili. Multiple eigenvalues of the Laplace operator. *Mathematics of the USSR-Sbornik*, 61(1):225, 1988.

9 Rudolf Anton Pendavingh. *Spectral and Geometrical Graph Characterizations.* PhD thesis, Amsterdam University, 1998.

10 Bruno Sévennec. Multiplicity of the second Schrödinger eigenvalue on closed surfaces. *Mathematische Annalen*, 324(1):195–211, 2002.

11 Edwin H. Spanier. *Algebraic topology.* Springer Verlag, corrected reprint of the 1966 original edition, 1989.

12 Hein Van der Holst. A short proof of the planarity characterization of Colin de Verdière. *Journal of Combinatorial Theory, Series B*, 65(2):269–272, 1995. URL: `https://www.sciencedirect.com/science/article/pii/S0095895685710544/pdf?md5=cb7b57b58c203666ae179c8de5e87e19&pid=1-s2.0-S0095895685710544-main.pdf`.

13 Hein Van der Holst, László Lovász, and Alexander Schrijver. The Colin de Verdière graph parameter. *Graph Theory and Computational Biology (Balatonlelle, 1996)*, pages 29–85, 1999.

On the algebraic connectivity of token graphs of a cycle[*]

M. A. Reyes[1], C. Dalfó[1], M. A. Fiol[2], and A. Messegué[1]

1 Dept. de Matemàtica, Universitat de Lleida, Lleida/Igualada, Catalonia
`{monicaandrea,cristina.dalfo,visitant.arnau.messegue@udl.cat}`

2 Dept. de Matemàtiques, Universitat Politècnica de Catalunya, Barcelona, Catalonia
`miguel.angel.fiol@upc.edu`

Abstract

We study the algebraic connectivity (the smallest non-zero eigenvalue of the Laplacian matrix and also known as Fiedler eigenvalue) of token graphs. The k-token graph $F_k(G)$ of a graph G is the graph whose vertices are the k-subsets of vertices from G, two of which are adjacent if and only if their symmetric difference is a pair of adjacent vertices in G. Recently, it was conjectured that the algebraic connectivity of $F_k(G)$ equals the algebraic connectivity of G. In this paper, we prove the conjecture for the cycle graphs C_n for all n when $k = 2$.

1 Introduction

In computer networks, and particularly in digital communications, we often have a graph designed in such a way that pieces of information, or *tokens*, can be moved from one vertex to another. A large class of such graphs is the token graphs. Given a graph G, the k-token graph $F_k(G)$ of G is the graph whose vertices are the k-subsets of vertices of G, two of which are adjacent if and only if their symmetric difference is a pair of adjacent vertices in G. Vertices of $F_k(G)$ correspond to the configurations of k indistinguishable tokens placed at distinct vertices of G, where two configurations are adjacent (as vertices of $F_k(G)$) if and only if one configuration can be reached from the other by moving in G one token along an edge from its current position to an unoccupied vertex.

The algebraic connectivity can be used to find the size of the separator of a graph, giving partitioning techniques. These techniques are useful in many scientific numerical algorithms. Besides, the Embedding Lemma by Spielman and Teng [6] has been applied to relate drawings of graphs with the algebraic connectivity. For more information, see Fabila-Monroy, Hidalgo-Toscano, Huemer, Lara, and Mitsche [4].

Many properties of token graphs, such as its diameter, chromatic number, and Hamiltonian paths have recently been studied by Fabila-Monroy, Flores-Peñaloza, Huemer, Hurtado, Urrutia, and Wood [5]. We focus on the study of the algebraic connectivity of the token graph $F_k(G)$, denoted by $\alpha(F_k(G))$. Recently, it was conjectured by Dalfó, Duque, Fabila-Monroy, Fiol, Huemer, Trujillo-Negrete, and Zaragoza Martínez [2] that the algebraic connectivity of G and $F_k(G)$ are equal, that is, $\alpha(G) = \alpha(F_k(G))$. This conjecture has already been proven when G is a complete graph, a complete bipartite graph or a tree, among other special cases in [2] and [3]. We present results on this conjecture when G is the cyclic graph C_n on n

[*] This research has been supported by AGAUR from the Catalan Government under project 2021SGR00434 and MICINN from the Spanish Government under project PID2020-115442RB-I00.

vertices. In particular, for the case $k = 2$, we give an efficient algorithm to compute the whole spectrum of $F_2(C_n)$ (and, in particular, its algebraic connectivity).

1.1 Some notation

We use the following notation throughout the paper: $\boldsymbol{L} = \boldsymbol{L}(G)$ is the Laplacian matrix of the graph G, $\pi = V_1 \cup V_2 \cup \cdots \cup V_r$ a (regular or not) partition of the vertex set, G/π is the quotient graph over π, $\boldsymbol{A}(G/\pi)$ and $\boldsymbol{L}(G/\pi)$ the adjacency and Laplacian matrices of G/π, respectively.

The transpose of a matrix $\boldsymbol{M}$ is denoted by $\boldsymbol{M}^\top$, the identity matrix by $\boldsymbol{I}$ and the all-1 vector $(1, \dots, 1)^\top$ by $\boldsymbol{1}$.

The Laplacian matrix $\boldsymbol{L}(G)$ of a graph G on n vertices is positive semidefinite, with eigenvalues $(0 =)\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$. Given some integers n and k (with $k \in [n]$), we define the $(n; k)$-*binomial matrix* $\boldsymbol{B}$. This is a $\binom{n}{k} \times n$ matrix whose rows are the characteristic vectors of the k-subsets of $[n]$ in a given order. Thus, if the i-th k-subset is A, then

$$(\boldsymbol{B})_{ij} = \begin{cases} 1 & \text{if } j \in A, \\ 0 & \text{otherwise.} \end{cases}$$

A partition $\pi = (V_1, \dots, V_r)$ of V is called *regular* (or *equitable*) whenever, for any $i, j = 1, \dots, r$, the *intersection numbers* $b_{ij}(u) = |N(u) \cap V_j|$, where $u \in V_i$, do not depend on the vertex u but only on the subsets (usually called *classes* or *cells*) V_i and V_j and $N(u)$ denote the set of vertices adjacent to u. In this case, such numbers are simply written as b_{ij}, and the $r \times r$ matrix $\boldsymbol{Q}_L = \boldsymbol{L}(G/\pi)$ with entries

$$(\boldsymbol{Q}_L)_{ij} = \begin{cases} -b_{ij} & \text{if } i \neq j, \\ \displaystyle\sum_{h=1}^{r} b_{ih} - b_{ii} & \text{if } i = j, \end{cases}$$

is referred to as the *quotient Laplacian matrix* of G with respect to π.

2 The spectrum of $F_2(C_n)$

In this section, we give some results about the eigenvalues of $F_2(C_n)$.

▶ **Theorem 2.1.** *(i) If $n = 2\nu$ is even, then the 2-token graph $F_2(C_n)$ has the eigenvalues*

$$\lambda_r = 8 \sin^2\left(\frac{r\pi}{n-1}\right), \qquad r = 0, 1, \dots, \nu - 1, \tag{1}$$

with $\lambda_0 = 0$, and $\lambda_{\nu-1} = 8\sin^2\left(\frac{n-2}{n-1}\frac{\pi}{2}\right)$ is the spectral radius of $F_2(C_n)$.

(ii) If $n = 2\nu + 1$ is odd, then the 2-token $F_2(C_n)$ has the eigenvalues

$$\lambda_r = 8 \cos^2\left(\frac{r\pi}{n-1}\right), \qquad r = 1, 2, \dots, \nu, \tag{2}$$

with $\lambda_\nu = 0$, and $\lambda_1 = 8\cos^2\left(\frac{\pi}{n-1}\right)$ is a lower bound for the spectral radius (the maximum eigenvalue of its Laplacian matrix) of $F_2(C_n)$.

Proof. Let us see that $F_2 = F_2(C_n)$ has a regular 'path-shaped' partition π with $r = \lfloor n/2 \rfloor$ classes $V_1, V_2, \dots, V_r$, where V_i consists of the vertices $\{u, v\}$ such that $\mathrm{dist}(u, v) = i$ in C_n.

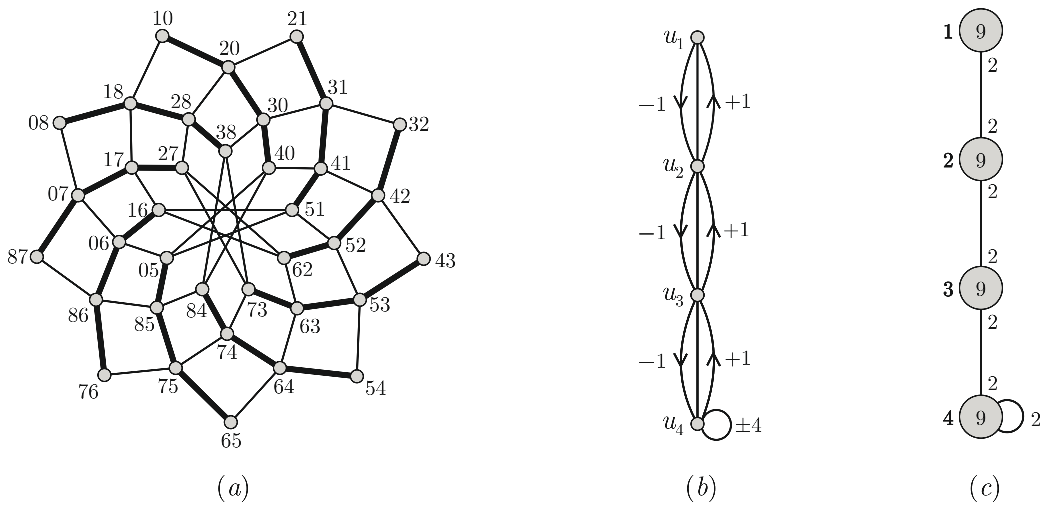


Figure 1 (a) The 2-token graph $F_2(C_9)$ of the cycle graph C_9. The thick edges correspond to each of the copies of the base graph or the quotient graph. (b) Its base graph with voltages on $\mathbb{Z}_9$. (c) The quotient graph of its path-shaped regular partition. In boldface, there is the numbering of the vertex classes. In class $c \in \{1,2,3,4\}$, there are the vertices ij that satisfy $i - j = c \,(\mathrm{mod}\, 9)$.

- Each vertex $\{u,v\}$ in V_1 is adjacent to 2 vertices $\{u+1,v\}$ and $\{u,v+1\}$ in V_2 (all arithmetic is modulo n).
- Each vertex $\{u,v\}$ in V_i, for $i = 2,\dots,r-1$, is adjacent to 2 vertices $\{u-1,v\}$ and $\{u,v-1\}$ in V_{i-1}, and 2 vertices $\{u+1,v\}$ and $\{u,v+1\}$ in V_{i+1}.
- Every vertex $\{u,v\}$ in V_r is adjacent to 4 vertices $\{u\pm 1,v\}$ and $\{u,v\pm 1\}$. If n is even all these vertices are in V_{r-1}. If n is odd, two of them are in V_{r-1} and the other two in V_r.

Then, (1) and (2) correspond to the eigenvalues of the matrix $\boldsymbol{Q}_L$ for the even and odd cases of n, respectively (Yueh [7, Section 3]). Moreover, in the case of even n, the maximum eigenvalue in (1) is obtained when $r = \nu - 1$. Then, $\lambda_{\nu-1}$ is the spectral radius of $F_2(C_n)$. ◀

Next, we show that, depending on the parity of n, much more can be stated about the spectrum of $F_k(C_n)$.

2.1 The case of odd n

We now study the case of odd n. We follow the notation of [1].

Let $\mathcal{G}$ be a group. An (*ordinary*) *voltage assignment* on the (di)graph (that is, graph o digraph) $G = (V,E)$ is a mapping $\beta : E \to \mathcal{G}$ with the property that $\beta(a^-) = (\beta(a^+))^{-1}$ for every arc $a \in E$. Thus, a voltage assigns an element $g \in \mathcal{G}$ to each arc of the (di)graph, so that a pair of mutually reverse arcs a^+ and a^-, forming an undirected edge, receive mutually inverse elements g and g^{-1}. The (di)graph G and the voltage assignment β determine a new (di)graph G^β, called the *lift* of G, which is defined as follows. The vertex and arc sets of the lift are simply the Cartesian products $V^\beta = V \times \mathcal{G}$ and $E^\beta = E \times \mathcal{G}$, respectively. Moreover, for every arc $a \in E$ from a vertex u to a vertex v for $u,v \in V$ (possibly, $u = v$) in G, and for every element $g \in \mathcal{G}$, there is an arc $(a,g) \in E^\beta$ from the vertex $(u,g) \in V^\beta$ to the vertex $(v, g\beta(a)) \in V^\beta$. For more details, see again [1].

▶ **Theorem 2.2** ([1]). *The 2-token graph $F_2(C_n)$ of the cycle with an odd number $n = 2\nu + 1$ of vertices is the lift $G^{\beta}(P_{\nu}^{+})$ of the base graph the path P_{ν}^{+} with vertex set $\{u_1, u_2, \dots, u_{\nu}\}$ and a loop at u_{ν}, and voltages on the group $\mathbb{Z}_n$ as follows:*

$$\begin{aligned} \beta(u_i u_{i+1}) &= -1 \qquad \text{for } i = 1, \dots, \nu - 1, \\ \beta(u_{i+1} u_i) &= +1 \qquad \text{for } i = 1, \dots, \nu - 1, \\ \beta(u_{\nu} u_{\nu}) &= \pm\nu. \end{aligned}$$

For example, in the case of $n = 9$, the 2-token graph $F_2(C_9)$ and its base graph are shown in Figure 1. Then, the whole spectrum of $F_2(C_n)$ can be obtained from its Laplacian base $\nu \times \nu$ matrix $\boldsymbol{B}(z)$ (see again [1]), with $z = e^{ir\frac{2\pi}{n}}$, or its similar tridiagonal matrix $\boldsymbol{B}^*(r)$, for $r = 0, 1, \dots, n - 1$.

$$\boldsymbol{B}(z) = \begin{pmatrix} 2 & -1-z^{-1} & 0 & 0 & \dots & 0 \\ -1-z & 4 & -1-z^{-1} & 0 & \dots & 0 \\ 0 & -1-z & 4 & -1-z^{-1} & \ddots & 0 \\ 0 & 0 & -1-z & \ddots & \ddots & 0 \\ \vdots & \vdots & \ddots & \ddots & 4 & -1-z^{-1} \\ 0 & 0 & \dots & 0 & -1-z & 4-z^{\nu}-z^{-\nu} \end{pmatrix} \cong$$

$$\boldsymbol{B}^*(r) = \begin{pmatrix} 2 & 2\cos(\frac{r\pi}{n}) & 0 & 0 & \dots & 0 \\ 2\cos(\frac{r\pi}{n}) & 4 & 2\cos(\frac{r\pi}{n}) & 0 & \dots & 0 \\ 0 & 2\cos(\frac{r\pi}{n}) & 4 & 2\cos(\frac{r\pi}{n}) & \ddots & 0 \\ 0 & 0 & 2\cos(\frac{r\pi}{n}) & \ddots & \ddots & 0 \\ \vdots & \vdots & \ddots & \ddots & 4 & 2\cos(\frac{r\pi}{n}) \\ 0 & 0 & \dots & 0 & 2\cos(\frac{r\pi}{n}) & 4+4(-1)^{r+1}\cos(\frac{r\pi}{n}) \end{pmatrix}. \tag{3}$$

▶ **Theorem 2.3.** *Every eigenvalue λ of $F_2(C_n) \cong G^{\beta}(P_{\nu}^{+})$, with $n = 2\nu + 1$, has an eigenvector $\boldsymbol{y} \in \mathbb{R}^{\binom{n}{2}}$ with components*

$$y_{(i,j)} = f_{i-j}\zeta^j = f_h\zeta^j \qquad j = 0, \dots, n-1, \; h = 1, \dots, \nu, \tag{4}$$

where ζ is an n-th root of unity, and $\boldsymbol{f} = (f_1, \dots, f_{\nu})$ is a λ-eigenvector of the matrix $\boldsymbol{B}(\zeta)$.

2.2 The case of even n and odd $n/2$

In the case of $n \equiv 2 \pmod 4$ (so that $n/2$ is odd), the 2-token $F_2(C_n)$ can also be seen as a lifted graph, as shown in the following result. See an example of this kind of token graph in Figure 2.

Given an integer r, let us consider the path graph $G = P_{4r+1}$ with vertices

$$u_{-2r}, u_{-2r+1}, \dots, u_{-1}, u_0, u_1, \dots, u_{2r-1}, u_{2r},$$

(with its corresponding edges) and additional arcs

$$\begin{aligned} a_i^+ &= u_{-i}u_{i-1}, \quad a_i^- = u_{i-1}u_{-i}, \quad \text{for } i = 0, 1 \dots, 2r, \\ b_i^+ &= u_i u_{-i+1}, \quad b_i^- = u_{-i+1}u_i, \quad \text{for } i = 0, 1, \dots, 2r. \end{aligned}$$

Let β be the voltage assignment on G in the cyclic group $\mathbb{Z}_{2r+1}$ given by

$$\begin{aligned} \beta(a_i^+) = \beta(b_i^+) &= +r, \\ \beta(a_i^-) = \beta(b_i^-) &= -r. \end{aligned} \tag{5}$$

Figure 2(b) shows the base graph G for $r = 2$. Now, we have the following result.

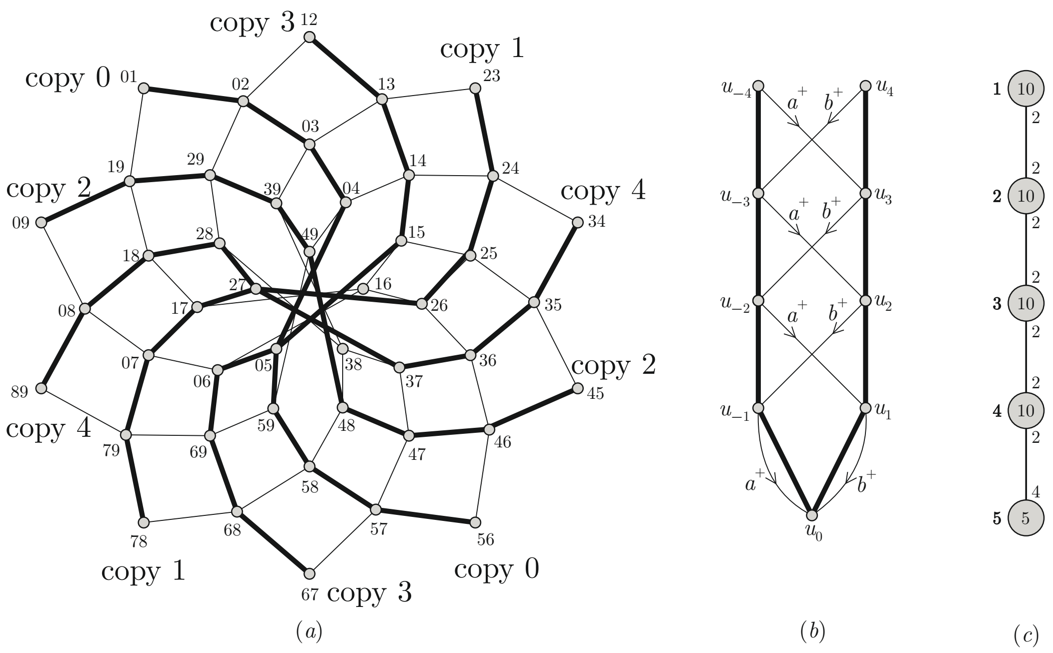


Figure 2 (a) The token graph $F_2(C_{10})$ of the cycle graph C_{10}, with the different copies of the U-shaped regular partition, here drawn as paths. (b) Its base digraph with voltages in $\mathbb{Z}_{10}$, which gives rise to the U-shaped regular partition. The thick edges represent the path P_9, obtained with this partition. (c) The quotient graph of the path-shaped regular partition. In boldface, there is the numbering of the vertex classes.

▶ **Lemma 2.4.** *Given $G = P_{4r+1}$ with the voltage assignment* (5) *on $\mathbb{Z}_{2r+1}$, the 2-token graph of the cycle C_n with $n = 4r + 2$ is the lift graph G^β. That is,*

$$F_2(C_{4r+2}) \cong G^\beta.$$

Proof. The vertex set $V(G^\beta)$ of the lift G^β has elements labeled with the pairs (u_i, g) for $i = -2r, \ldots, 2r$ and $g \in \mathbb{Z}_{2r+1}$. Thus $|V(G^\beta)| = (4r + 1)(2r + 1) = \binom{4r+2}{2}$, which corresponds to the number of vertices of $F_2(C_{4r+2})$. Indeed, such vertices correspond to 2-subsets $\{i, j\}$ of the set $\{1, 2, \ldots, 4r + 2\}$. Thus, we have to show a 1-to-1 mapping between $V(G^\beta)$ and $V(F_2(C_{4r+2}))$ that must be consequent with the adjacencies of both graphs. Let us take an example. The vertex $(u_{-2r+1}, 1) \equiv \{2, 4\}$ (written as 24 in Figure 2) of G^β is adjacent to:

- The vertices $(u_{-2r}, 1)$ and $(u_{-2r+2}, 1)$ of the same 'copy'.
- The vertex $(u_{2r-2}, r + 1)$ by the arc a^+_{2r-1} with voltage $+r$.
- The vertex $(u_{2r}, r + 2)$ by the arc b^-_{2r} with voltage $-r$.

Then, we find the following equivalences:

- $(u_{-2r}, 1) \equiv \{2, 3\}$ and $(u_{-2r+2}, 1) \equiv \{2, 5\}$,
- $(u_{2r-2}, r + 1) \equiv \{1, 4\}$ and $(u_{2r}, r + 2) \equiv \{3, 4\}$,

which correspond to the vertices adjacent to $\{2, 4\}$ in $F_2(C_{4r+2})$. ◀

As a consequence, for each $z = \omega^\ell$, where $\omega = e^{i\frac{2\pi}{2r+1}}$, with $\ell = 0, 1, \ldots, 2r$, an irreducible representation of the Laplacian base matrix of $G^\beta \cong F_2(C_{2r+2})$ is the matrix $\boldsymbol{B}(z)$ as shown next.

Note that matrix $\boldsymbol{B}(z)$ is tridiagonal with respect to the main and the secondary diagonals.

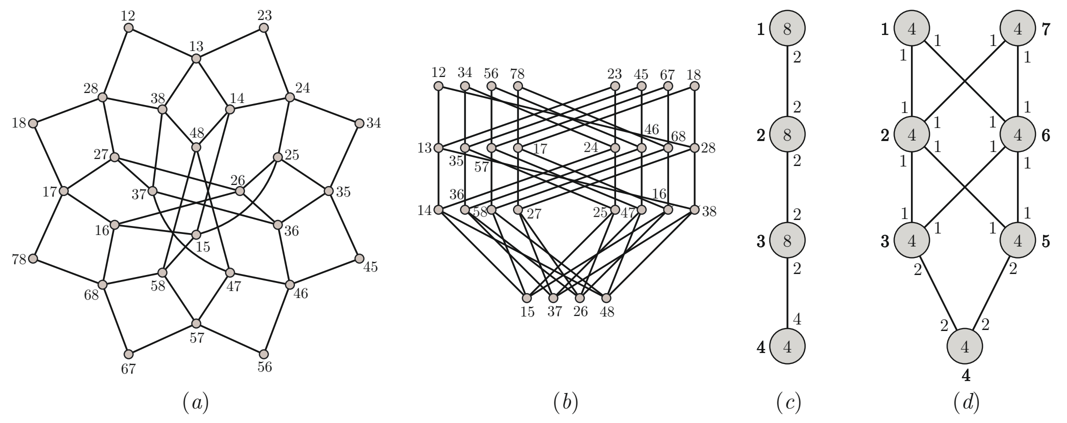


Figure 3 (a) The 2-token graph $F_2(C_8)$ of the cycle graph on 8 vertices. (b) Another view of $F_2(C_8)$. (c) The quotient graph from the path-shaped regular partition. (d) The quotient graph of the U-shaped regular partition obtained from (b). In boldface, there is the numbering of the vertex classes.

$$
\boldsymbol{B}(z) = \begin{pmatrix}
2 & -1 & 0 & \cdots & 0 & 0 & 0 & 0 & 0 & \cdots & 0 & -z^{r} & 0 \\
-1 & 4 & -1 & \cdots & 0 & 0 & 0 & 0 & 0 & \cdots & -z^{r} & 0 & -z^{-r} \\
0 & -1 & 4 & \cdots & 0 & 0 & 0 & 0 & 0 & \cdots & 0 & -z^{-r} & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\
0 & 0 & 0 & \cdots & 4 & -1 & 0 & -z^{r} & 0 & \cdots & 0 & 0 & 0 \\
0 & 0 & 0 & \cdots & -1 & 4 & -1-z^{r} & 0 & -z^{-r} & \cdots & 0 & 0 & 0 \\
0 & 0 & 0 & \cdots & 0 & -1-z^{-r} & 4 & -1-z^{-r} & 0 & \cdots & 0 & 0 & 0 \\
0 & 0 & 0 & \cdots & -z^{-r} & 0 & -1-z^{r} & 4 & -1 & \cdots & 0 & 0 & 0 \\
0 & 0 & 0 & \cdots & 0 & -z^{r} & 0 & -1 & 4 & \cdots & 0 & 0 & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\
0 & -z^{-r} & 0 & \cdots & 0 & 0 & 0 & 0 & 0 & \cdots & 4 & -1 & 0 \\
-z^{-r} & 0 & -z^{r} & \cdots & 0 & 0 & 0 & 0 & 0 & \cdots & -1 & 4 & -1 \\
0 & -z^{r} & 0 & \cdots & 0 & 0 & 0 & 0 & 0 & \cdots & 0 & -1 & 2
\end{pmatrix}.
$$

2.3 The case of even n and $n/2$

When we consider the case of cycles C_n with n and $n/2$ even, it is not very useful to represent $F_2(C_n)$ as a lift graph to compute the whole spectrum. The reason is that, the base graph would have too many vertices with respect to the original graph. Alternatively, besides the spectrum of C_n, we can easily find another part of the spectrum by means of regular partitions. As an example, the 2-token graph of C_8 is shown in Figure 3(a) and (b), together with its regular partitions (c) (the path $P_{n/2}$) and (d) (the U-shaped graph). Compare such partitions with those in Figure 2(b) and (c). We use the new method of over-lifts to solve the case with even n.

2.4 The method of over-lifts

In this subsection, we use a new method called *over-lifts*, which allows us to unify the cases of cycles with even n and compute the whole spectrum of $F_2(C_n)$. This is accomplished by means of a new polynomial matrix $\boldsymbol{B}(z)$ that does **not** correspond to the base graph of a lift. By its characteristics, we say that $\boldsymbol{B}(z)$ is associated with an over-lift. The basic difference is that such a matrix has dimension $\nu \times \nu$ (recall that $n = 2\nu$), and there are n possible values for z (n-th roots of unity). Thus, the total number of eigenvalues obtained is νn. However, $F_2(C_n)$ has $\binom{n}{2} = \nu(n-1)$ vertices, which is the number of eigenvalues of Ł. We will see that, in fact, the ν 'extra' eigenvalues provided by $\boldsymbol{B}(z)$ are all equal to 4.

▶ **Theorem 2.5.** *Let $\boldsymbol{L}$ be the Laplacian matrix of $F_2(C_n)$, with $n = 2\nu$. Let Λ be the multiset with elements $4, 4, \overset{(\nu)}{\ldots}, 4$. Then, the spectrum of $\boldsymbol{L}$ can be obtained from the spectrum of the $\nu \times \nu$ matrix $\boldsymbol{B}(z)$ or, equivalently, from the spectrum of its similar matrix $\boldsymbol{B}^*(r)$:*

$$\boldsymbol{B}(z) = \begin{pmatrix} 2 & -1-z^{-1} & 0 & 0 & \dots & 0 \\ -1-z & 4 & -1-z^{-1} & 0 & \dots & 0 \\ 0 & -1-z & 4 & -1-z^{-1} & \ddots & 0 \\ 0 & 0 & -1-z & \ddots & \ddots & 0 \\ \vdots & \vdots & \ddots & \ddots & 4 & -1-z^{-1} \\ 0 & 0 & \dots & 0 & -1-z-z^{\nu}-z^{\nu+1} & 4 \end{pmatrix}$$

for $z = \zeta^r = e^{ir\frac{2\pi}{n}}$, $r = 0, 1, \ldots, n-1$, and

$$\boldsymbol{B}^*(r) = \begin{pmatrix} 2 & 2\cos(\frac{r\pi}{n}) & 0 & 0 & \dots & 0 \\ 2\cos(\frac{r\pi}{n}) & 4 & 2\cos(\frac{r\pi}{n}) & 0 & \dots & 0 \\ 0 & 2\cos(\frac{r\pi}{n}) & 4 & 2\cos(\frac{r\pi}{n}) & \ddots & 0 \\ 0 & 0 & 2\cos(\frac{r\pi}{n}) & \ddots & \ddots & 0 \\ \vdots & \vdots & \ddots & \ddots & 4 & 2\cos(\frac{r\pi}{n}) \\ 0 & 0 & \dots & 0 & 2\cos(\frac{r\pi}{n}) + 2\cos(\frac{r(n-1)\pi}{n}) & 4 \end{pmatrix} \tag{6}$$

for $r = 0, 1, \ldots, n-1$. Formally,

$$\bigcup_{z\in R(n)} \operatorname{sp} \boldsymbol{B}(z) = \bigcup_{r=0}^{n-1} \operatorname{sp} \boldsymbol{B}^*(r) = \Lambda \cup \operatorname{sp} \boldsymbol{L}, \tag{7}$$

where $R(n)$ denotes the set of the n-th roots of unity,

3 The algebraic connectivity of $F_2(C_n)$

In this last section, we show that the conjecture posed in Dalfó, Duque, Fabila-Monroy, Fiol, Huemer, Trujillo-Negrete, and Zaragoza Martínez [2] holds for the 2-token graphs of a cycle. Of all the cases in which this conjecture is known to hold, this case turned out to be the most involved.

▶ **Theorem 3.1.** *The algebraic connectivity of the 2-token graph $F_2(C_n)$ equals the algebraic connectivity of the cycle C_n.*

▶ **Corollary 3.2.** *The algebraic connectivity of the 2-token graph $F_2(G)$ of a unicyclic graph G equals the algebraic connectivity of G.*

References

1 C. Dalfó, M. A. Fiol, S. Pavlíková, and J. Širán. A note on the spectra and eigenspaces of the universal adjacency matrices of arbitrary lifts of graphs. *Linear Multilinear Algebra*, 2022, DOI: 10.1080/03081087.2022.2042174.

2 C. Dalfó, F. Duque, R. Fabila-Monroy, M. A. Fiol, C. Huemer, A. L. Trujillo-Negrete, and F. J. Zaragoza Martínez. On the laplacian spectra of token graphs. *Linear Algebra Appl.*, 625 (322-348), 2021.

3 C. Dalfó and M. A. Fiol. On the algebraic connectivity of token graphs. 2022, arXiv:2209.01030v1.

4 R. Fabila-Monroy, C. Hidalgo-Toscano Carlos, C. Huemer, D. Lara, and D. Mitsche. Optimal grid drawings of complete multipartite graphs and an integer variant of the algebraic connectivity. *Graph Drawing and Network Visualization. Lecture Notes in Computer Science*, 11282 (593-605), 2018.

5 R. Fabila-Monroy, D. Flores-Peñaloza, C. Huemer, F. Hurtado, J. Urrutia, and D. R. Wood. Token graphs. *Graphs Combin.*, 28 (no. 3, 365-380), 2012.

6 D. A. Spielman and S.-H. Teng. Spectral partitioning works: Planar graphs and finite element meshes. *Linear Algebra and its Applications*, 421 (284-305), 2007, https://doi.org10.1016/j.laa.2006.07.020.

7 W. C. Yueh. Eigenvalues of several tridiagonal matrices. *Appl. Math. E-Notes*, 5 (66-74), 2005.

Euclidean One-of-a-Set TSP

Henk Alkema[1] **and Mark de Berg**[2]

1 **Department of Mathematics and Computer Science, TU Eindhoven, the Netherlands**
`h.y.alkema@tue.nl`

2 **Department of Mathematics and Computer Science, TU Eindhoven, the Netherlands**
`m.t.d.berg@tue.nl`

Abstract

In One-of-a-Set tsp, also known as Generalised Traveling Salesman Problem, the input is a collection $\mathcal{P} := \{P_1, ..., P_r\}$ of sets in a metric space and the goal is to compute a minimum-length tour that visits one element from each set. We study the Euclidean variant of this problem where each P_i is a set of points in $\mathbb{R}^d$ that is contained in a given hypercube H_i. We investigate how the complexity of Euclidean One-of-a-Set tsp depends on λ, the ply of the set $\mathcal{H} := \{H_1, ..., H_r\}$ of hypercubes, and we show that the problem can be solved in $2^{O(\lambda^{1/d+\varepsilon} n^{1-1/d})}$ time for any fixed $\varepsilon > 0$, where $n := \sum_{i=1}^{r} |P_i|$ is the total number of points. Furthermore, we show that the problem cannot be solved in $2^{o(n)}$ time when $\lambda = \Theta(n)$, unless the Exponential Time Hypothesis (ETH) fails.

1 Introduction

In the Traveling Salesman Problem we are given an edge-weighted complete graph and the goal is to compute a tour, i.e., a simple cycle visiting all nodes, of minimum total weight. The Traveling Salesman Problem is among the most famous problems in computer science and combinatorial optimization. One variation is the Euclidean Traveling Salesman Problem. In Euclidean Traveling Salesman Problem the input is a set P of n points in $\mathbb{R}^d$, and the goal is to compute a minimum-length tour visiting each point. This problem was proven to be np-hard in the 1970s [3, 10]. However, unlike the general (metric) version, Euclidean Traveling Salesman Problem in the plane can be solved in *subexponential* time, i.e., in time $2^{o(n)}$. Both Kann [7] and Hwang *et al.* [5] have given algorithms with $n^{O(\sqrt{n})}$ running time. Smith and Wormald [11] gave a subexponential algorithm that works in any (fixed) dimension d, taking $n^{O(n^{1-1/d})}$ time. Recently De Berg *et al.* [2] improved this to $2^{O(n^{1-1/d})}$, which is tight up to constant factors in the exponent, under the Exponential-Time Hypothesis (ETH) [6].

Meanwhile, generalised versions of the Traveling Salesman Problem have also been studied. One popular example is the One-of-a-Set tsp, also known as Generalised Traveling Salesman Problem. Here, the n nodes of the graph are partitioned into sets V_i and the goal is to compute a tour of minimal weight visiting at least (or exactly) one node of every set. Many publications have been made regarding the One-of-a-Set tsp, see for example the survey by Gutin and Punnen [4]. One simple but important result is that One-of-a-Set tsp can be reduced to Traveling Salesman Problem for arbitrary $|V_i|$.

In this paper, we focus on the combination of the two versions, Euclidean One-of-a-Set tsp. For this variation, most research has been focused on the so-called *grid cluster* variant, introduced by Bhattacharya *et al.* [1]. In this variant, a partition is specified by the cells of the integer 1×1 grid (on the Euclidean plane); from every non-empty cell, exactly one point needs to be visited. Khachay and Neznakhina have published many papers on

39th European Workshop on Computational Geometry, Barcelona, Spain, March 29–31, 2023.
This is an extended abstract of a presentation given at EuroCG'23. 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 topic between 2016 and 2020, showing that a PTAS exists if there are many ($O(\log n)$) or few ($n - O(\log n)$) non-empty cells [8], and that if the grid has fixed height or width, a solution can be found in polynomial time [9].

Our contribution. We investigate the complexity of EUCLIDEAN ONE-OF-A-SET TSP. Let $\mathcal{H}$ be a set $\{H_1, ..., H_r\}$ of hypercubes in $\mathbb{R}^d$. Let $\mathcal{P}$ be a family of sets of points $P_1, ..., P_r$ with $P_i \subset \mathcal{H}_i$ and $|P_i| \leq k$ for all i. We will use P to denote $\cup_i P_i$. Let λ be the *ply* of the given hypercubes, i.e., the smallest number such that every point in $\mathbb{R}^d$ is in at most λ of the hypercubes. Our objective is to find a shortest tour $T = (p_1, ..., p_n)$ such that for every P_i there exists a $p \in P_i$ such that $p \in T$. To do so, we will adapt the algorithm by De Berg *et al.* [2]. This results in an algorithm running in $2^{O(\lambda^{1/d+\varepsilon} n^{1-1/d})}$ time, for any fixed $\varepsilon > 0$. Finally, we show that EUCLIDEAN ONE-OF-A-SET TSP in $\mathbb{R}^2$ cannot be solved in time $2^{o(n)}$ when $\lambda = \Theta(n)$, unless ETH fails.

2 A subexponential algorithm for Euclidean One-of-Set TSP

Our separator theorem for EUCLIDEAN ONE-OF-A-SET TSP is similar to the distance-based separator introduced by De Berg *et al.* [2]. Before we can state our result, we need to introduce some terminology and notation from their paper. A *separator* σ is defined to be the boundary of an axis-aligned hypercube. We denote the region of all points in $\mathbb{R}^d$ inside or on σ by σ_{in}, and the region of all points in $\mathbb{R}^d$ strictly outside σ by σ_{out}. The *size* of a separator σ, denoted by $\mathit{size}(\sigma)$, is defined to be its edge length. For a separator σ and a scaling factor $t > 0$, we define $t\sigma$ to be the separator obtained by scaling σ by a factor t with respect to its center. Note that a separator σ induces a partition of the given point set P into two subsets, namely $P \cap \sigma_{\text{in}}$ and $P \cap \sigma_{\text{out}}$. A separator is *balanced* with respect to a set $Q \subseteq P$ if $\max(|Q \cap \sigma_{\text{in}}|, |Q \cap \sigma_{\text{out}}|) \leq \frac{4^d}{4^d+1} n$. If a separator is balanced with respect to P itself, we call it a balanced separator.

The separator used by De Berg *et al.*, which is the same separator we will use, is chosen such that there are only few points close to it. To quantify this, let the *relative distance* from a point p to σ, denoted by $\mathit{rdist}(p, \sigma)$, be defined as follows:

$$\mathit{rdist}(p, \sigma) := d_\infty(p, \sigma)/\mathit{size}(\sigma),$$

where $d_\infty(p, \sigma)$ denotes the shortest ℓ_∞-distance between p and any point on σ. Note that if t is the scaling factor such that $p \in t\sigma$, then $\mathit{rdist}(p, \sigma) = |1 - t|/2$. For integers i define

$$P(i, \sigma) := \{p \in P : \mathit{rdist}(p, \sigma) \leq 2^i / n^{1/d}\}.$$

Note that the smaller i is, the closer to σ the points in $P(i, \sigma)$ are required to be. De Berg *et al.* choose σ such that the size of the sets $P(i, \sigma)$ decrease rapidly as i decreases. Our slightly generalised theorem allows for more control over the dependence of i of the sizes of these sets.

▶ **Theorem 2.1.** *Let P be a set of n points in $\mathbb{R}^d$ and let $Q \subseteq P$. Let $\zeta > 0$ be arbitrary but fixed. Then there is a separator σ that is balanced with respect to Q and such that*

$$|P(i, \sigma)| = \begin{cases} O((3/2)^i n^{1-1/d}) & \text{for all } i < 0 \\ O((2+\zeta)^i n^{1-1/d}) & \text{for all } i \geq 0 \end{cases}$$

Moreover, such a separator can be found in $O(n^{d+1})$ time.

The proof is analogous to the proof of Theorem 1 and Corollary 3 in the original paper by De Berg *et al.* We will now use this distance-based separator theorem to present an efficient algorithm for Euclidean One-of-a-Set tsp.

Let $S(P) := \{pq : (p,q) \in P \times P\}$ be the set of all line segments defined by P. We say a point set P_i is *split* by a separator σ if at least one point of P_i is in σ_{in} and at least one point of P_i is in σ_{out}. We want to find a separator that is crossed only a few times by an optimal TSP tour. Moreover, this separator should split only few of the point sets P_i. Finally, we want to control the number of ways in which we have to "guess" a set of crossing segments. For this De Berg *et al.* uses the so-called *Packing Property*, which is known to hold for the set of edges of an optimal TSP. Intuitively, it states that an optimal tour cannot contain many long edges close together; the precise definition is not important for this paper. Note that the packing property also holds for the edges in an optimal tour for Euclidean One-of-a-Set tsp. Hence, we can restrict our attention to subsets of $S(P)$ with the packing property. For a separator σ, we are thus interested in the following collection of sets of segments crossing σ:

$$\mathcal{C}(\sigma, P) := \{S \subseteq S(P) : S \text{ has the packing property and all segments in } S \text{ cross } \sigma\}.$$

Our main separator theorem, presented next, states that we can find a separator σ that is balanced, splits few P_i, and is such that the sets in $\mathcal{C}(\sigma,P)$, as well as the collection $\mathcal{C}(\sigma,P)$ itself, are small. Since the packing property is hard to test, we will not enumerate $\mathcal{C}(\sigma,P)$ but a slightly larger collection of candidate sets, which we denote by $\mathcal{C}'(\sigma,P)$.

▶ **Theorem 2.2.** *Let $\mathcal{P} = \{P_1, ..., P_r\}$ be a family of point sets in $\mathbb{R}^d$, and let $\mathcal{H} = \{H_1, ..., H_r\}$ be a set of hypercubes such that $P_i \subset H_i$. Let $Q \subseteq P$, where $P = P_1 \cup \dots \cup P_r$. Then, for any fixed $0 < \varepsilon < 1/d$, there exists a separator σ with the following properties:*
1. *σ is balanced with respect to Q.*
2. *Each candidate set $S \in \mathcal{C}'(\sigma,P)$ contains $O(n^{1-1/d})$ segments.*
3. *$\mathcal{C}(\sigma,P) \subseteq \mathcal{C}'(\sigma,P)$ and $|\mathcal{C}'(\sigma,P)| = 2^{O(n^{1-1/d})}$.*
4. *σ splits $O(\lambda^{\frac{1}{d}+\varepsilon} n^{1-1/d})$ of the sets P_i, where λ is the ply of $\mathcal{H}$.*

Moreover, σ and the collection $\mathcal{C}'(\sigma,P)$ can be computed in $2^{O(n^{1-1/d})}$ time.

Proof. The separator chosen is the one found by applying Theorem 2.1 with $\zeta = d\varepsilon$. Hence, the proof of properties 1-3 is analogous to that of the original paper by De Berg *et al.*

It remains to prove that σ splits $O(\lambda^{\frac{1}{d}+\varepsilon} n^{1-1/d})$ of the P_i. Assume without loss of generality that $size(\sigma) = 1$. Let $j = \left\lceil \frac{\log \lambda}{d} \right\rceil$. We can now partition the family of point sets P_i split by σ into two categories, neither of which contains too many sets:

- **P_i split by σ with at least one point in $P(j,\sigma)$.** Since there are $O((2+\zeta)^j n^{1-1/d})$ points in $P(j,\sigma)$, the number of such point sets is limited to

$$O((2+\zeta)^j n^{1-1/d}) = O((2+d\varepsilon)^{\frac{\log\lambda}{d}} n^{1-1/d}) = O(\lambda^{\frac{\log(2+d\varepsilon)}{d}} n^{1-1/d}) = O(\lambda^{1/d+\varepsilon} n^{1-1/d}).$$

- **P_i split by σ with no points in $P(j,\sigma)$.** Such P_i contain at least one point p in σ_{in} at least a distance $2^j/n^{1/d} \geq (\lambda/n)^{1/d}$ from σ. Therefore, at least $((\lambda/n)^{1/d})^{d-1} = (\lambda/n)^{1-1/d}$ of σ is 'covered' by H_i, i.e., the volume vol$(\sigma \cap H_i)$ is at least $(\lambda/n)^{1-1/d}$. Note that H_i is a d-dimensional hypercube, while σ is only the boundary of a d-dimensional hypercube. Furthermore, note that the total ($d-1$-dimensional) volume of σ is $2d$. See Figure 1 for an example. Furthermore, since $\mathcal{H}$ has a ply of λ, we get that $\sum_i \text{vol}(\sigma \cap H_i) \leq \int_\sigma \lambda \mathrm{d}\sigma = 2d\lambda$. In conclusion, there are $O(2d\lambda/(\lambda/n)^{1-1/d}) = O(\lambda^{1/d} n^{1-1/d})$ point sets P_i split by σ with no points in $P(j,\sigma)$.

In total, there are therefore $O(\lambda^{1/d+\varepsilon} n^{1-1/d})$ point sets split by σ. ◀

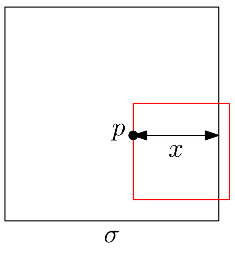


Figure 1 Any hypercube (in red) containing a point p in σ_{in} covers at least x^{d-1} of σ (in black), where x is the distance between p and σ. In this example, the hypercubes are 2-dimensional and σ is 1-dimensional.

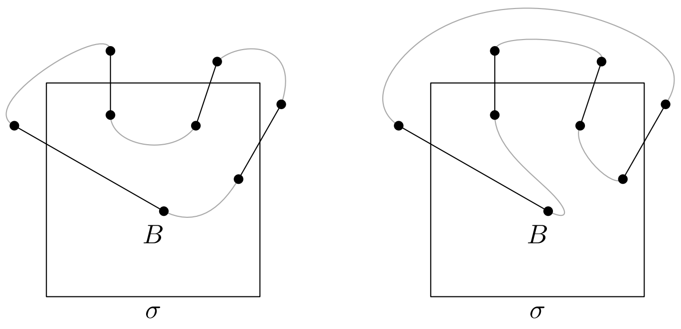


Figure 2 Two different ways to pair up the boundary points B (the endpoints of the segments crossing σ). For both options, the two resulting subproblems can be solved independently.

We are now ready to describe the algorithm itself. We will start with a brief overview of how the original algorithm works. Then, we will explain what changes are made. Finally, we will show that the claimed running time is indeed obtained.

The original algorithm starts by finding a suitable separator σ using its equivalent of Theorem 2.2. Then, it "guesses" how σ is crossed, by simply iterating over all possible candidate sets. By doing so, the problem is split into two subproblems, one containing the points in σ_{in}, and one containing the points in σ_{out}. Note that a candidate set by itself is not enough to create two independent subproblems; the order and direction in which the segments crossing σ are traversed is also important. Specifically, all that matters is how the *boundary points*, the endpoints of the segments crossing σ, are paired up on both sides. See Figure 2 for an example.

Therefore, for every possible matching on the boundary points in σ_{in}, the corresponding Euclidean Path Cover subproblem on all points in σ_{in} is created and solved recursively. The same is done σ_{out}. After solving all versions of both subproblems, the so-called rank-based approach is used to efficiently find the correct combination of matchings on both sides resulting in the shortest overall tour. Finally, we note that always simply taking a separator which is balanced with respect to the whole point set of the subproblem can result in arbitrarily large sets of boundary points; to prevent this from impacting the running time, as soon as the set of boundary points grows too large compared to the total number of points, a separator balanced with respect to the set of boundary points is taken instead.

Our adapted algorithm contains three changes compared to the original. Let $0 < \varepsilon < 1/d$ be arbitrary but fixed. First, we choose our separator σ using Theorem 2.2 instead of the equivalent from the original paper. Note that this does not impact the correctness or running time in any way. Second, candidate sets containing more than one point of any set P_i can be

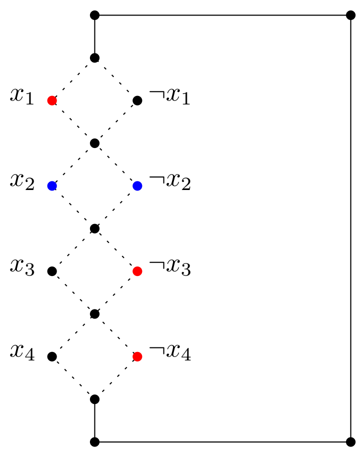


Figure 3 An example for the proof of Theorem 2.4. The unlabeled points all have to be visited. In blue, a point set ensuring we visit x_2 or $\neg x_2$. In red, a point set corresponding to the clause $x_1 \vee \neg x_3 \vee \neg x_4$. A short tour exists if and only if we need to visit only one of each pair $(x_i, \neg x_i)$.

ignored. Finally, when "guessing" the correct candidate set, for every point set P_i split by σ, we also guess whether a point of P_i in σ_{in} or a point in σ_{out} is used. By doing so, the subproblems generated are indeed once more independent. Note that for every boundary point this choice (if applicable) is automatically made.

This brings us to our main theorem, whose proof can be found in the full version of the paper.

▶ **Theorem 2.3.** *Let $\varepsilon > 0$ be arbitary but fixed. Then* Generalised Euclidean TSP *on point sets in d-dimensional hypercubes with ply λ can be solved in $2^{O(\lambda^{\frac{1}{d}+\varepsilon} n^{1-1/d})}$ time.*

Finally, we show that for $\lambda = \Theta(n)$, the problem cannot be solved in subexponential time.

▶ **Theorem 2.4.** *Euclidean One-of-a-Set tsp in $\mathbb{R}^2$ cannot be solved in $2^{o(n)}$ time, unless ETH fails.*

We will give a proof sketch below. For the full proof, see the full version of the paper. We give a reduction from 3-SAT. We map all literals x_i and $\neg x_i$ to points in the plane. We make one point set for each pair of literals, ensuring that for all i, at least one of x_i and $\neg x_i$ is visited. Then, we add points such that a short tour exists if and only if at most one of each x_i and $\neg x_i$ is visited. Finally, we add the clauses of our 3-SAT problem. See Figure 3 for an example.

3 Conclusion

We proved that Euclidean One-of-a-Set tsp on point sets in d-dimensional hypercubes with ply λ can be solved in $2^{O(\lambda^{1/d+\varepsilon} n^{1-1/d})}$ time, for any fixed $\varepsilon > 0$. This implies that when the hypercubes H_i are pairwise disjoint—or, more generally, when the ply of $\mathcal{H}$ is considered to be a constant—then the running time is the same time as for Euclidean Traveling Salesman Problem. Hence, in this case it is optimal, up to constants in the term $O(n^{1-1/d})$ in the exponent [2].

For further research, there are many interesting problems. First of all, we wonder whether a $2^{O(\lambda^{1/d} n^{1-1/d})}$ algorithm can be found, that is, whether the ε can be removed from the exponent. Secondly, we currently measure how 'intertwined' the point sets are by using the ply of hypercubes containing the point sets. As the hypercubes are not used in the algorithm itself, other measuring methods might result in better calculated running times. Finally, instead of using finite point sets P_i, it is interesting to consider a continuous version of the problem, where each set P_i is a region of space (such a ball, hypercube, or polytope).

References

1 Binay Bhattacharya, Ante Ćustić, Akbar Rafiey, Arash Rafiey, and Vladyslav Sokol. Approximation algorithms for generalized mst and tsp in grid clusters. In Zaixin Lu, Donghyun Kim, Weili Wu, Wei Li, and Ding-Zhu Du, editors, *Combinatorial Optimization and Applications*, pages 110–125, Cham, 2015. Springer International Publishing.

2 Mark de Berg, Hans L. Bodlaender, Sándor Kisfaludi-Bak, and Sudeshna Kolay. An eth-tight exact algorithm for euclidean TSP. In *59th IEEE Annual Symposium on Foundations of Computer Science, FOCS*, pages 450–461, 2018. `doi:10.1109/FOCS.2018.00050`.

3 M. R. Garey, Ronald L. Graham, and David S. Johnson. Some NP-complete geometric problems. In *STOC*, pages 10–22. ACM, 1976.

4 G. Gutin and A.P. Punnen. *The Traveling Salesman Problem and Its Variations*. Combinatorial Optimization. Springer US, 2006. URL: `https://books.google.nl/books?id=JBK_BAAAQBAJ`.

5 R. Z. Hwang, R. C. Chang, and Richard C. T. Lee. The searching over separators strategy to solve some NP-hard problems in subexponential time. *Algorithmica*, 9(4):398–423, 1993. `doi:10.1007/BF01228511`.

6 Russell Impagliazzo and Ramamohan Paturi. On the complexity of k-SAT. *Journal of Computer and System Sciences*, 62(2):367–375, 2001. `doi:10.1006/jcss.2000.1727`.

7 Viggo Kann. *On the approximability of NP-complete optimization problems*. PhD thesis, Royal Institute of Technology Stockholm, 1992.

8 Michael Khachay and Katherine Neznakhina. Approximation algorithms for generalized tsp in grid clusters. volume 1623, page 39 – 48, 2016. Cited by: 0. URL: `https://www.scopus.com/inward/record.uri?eid=2-s2.0-85019587518&partnerID=40&md5=63c5e550cfc1fb1682373e7df6cf4651`.

9 Michael Khachay and Katherine Neznakhina. Complexity and approximability of the euclidean generalized traveling salesman problem in grid clusters. *Annals of Mathematics and Artificial Intelligence*, 88(1):53–69, Mar 2020. `doi:10.1007/s10472-019-09626-w`.

10 Christos H. Papadimitriou. The Euclidean traveling salesman problem is NP-complete. *Theor. Comput. Sci.*, 4(3):237–244, 1977.

11 Warren D. Smith and Nicholas C. Wormald. Geometric separator theorems & applications. In *FOCS*, pages 232–243. IEEE Computer Society, 1998. `doi:10.1109/SFCS.1998.743449`.

Shortest coordinated motion for a pair of square robots[*]

Guillermo Esteban[1,2], Dan Halperin[3], Víctor Ruíz[4], Vera Sacristán[4], and Rodrigo I. Silveira[4]

1 **Departamento de Física y Matemáticas, Universidad de Alcalá**
2 **School of Computer Science, Carleton University**
3 **School of Computer Science, Tel Aviv University**
4 **Departament de Matemàtiques, Universitat Politècnica de Catalunya**

Abstract

We study the problem of determining minimum-length coordinated motions for two axis-aligned square robots translating in an obstacle-free plane: Given feasible start and goal configurations, find a continuous motion for the two squares from start to goal, comprising only robot-robot collision-free configurations, such that the total Euclidean distance traveled by the two squares is minimal among all possible such motions. We present an adaptation of the tools developed for the case of discs by Kirkpatrick and Liu [*Characterizing minimum-length coordinated motions for two discs.* Proceedings 28th CCCG, 252–259, 2016.] to the case of squares. Certain aspects of the case of squares are more complicated, requiring additional and more involved arguments over the case of discs. Our contribution can serve as a basic component in optimizing the coordinated motion of two squares among obstacles, as well as for *local planning* in sampling-based algorithms, which are often used in practice, in the same setting.

1 Introduction

The basic motion planning problem is, given start and goal placements for moving objects (robots), to decide whether the objects can move from start to goal without colliding with obstacles in the environment nor with one another, and if so, to plan such a motion. This problem has been intensively investigated for almost five decades now; see, e.g., several books and surveys [1, 3, 4, 7, 9, 11]. The basic problem is relatively well understood, has general theoretical solutions as well as an arsenal of more practical approaches used by practitioners in robotics, molecular biology, animation, computer games, and additional domains where one needs to automatically plan or simulate feasible collision-free motions; see, e.g., [8].

1.1 Optimal motion in the absence of obstacles

Let $\mathbb{A}$ and $\mathbb{B}$ be two axis-aligned square robots in the plane. The position of robot $\mathbb{A}$ (resp., $\mathbb{B}$) at a given moment is denoted by A (resp., B), and refers to the coordinates of its center. We define the *radius* of a square as the length of its apothem. Let $r_{\mathbb{A}}$ and $r_{\mathbb{B}}$ be the radii of robot $\mathbb{A}$ and robot $\mathbb{B}$, respectively.

Given any point X in the plane, we denote by $sq(X)$ the open axis-aligned square, centered at X, with radius $r = r_{\mathbb{A}} + r_{\mathbb{B}}$. We say that a pair of positions (A, B) is *feasible* if $A \notin sq(B)$.

[*] Partially supported by project PID2019-104129GB-I00/MCIN/AEI/10.13039/501100011033 of the Spanish Ministry of Science and Innovation. Work by D.H. has been supported in part by the Israel Science Foundation (grant no. 1736/19), by NSF/US-Israel-BSF (grant no. 2019754), by the Israel Ministry of Science and Technology (grant no. 103129), by the Blavatnik Computer Science Research Fund, and by the Yandex Machine Learning Initiative for Machine Learning at Tel Aviv University.

Notice that this implies that $B \notin sq(A)$. An *instance* of our problem consists of a feasible pair of initial and final positions (A_0, B_0) and (A_1, B_1), respectively. A *trajectory* from a point X_0 to a point X_1 in the plane is any continuous rectifiable curve $m_X : [0,1] \to \mathbb{R}^2$ such that $m_X(0) = X_0$ and $m_X(1) = X_1$. Given an instance of our problem, a *coordinated motion* m is a pair of trajectories $m = (m_A, m_B)$. Throughout this paper we refer to *coordinated motions* simply as *motions.* A motion is *feasible* if for all $t \in [0,1]$ the pair of positions $m(t) = (m_A(t), m_B(t))$ is feasible. We denote the Euclidean arc-length of a trajectory m_X by $\ell(m_X)$. We define the *length* of a motion $m = (m_A, m_B)$ as $\ell(m) = \ell(m_A) + \ell(m_B)$. We focus in this paper on minimum-length coordinated translational motion for two squares. Our goal is to find a description of a minimum-length feasible motion (m_A, m_B), for each instance of the problem. We note that, in the figures of this paper, squares representing robots are drawn with filled color, while squares $sq(X)$ are depicted with a white filling.

Feasible motion for two squares translating among polygonal obstacles with n vertices can be found in $O(n^2)$ time, if one exists [12]. For an arbitrary number of unit squares in the same setting, the problem is known to be PSPACE-hard [13]. Other results for the geometric shortest path problem are reviewed in [10]. Recently, Kirkpatrick and Liu [6] solved the minimum-length coordinated motion for two discs, as we discuss next.

1.2 The Kirkpatrick-Liu analysis for two discs

Kirkpatrick and Liu [6] describe, for any pair of initial and final positions of the discs, two motions that involve at most six (straight or circular-arc) segments. Then, either (i) a single motion is feasible and optimal, or (ii) among the two motions, one is optimal among all clockwise[1] motions and the other is optimal among all counterclockwise motions. The proof of the optimality of the motions involves an extensive case analysis that depends on the relative initial and final positions of the discs. However, all motions have a simple structure:

1. Move robot $\mathbb{A}$ from its initial position to an intermediate position A_{int}.
2. Move robot $\mathbb{B}$ from its initial position to its final position.
3. Move robot $\mathbb{A}$ from the intermediate position A_{int} to its final position.

The main mathematical tool employed in [6] is *Cauchy's surface area formula.* Its use in the context of optimal motion planning was introduced by Icking et al. [5] for a line segment translating and rotating in the plane. The study of the full rigid motion of a segment involves rotation, which raises the question how to measure the distance between two configurations of the moving object. Icking et al. [5] focus on what they call the d_2-distance, which measures the length of the motion of a segment $\overline{pq}$ by averaging the distance travelled by its two endpoints p and q. We remark that the case of a segment has attracted much attention—the interesting history of the problem, as well as other distance measures, are reviewed in [5].

There is a close relation between minimizing the d_2-distance traveled by a segment and the minimum-length coordinated motion of two discs. Assume the sum of the radii of the two discs equals the length $|pq|$ of the segment. Then, if the two discs are osculating throughout the whole motion, the two problems are equivalent.

In this paper, we adapt the techniques from [6] to square robots. Our problem involves several differences because certain results are not applicable to discs, or because some cases

[1] Formally defined in Section 2; roughly, clockwise here refers to the direction of rotation of a vector from the center of one robot to the center of the other robot throughout the motion, from start to goal.

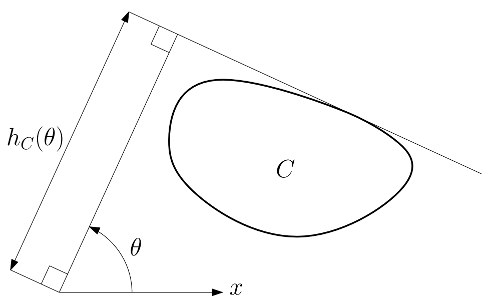


Figure 1 The support function $h_C(\theta)$ of a closed curve C.

and other details were omitted in [6]. First, for squares, the shape and relative position matters. The square corners create discontinuities in tangency points, which impacts the definition of the cases that need to be analyzed. The relative position of the two squares (more precisely, the slope of the line passing through the centers of the two squares) also has to be taken into account. For example, the possible shapes of optimal motions can vary considerably between horizontally-aligned and non-horizontally-aligned squares, giving rise to situations that do not exist for disc robots. Second, when a disc slides along the boundary of another disc, the distance between their centers remains constant, as opposed to the case for squares, where the distance varies with the angle of contact. This forces us to derive new conditions that guarantee the optimality of motions

2 The general approach

In this section we present the general framework used by Kirkpatrick and Liu [6] to prove that a motion is optimal. The *trace* of a trajectory m_X is defined as the image of m_X in the plane. For the remaining of this paper, we will use the notation m_X to refer to a trajectory m_X and its trace. Let $\hat{m}_X$ be the boundary of the convex hull of m_X. The trajectory m_X is said to be *convex* if $\hat{m}_X = m_X \cup \overline{X_0X_1}$, where $\overline{X_0X_1}$ is the segment whose endpoints are $m_X(0) = X_0$ and $m_X(1) = X_1$.

Let $m = (m_A, m_B)$ be a motion from (A_0, B_0) to (A_1, B_1). If both m_A and m_B are convex, and $\ell(\hat{m}_A) + \ell(\hat{m}_B)$ is minimized over all motions from (A_0, B_0) to (A_1, B_1), then m is optimal among all motions from (A_0, B_0) to (A_1, B_1). The convexity of m_A and m_B is easy to verify, but the minimality of $\ell(\hat{m}_A) + \ell(\hat{m}_B)$ is not. In order to facilitate proving optimality, the problem of measuring $\ell(\hat{m}_A) + \ell(\hat{m}_B)$ can be translated into the problem of computing the width of a strip defined by a pair of *supporting lines*, one for $\hat{m}_A$ and one for $\hat{m}_B$ [5]. Given a closed curve C, its *support function* $h_C(\theta)$ can be seen as the distance from the origin to the extremal supporting line of C in the direction $\theta + \pi/2$. See Figure 1 for an illustration. Cauchy's surface area formula [2] implies that if C_1 and C_2 are two closed convex curves, then $\ell(C_1) + \ell(C_2) = \int_0^{2\pi}(h_{C_1}(\theta) + h_{C_2}(\theta + \pi))d\theta$.

We define $h_{m_A}(\theta)$ (resp., $h_{m_B}(\theta)$) to be the support function of $\hat{m}_A$ (resp., $\hat{m}_B$). Then $h_m(\theta) = h_{m_A}(\theta) + h_{m_B}(\theta + \pi)$ can be interpreted as follows. For each $\theta \in S^1$, let $c_{m_A}(\theta)$ be the supporting line for $\hat{m}_A$ in direction $\theta + \pi/2$ and $c_{m_B}(\theta)$ the supporting line for $\hat{m}_B$ in direction $\theta + \pi + \pi/2$. Then $h_m(\theta)$ is the distance between the two supporting lines, as illustrated in Figure 2.

Given two points X, Y, we define $\angle(X, Y)$ as the angle that vector $\overrightarrow{YX}$ forms with the positive x-axis. Let $\theta_0 = \angle(A_0, B_0)$ and $\theta_1 = \angle(A_1, B_1)$. If I is the range of angles $\angle(m_A(t), m_B(t))$ for all $t \in [0, 1]$, then either $[\theta_0, \theta_1] \subseteq I$ or $S^1 \setminus [\theta_0, \theta_1] \subseteq I$ (or both), due

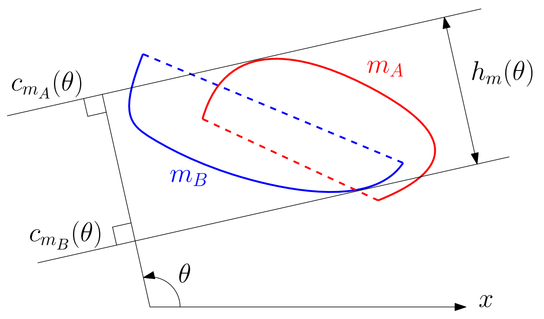


Figure 2 The support function $h_m(\theta)$ of a motion $m = (m_A, m_B)$.

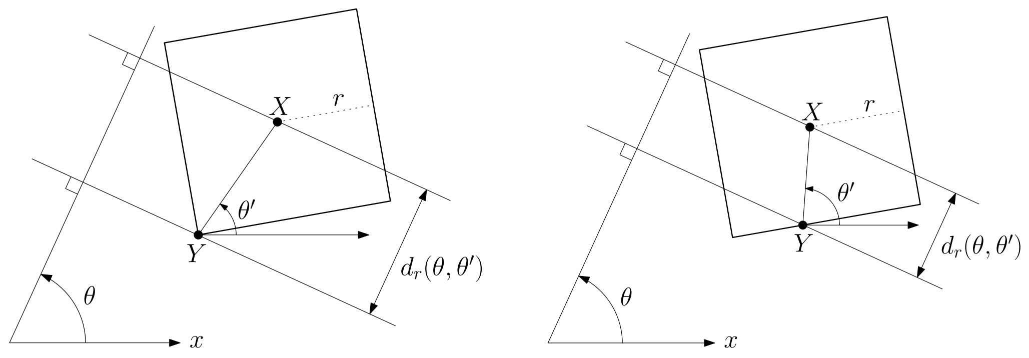


Figure 3 Illustration of $d_r(\theta, \theta')$ for the same value of θ and two different values of θ'.

to the continuity of the trajectories. In the first case, we call m a *counterclockwise* motion. In the second case, we call it *clockwise*. All motions fall in at least one of the two categories, counterclockwise or clockwise. Moreover, a motion can be clockwise and counter-clockwise at the same time. Therefore, our strategy consists in finding a feasible minimum-length motion in each of the two categories, and taking the best of both.

Given $r = r_{\mathbb{A}} + r_{\mathbb{B}}$ and two angles θ and θ', we define $d_r(\theta, \theta')$ as the width of the minimum strip containing any pair of points X and Y such that i) the bounding lines of the strip have slope $\theta + \pi/2$, ii) $\angle(X, Y) = \theta'$ and iii) Y lies on the boundary of $sq(X)$. Refer to Figure 3.

Let $1_{[\theta_0,\theta_1]}$ be the indicator function of the interval $[\theta_0, \theta_1]$, and let $\overline{h}_m(\theta) = \overline{h}_{m_A}(\theta) + \overline{h}_{m_B}(\theta), \theta \in S^1$, where $\overline{h}_{m_A}(\theta)$ is the support function of the segment $\overline{A_0A_1}$, and $\overline{h}_{m_B}(\theta)$ is that of $\overline{B_0B_1}$. Let $LB(\theta) = \max\{\overline{h}_m(\theta), s(\theta) \cdot 1_{[\theta_0,\theta_1]}\}$, where $s(\theta) = \max_{\theta' \in [\theta_0,\theta_1]} d_r(\theta, \theta')$ is defined as the distance between the centers of the robots. We can then observe the following.

▶ **Observation 2.1.** *If $m = (m_A, m_B)$ is a feasible counterclockwise motion from (A_0, B_0) to (A_1, B_1), then $h_m(\theta) \geq LB(\theta)$ for all $\theta \in S^1$.*

An analogous result holds for clockwise motions by replacing $1_{[\theta_0,\theta_1]}$ by $1_{S^1 \setminus [\theta_0,\theta_1]}$ in the definition of function $LB(\theta)$. Finally, we present two conditions guaranteeing that the support function $h_m(\theta)$ coincides with $LB(\theta)$, hence implying that m has minimum length.

▶ **Lemma 2.2.** *Let $m = (m_A, m_B)$ be a motion from (A_0, B_0) to (A_1, B_1). For any angle $\theta \in S^1$, if the support points for $h_m(\theta)$ are A_i and B_j, for $i, j \in \{0, 1\}$, then $h_m(\theta) = LB(\theta)$.*

▶ **Lemma 2.3.** *Let $m = (m_A, m_B)$ be a motion from (A_0, B_0) to (A_1, B_1). For any angle $\theta \in [\theta_0, \theta_1]$, if the support points for $h_m(\theta)$ are one point X of m_A or m_B and a boundary point of its square $sq(X)$, then $h_m(\theta) = LB(\theta)$.*

Recall that by Cauchy's surface area formula, the length of a (convex) motion equals the integral of $h_m(\theta)$ over all angles θ. Lemma 2.2 says that for a particular θ, no motion can have smaller $h_m(\theta)$ than the motion consisting of the two line segments $\overline{A_0A_1}$ and $\overline{B_0B_1}$. Lemma 2.3 applies to the angles where the supporting lines are at a point X for one robot and at the boundary of $sq(X)$ for the other robot. In this case, the distance between the supporting lines has to be at least $s(\theta)$, achieved when one robot is sliding around the other one. Any smaller distance at that angle would imply a collision.

3 The minimum-length motion

In this section we present optimal motions for any initial and final configurations of two square robots, which will be different depending on the relative positions of A_0, A_1, B_0, B_1.

▶ **Theorem 3.1.** *Let $\mathbb{A}$ and $\mathbb{B}$ be two axis-aligned square robots. Let their initial positions be A_0 and B_0, and their final positions be A_1 and B_1, respectively. Up to exchanging the roles of $\mathbb{A}$ and $\mathbb{B}$, there exists a position A_{int} such that the following is a minimum-length feasible motion:*

1. *Move robot $\mathbb{A}$ along the shortest path from A_0 to A_{int} avoiding robot $\mathbb{B}$.*
2. *Move robot $\mathbb{B}$ along the shortest path from B_0 to B_1 avoiding robot $\mathbb{A}$.*
3. *Move robot $\mathbb{A}$ along the shortest path from A_{int} to A_1 avoiding robot $\mathbb{B}$.*

The *corridor* $corr_{\mathbb{A}}$ is the Minkowski sum of the closed line-segment $\overline{A_0A_1}$ and a square $sq(X)$, where X is the origin. The definition of $corr_{\mathbb{B}}$ is analogous. See Figure 4, where $corr_{\mathbb{A}}$ is depicted in red and $corr_{\mathbb{B}}$ in blue. The relative position of A_0, A_1, B_0, B_1 with respect to $corr_{\mathbb{A}}$ and $corr_{\mathbb{B}}$ completely determines the shape of an optimal motion. Up to symmetry, exchanging the roles of $\mathbb{A}$ and $\mathbb{B}$, or exchanging the roles of A_0 by A_1 and B_0 by B_1, we can classify these relative positions in three types:

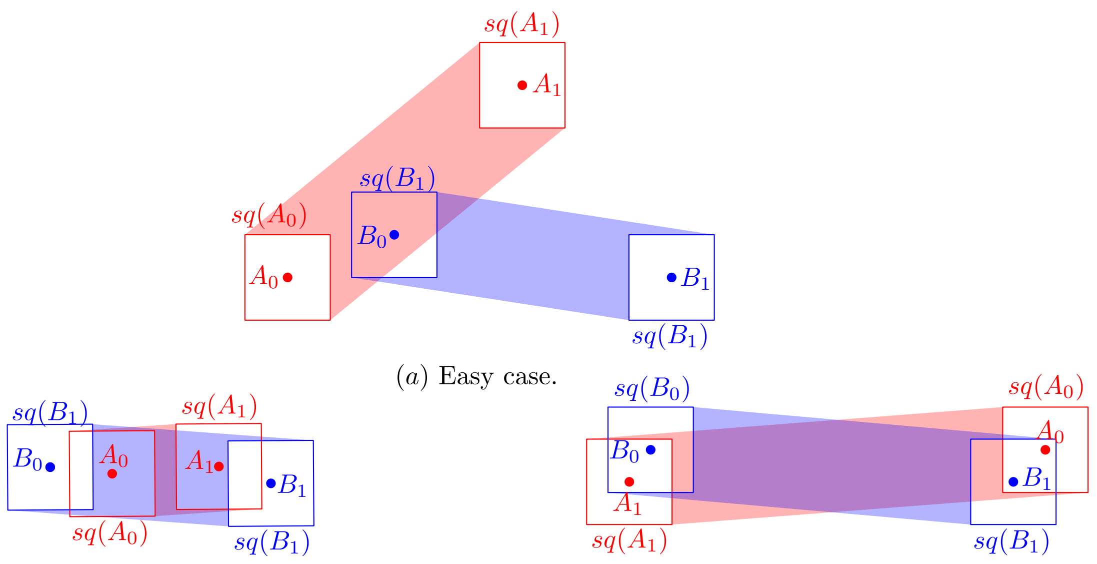
- *Easy*: $A_0 \notin corr_{\mathbb{B}}$ and $B_1 \notin corr_{\mathbb{A}}$. See Figure 4(a).
- *Nested*: $A_0 \in corr_{\mathbb{B}}$, $A_1 \in corr_{\mathbb{B}}$, $B_0 \notin corr_{\mathbb{A}}$ and $B_1 \notin corr_{\mathbb{A}}$. See Figure 4(b).
- *Multi-obstruction*: $A_0 \in corr_{\mathbb{B}}$, $B_0 \in corr_{\mathbb{A}}$. See Figure 4(c).

Due to space limitations, we only present a high-level description of the cases to convey the general nature of our constructions, and defer most of the details to the full version.

For each case, the main challenges are i) defining an intermediate position A_{int} and ii) proving that the motion defined in Section 1.2 is feasible and optimal. In the following, it will be more convenient to assume, without loss of generality, that B_0 and B_1 are horizontally aligned; this can be achieved with a suitable rotation (note that in the figures that follow, squares have been rotated accordingly).

We start by noting that in the easy case, an optimal motion consists of translating each of the robots directly from its initial to its final position along a straight-line segment, but the order might be relevant. In the remaining cases, a straight-line motion is not possible, see, for instance, Figure 5, so we need a finer analysis.

Let $vis(A_0)$ be the region of $corr_{\mathbb{B}} \setminus (sq(B_0) \cup sq(B_1))$ that is visible from A_0. Let $cone(A_0)$ be the set of all points x such that the segment $\overline{A_0x}$ intersects $sq(B_0)$, but not $sq(B_1)$. Let t_{ij} denote the upper tangent line from A_i to $sq(B_j)$, for $i, j = 0, 1$. In the nested case (resp., multi-obstruction) $vis(A_0)$ (resp., $cone(A_0)$) is decomposed into four *zones*, defined by the tangents t_{ij}, see Figures 6 and 7. For each zone we specify a different location for the intermediate point A_{int}. Let p_{ij} be the support point of line t_{ij} in $sq(B_j)$. We say that two tangents t_{i0} and t_{i1} are *twisted* if p_{i1} is to the left of t_{i0}, for $i \in \{0, 1\}$, see Figure 6.



(a) Easy case.

(b) Nested case.

(c) Multi-obstruction case.

Figure 4 Examples of corridors $corr_{\mathbb{A}}$ (red) and $corr_{\mathbb{B}}$ (blue).

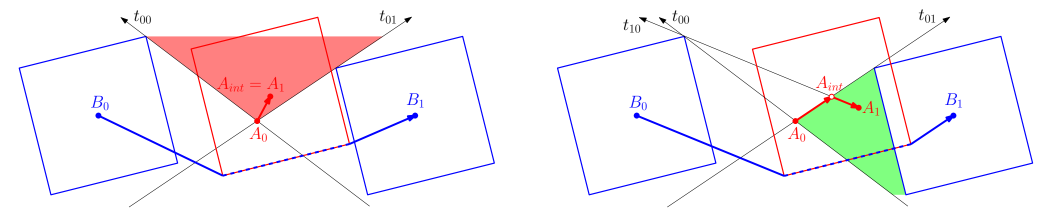


Figure 5 Some optimal motions in the nested case. Left: $A_1 \in$ Zone I. Right: $A_1 \in$ Zone III.

To prove optimality we only examine motions that are counterclockwise; clockwise optimal motions can be obtained by reflecting the initial and final placements across the x-axis, and then examining counterclockwise motions. In the nested case, we differentiate three cases. If the motion is *fully* counterclockwise (i.e., it does not contain clockwise (sub)motion parts), and none of the tangents are twisted, we can verify that the sufficient conditions from Lemmas 2.2 and 2.3 are fulfilled for any relative position of A_0, A_1, B_0, B_1. To that end, we argue that for any angle $\theta \in [\theta_0, \theta_1]$, either both supporting lines are touching A_i and B_j, for $i, j \in \{0, 1\}$, or one is touching a point X of one of the motions, and the other a boundary point of its square $sq(X)$. If the motion is fully counterclockwise, but at least one pair of tangents t_{i0}, t_{i1} is twisted, it might happen that the motion described above is not feasible. Hence, we prove that there exists a clockwise optimal motion that is feasible and globally optimal. Finally, if the motion contains clockwise (sub)motion parts, for some angles, the motion might not satisfy any of the two sufficient conditions from Lemmas 2.2 and 2.3. Therefore, we present an alternative motion m', which is different in the coordination scheme from m, but that is counterclockwise all the time and has exactly the same trace for $\mathbb{A}$ and $\mathbb{B}$ (thus also the same length). By proving that m' is optimal, we obtain that m is optimal as well.

In the multi-obstruction case we show that the motion m is fully counterclockwise. Then, by a reasoning analogous to the nested case, we prove that at least one of the sufficient

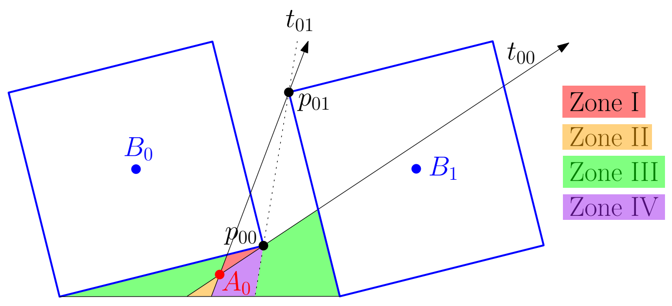


Figure 6 Zones in the nested case when t_{00} and t_{01} are twisted.

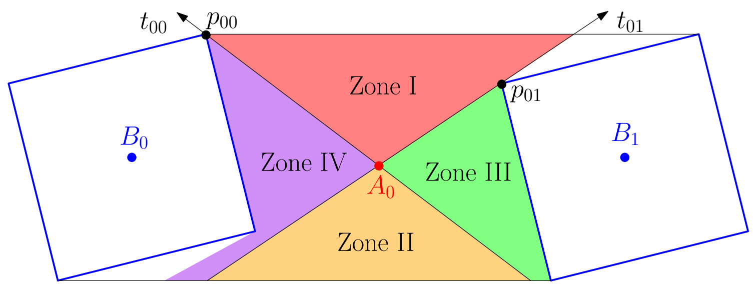


Figure 7 Zones in the nested case when t_{00} and t_{01} are not twisted.

conditions from Lemmas 2.2 and 2.3 is fulfilled for each of the six cases that arise, hence proving the optimality of the motion.

References

1 Howie Choset, Kevin M. Lynch, Seth Hutchinson, George Kantor, Wolfram Burgard, Lydia E. Kavraki, and Sebastian Thrun. *Principles of Robot Motion: Theory, Algorithms, and Implementation.* MIT Press, June 2005.

2 Harold Gordon Eggleston. Convexity, 1966.

3 Dan Halperin, Lydia Kavraki, and Kiril Solovey. Robotics. In Jacob E. Goodman, Joseph O'Rourke, and Csaba Tóth, editors, *Handbook of Discrete and Computational Geometry*, chapter 51, pages 1343–1376. Chapman & Hall/CRC, 3rd edition, 2018.

4 Dan Halperin, Oren Slazman, and Micha Sharir. Algorithmic motion planning. In Jacob E. Goodman, Joseph O'Rourke, and Csaba Tóth, editors, *Handbook of Discrete and Computational Geometry*, chapter 50, pages 1311–1342. Chapman & Hall/CRC, 3rd edition, 2018.

5 Christian Icking, Günter Rote, Emo Welzl, and Chee Yap. Shortest paths for line segments. *Algorithmica*, 10(2):182–200, 1993.

6 David G. Kirkpatrick and Paul Liu. Characterizing minimum-length coordinated motions for two discs. In *Proc. 28th CCCG*, pages 252–259, 2016.

7 Jean-Claude Latombe. *Robot Motion Planning.* Kluwer, Boston, MA, 1991.

8 Jean-Claude Latombe. Motion planning: A journey of robots, molecules, digital actors, and other artifacts. *International Journal of Robotics Research*, 18(11):1119–1128, 1999.

9 Steven M. LaValle. *Planning Algorithms.* Cambridge University Press, 2006.

10 Joseph S. B. Mitchell. Shortest paths and networks. In Jacob E. Goodman, Joseph O'Rourke, and Csaba Tóth, editors, *Handbook of Discrete and Computational Geometry*, chapter 31, pages 811–848. Chapman & Hall/CRC, 3rd edition, 2018.

11 Oren Salzman. Sampling-based robot motion planning. *Commun. ACM*, 62(10):54–63, 2019. URL: `https://doi.org/10.1145/3318164`, `doi:10.1145/3318164`.

12 Micha Sharir and Shmuel Sifrony. Coordinated motion planning for two independent robots. *Ann. Math. Artif. Intell.*, 3(1):107–130, 1991. URL: `https://doi.org/10.1007/BF01530889`, `doi:10.1007/BF01530889`.

13 Kiril Solovey and Dan Halperin. On the hardness of unlabeled multi-robot motion planning. *Int. J. Robotics Res.*, 35(14):1750–1759, 2016. URL: `https://doi.org/10.1177/0278364916672311`, `doi:10.1177/0278364916672311`.

Maintaining Triconnected Components under Node Expansion[*]

Simon D. Fink and Ignaz Rutter

Faculty of Informatics and Mathematics, University of Passau, Germany
`{finksim,rutter}@fim.uni-passau.de`

Abstract

SPQR-trees model the decomposition of a biconnected graph into triconnected components. In this paper, we study the problem of dynamically maintaining an SPQR-tree while expanding vertices into arbitrary biconnected graphs. This allows us to efficiently merge two SPQR-trees by identifying the edges incident to two vertices with each other.

Using efficient expansions and merges allows us to improve the runtime of the Synchronized Planarity algorithm by Bläsius et al. [2] from $O(m^2)$ to $O(m \cdot \Delta)$, where Δ is the maximum pipe degree. This also reduces the time for solving several constrained planarity problems, e.g. for Clustered Planarity from $O((n+d)^2)$ to $O(n + d \cdot \Delta)$, where d is the total number of cluster border–edge crossings and Δ is the maximum number of edges crossing a single cluster border.

Related Version A full version of this paper is available at arXiv:2301.03972.

1 Introduction

The SPQR-tree is a data structure that represents the decomposition of a graph at its *separation pairs*, that is the pairs of vertices whose removal disconnects the graph. The components obtained by this decomposition are called *skeletons*. SPQR-trees form a central component of many graph visualization techniques and are used for, e.g., planarity testing and variations thereof [4, 7] and for computing embeddings and layouts [6, 9]. Initially, SPQR-trees were devised by Di Battista and Tamassia for incremental planarity testing [4]. Their use was quickly expanded to other on-line problems [3] and to the fully-dynamic setting, that is allowing insertion and deletion of vertices and edges in $O(\sqrt{n})$ time [5], where n is the number of vertices in the graph.

In this paper, we consider an incremental setting where we allow a single operation that expands a vertex v into an arbitrary biconnected graph G_ν. The approach of Eppstein et al. [5] allows this in $O((\deg(v) + |G_\nu|) \cdot \sqrt{n})$ time by only representing parts of triconnected components.[1] We improve this to $O(\deg(v) + |G_\nu|)$ using an algorithm that is much simpler and explicitly yields full triconnected components together with an embedding of their skeletons, which will become important for our applications later.

The main idea of our approach is that the subtree of the SPQR-tree affected by expanding a vertex v has size linear in the degree of v, but may contain arbitrarily large skeletons. In a "non-normalized" version of an SPQR-tree, the affected cycle ('S') skeletons can easily be split to have a constant size, while we develop a custom splitting operation to limit the size of triconnected 'R' skeletons. This limits the size of the affected structure to be linear in the degree of v and allows us to perform the expansion efficiently. In Section 2 we describe the datastructure we build upon to apply (and show the correctness of) these operations. Our

[*] Funded by DFG-grant RU-1903/3-1.

[1] Unfortunately, the recent improvements by Holm and Rotenberg are not applicable here, as they maintain triconnectivity in an only incremental setting [8], while maintaining only planarity information in the fully-dynamic setting [7].

39th European Workshop on Computational Geometry, Barcelona, Spain, March 29–31, 2023.

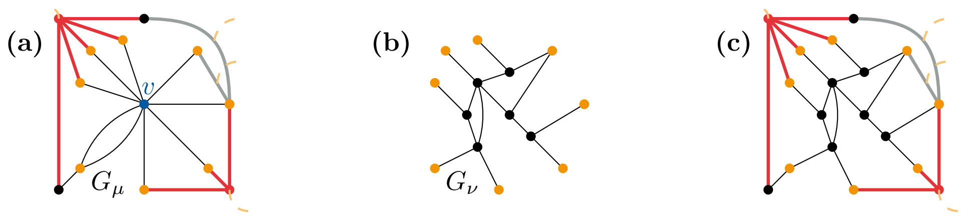


Figure 1 Expanding a vertex v of G_μ into a graph G_ν with the mapping ϕ indicated by orange vertices at the same position, resulting in the graph $G_\mu[v \rightarrow_\phi G_\nu]$. The red and orange dashed edges are only relevant in the context of Section 3, in which the figure can also be interpreted as follows. **(a)** The single allocation skeleton G_μ of u with the single allocation vertex v of u from Figure 5b. The neighbors of v are marked in orange. **(b)** The inserted graph G_ν with orange marked vertices. **(c)** The result of applying `InsertGraph`$(\mathcal{S}, u, G_\nu, \phi)$ followed by an application of `Integrate` on the generated virtual vertices v and v'.

main algorithm is described in Section 3, while Section 4 summarizes our improvements to the runtime of solving Synchronized- and Clustered Planarity.

Preliminaries Let G be a loop-free multi-graph with n vertices $V(G)$ and m edges $E(G)$. We denote the open neighborhood of a vertex v by $N(v)$, excluding v itself. We use $A \uplus B$ to denote the union of two disjoint sets A, B. A *bond* is a graph that consists solely of two *pole* vertices connected by multiple parallel edges, a *polygon* is a simple cycle, while a *rigid* is any simple triconnected graph. A *wheel* is a cycle with an additional central vertex connected to all other vertices. Finally, let G_α, G_β be two graphs, $u \in V(G_\alpha)$ and $M \subset V(G_\beta)$, and ϕ a bijection between $N(u)$ and M. The graph $G_\alpha[u \rightarrow_\phi G_\beta]$ where u in G_α was *expanded* into G_β is obtained from the disjoint union of G_α, G_β by identifying each neighbor x of u with $\phi(x)$ and removing u; see Figure 1 for an example.

2 (Extended) Skeleton Decompositions

A *skeleton structure* $\mathcal{S} = (\mathcal{G}, \mathrm{origV}, \mathrm{origE}, \mathrm{twinE})$ that *represents* a graph $G_\mathcal{S} = (V, E)$ consists of a set $\mathcal{G}$ of disjoint *skeleton* graphs together with three surjective mappings twinE, origE, and origV that satisfy the following conditions:

- Each skeleton $G_\mu = (V_\mu, E_\mu^{\mathrm{real}} \uplus E_\mu^{\mathrm{virt}})$ in $\mathcal{G}$ is a multi-graph where each edge is either in E_μ^{real} and thus called *real* or in E_μ^{virt} and thus called *virtual.*
- Bijection $\mathrm{twinE} : E^{\mathrm{virt}} \rightarrow E^{\mathrm{virt}}$ matches all virtual edges $E^{\mathrm{virt}} = \bigcup_\mu E_\mu^{\mathrm{virt}}$ such that $\mathrm{twinE}(e) \neq e$ and $\mathrm{twinE}^2 = \mathrm{id}$, where id is the identity function.
- Surjection $\mathrm{origV} : \bigcup_\mu V_\mu \rightarrow V$ maps all skeleton vertices to graph vertices.
- Bijection $\mathrm{origE} : \bigcup_\mu E_\mu^{\mathrm{real}} \rightarrow E$ maps all real edges to the graph edge set E.

As the mappings are surjective, V and E are exactly the images of origV and origE. For each vertex $v \in G_\mathcal{S}$, the skeletons that contain an *allocation vertex* v' with $\mathrm{origV}(v') = v$ are called the *allocation skeletons* of v. Furthermore, let $T_\mathcal{S}$ be the graph where each node μ corresponds to a skeleton G_μ of $\mathcal{G}$. Two nodes of $T_\mathcal{S}$ are adjacent if their skeletons contain a pair of virtual edges matched with each other. Figure 2 shows an example of $\mathcal{S}$, $G_\mathcal{S}$, and $T_\mathcal{S}$.

We call a skeleton structure a *skeleton decomposition* if it satisfies the following conditions:

1 **(bicon)** Each skeleton is biconnected.
2 **(tree)** Graph $T_\mathcal{S}$ is simple, loop-free, connected and acyclic, i.e., a tree.
3 **(orig-inj)** For each skeleton G_μ, the restriction $\mathrm{origV}|_{V_\mu}$ is injective.
4 **(orig-real)** For each real edge uv, the endpoints of $\mathrm{origE}(uv)$ are $\mathrm{origV}(u)$ and $\mathrm{origV}(v)$.

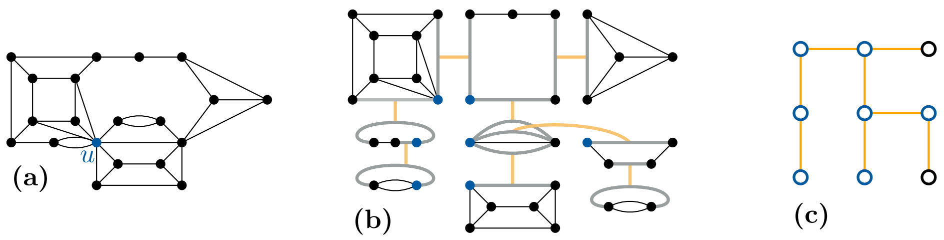


Figure 2 Different views on the skeleton decomposition $\mathcal{S}$. **(a)** The graph $G_{\mathcal{S}}$ with a vertex u marked in blue. **(b)** The skeletons of $\mathcal{G}$. Virtual edges are drawn in gray with their matching twinE being shown in orange. **(c)** The tree $T_{\mathcal{S}}$. Allocation vertices and skeletons of u are marked in blue.

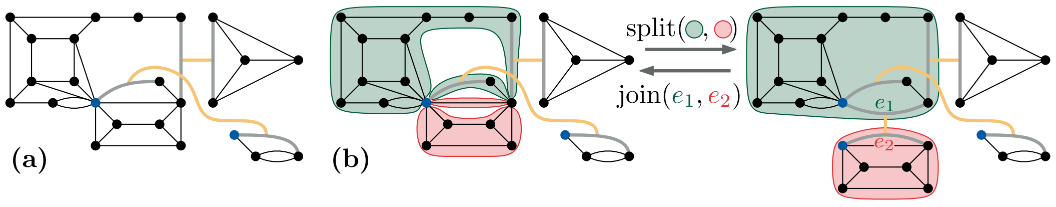


Figure 3 **(a)** The graph from Figure 2a with two arbitrary `SplitSeparationPair` operations already applied. **(b)** Splitting the big skeleton from the same graph along a bipartition into green and red bridges.

5 (orig-virt) Let uv and $u'v'$ be two virtual edges with $uv = \text{twinE}(u'v')$. For their respective skeletons G_μ and G'_μ, it is $\text{origV}(V_\mu) \cap \text{origV}(V_{\mu'}) = \text{origV}(\{u, v\}) = \text{origV}(\{u', v'\})$.

6 (subgraph) The allocation skeletons of any vertex of $G_{\mathcal{S}}$ form a connected subgraph of $T_{\mathcal{S}}$.

To model the decomposition into triconnected components, we define the operations `SplitSeparationPair` and its converse, `JoinSeparationPair`. The former splits a skeleton into two along a given bipartition of the bridges between a given separation pair, while the latter removes a given pair of matched virtual edges by merging their skeletons. Both leave the represented graph unaffected while splitting a node or contracting an edge in $T_{\mathcal{S}}$; see Figure 3 for an example and the full version for formal definitions and proofs. Similar to Angelini et al. [1], we define the unique *SPQR-tree* of $G_{\mathcal{S}}$ as a skeleton decomposition $\mathcal{S} = (\mathcal{G}, \text{origV}, \text{origE}, \text{twinE})$ where any skeleton in $\mathcal{G}$ is either a polygon, a bond, or triconnected ("rigid"), and two skeletons adjacent in $T_{\mathcal{S}}$ are never both polygons or both bonds.

We now define a further set of operations which allow us to isolate vertices out of arbitrary triconnected components by replacing them with a ("virtual") placeholder vertex. Modification of the edges incident to the placeholder is disallowed, which is why we call them "occupied". We keep track of these splits using an *extended* skeleton decomposition $\mathcal{S} = (\mathcal{G}, \text{origV}, \text{origE}, \text{twinE}, \text{twinV})$. Skeletons now have the form $G_\mu = (V_\mu \cup V_\mu^{\text{virt}}, E_\mu^{\text{real}} \cup E_\mu^{\text{virt}} \cup E_\mu^{\text{occ}})$, where the edges in E_μ^{occ} are *occupied*. Bijection $\text{twinV} : V^{\text{virt}} \to V^{\text{virt}}$ matches *virtual vertices* $V^{\text{virt}} = \bigcup_\mu V_\mu^{\text{virt}}$, such that $\text{twinV}(v) \neq v$, $\text{twinV}^2 = \text{id}$. Two virtual vertices matched by twinV induce an edge between their skeletons in $T_{\mathcal{S}}$. Condition 2 (tree) equally applies to these edges, which in particular ensures that there are no parallel twinE and twinV tree edges in $T_{\mathcal{S}}$. Similarly, the connected subgraphs of condition 6 (subgraph) can also contain tree edges induced by twinV. All other conditions remain unchanged, but we add two further conditions to ensure that twinV is consistent:

7 (stars) For each v_α, v_β with $\text{twinV}(v_\alpha) = v_\beta$, it is $\deg(v_\alpha) = \deg(v_\beta)$. All edges incident to v_α and v_β are part of E_μ^{occ} (and thus *occupied*) and have distinct endpoints (except

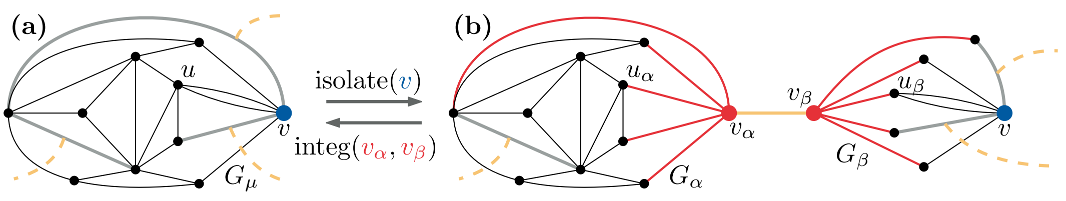


Figure 4 (a) A triconnected skeleton G_μ with a highlighted vertex v incident to two gray virtual edges. The matching of virtual edges is hinted at by orange dashes lines. **(b)** The result of applying `IsolateVertex` to isolate v out of the skeleton. The red occupied edges in the old skeleton G_α form a star with center v_α, while the red occupied edges in G_β connect all neighbors of v to form a star with center $v_\beta \neq v$. The centers v_α and v_β are virtual and matched with each other. Neighbor u of v was split into vertices u_α and u_β.

for v_α and v_β). Conversely, each occupied edge is adjacent to exactly one virtual vertex.

8 (orig-stars) Let v_α and v_β again be two virtual vertices matched with each other by twinV. For their respective skeletons G_α and G_β (where α and β are adjacent in $T_\mathcal{S}$), it is $\text{origV}(V_\alpha) \cap \text{origV}(V_\beta) = \text{origV}(N(v_\alpha)) = \text{origV}(N(v_\beta))$.

Operations `SplitSeparationPair` and `JoinSeparationPair` can also be applied to an extended skeleton decomposition, yielding an extended skeleton decomposition without modifying twinV. To ensure that conditions 7 (stars) and 8 (orig-stars) remain unaffected, `SplitSeparationPair` cannot be applied if a vertex of the separation pair is virtual.

Operations `IsolateVertex` and `Integrate` allow us to isolate vertices out of triconnected components and integrate them back in without changing the represented graph; see Figure 4. To isolate a vertex v, each neighbor $u \in N(v)$ is split into two non-adjacent vertices u_α and u_β, where u_β is incident to all edges connecting u with v, while u_α keeps all other edges of u. We connect all u_α to a single new virtual vertex v_α using occupied edges, and all u_β to a single new virtual vertex v_β using occupied edges. Finally, we match v_α and v_β via twinV. As `Integrate` is the converse of `IsolateVertex` and has no preconditions, any changes made by `IsolateVertex` can be undone at any time to obtain a (non-extended) skeleton decomposition, and thus possibly the SPQR-tree of the represented graph.

3 Node Expansion in Extended Skeleton Decompositions

We now introduce the dynamic operation `InsertGraph`$(\mathcal{S}, u, G_\nu, \phi)$ that changes the represented graph by expanding a single vertex u into an arbitrary connected graph G_ν. This is done by identifying $|N(u)|$ marked vertices in G_ν with the neighbors of u via a bijection ϕ and then removing u and its incident edges; see Figure 1. To ensure that connectivity is not reduced, we require G_ν to be biconnected when all marked vertices are collapsed into a single node. `InsertGraph` itself requires $u \in G_\mathcal{S}$ to have only a single allocation vertex $v \in G_\mu$ (and thus only a single allocation skeleton G_μ). The full procedure $\texttt{InsertGraph}_{\texttt{SPQR}}(\mathcal{S}, u, G_\nu, \phi)$ can be applied to any graph vertex u and, given an SPQR-tree $\mathcal{S}$, yields the SPQR-tree of $G_\mathcal{S}[u \rightarrow_\phi G_\nu]$. It consists of three preparations steps, the insertion of G_ν via `InsertGraph`, and two further clean-up steps:

1. Apply `SplitSeparationPair` so that each polygon allocation skeleton of u has size ≤ 3.
2. Isolate all allocation vertices of u in rigid skeletons by applying `IsolateVertex`.
3. Apply `JoinSeparationPair` to any pair of adjacent allocation skeletons of u. Condition 6 (subgraph) ensures this yields a single component G_μ with size linear in $\deg(u)$ that is

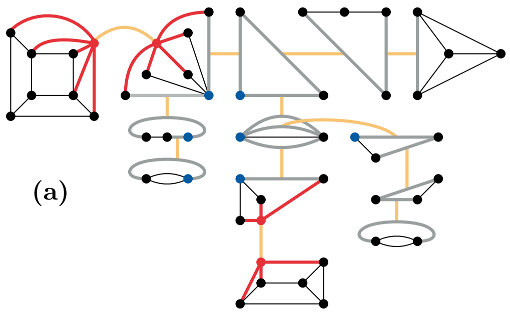


Figure 5 The preprocessing steps of `InsertGraph`$_{\texttt{SPQR}}$ being applied to the SPQR-tree of Figure 2b. **(a)** The state after Step 2, after all allocation skeletons of u have been split. **(b)** The state after Step 3, after all allocation skeletons of u have been merged into a single one.

the sole allocation skeleton of u with the single allocation vertex v of u; see Figure 5.

4. Apply `InsertGraph` to insert G_ν as skeleton, followed by an application of `Integrate` to the virtual vertices $\{v, v'\}$ introduced by the insertion, thus integrating G_ν into G_μ.
5. Apply `SplitSeparationPair` to all separation pairs in G_μ not containing a virtual vertex.
6. Exhaustively apply `Integrate` and also apply `JoinSeparationPair` to any two adjacent polygons and to any two adjacent bonds to obtain the SPQR-tree of the updated graph.

The basic idea behind the correctness of this procedure (which we show in the full version) is that splitting the newly inserted component according to its SPQR-tree in Step 5 yields biconnected components that are each either a polygon, a bond, or "almost" triconnected. The latter (and only those) might still contain virtual vertices and all their remaining separation pairs, which were not split in Step 5, contain one of these virtual vertices. This, together with the fact that there still may be pairs of adjacent skeletons where both are polygons or both are bonds, prevents the instance from being an SPQR-tree. Both issues are resolved in Step 6: The adjacent skeletons are obviously fixed by the `JoinSeparationPair` applications. In the full version, we show that all separation pairs are removed by the `Integrate` applications, making the remaining components triconnected. If the SPQR-tree of the to-be-inserted graph is already known, we can isolate both vertices that should be replaced, identify their neighborhoods and then only spend time linear in the degree of the replaced vertices on clean-up operations. We call this variant `Merge`$_{\texttt{SPQR}}$. Furthermore, it is easy to maintain rotation information (i.e. a cyclic order of all incident edges) for each vertex in a rigid, as each rigid allows exactly two planar embeddings, where one is the reverse of the other [4]. This also allows maintaining a flag indicating whether the graph remained planar (i.e. the rotations of integrated vertices agree with the rigid they are re-integrated to).

▶ **Theorem 3.1.** *SPQR-trees support the operation* `InsertGraph`$_{\texttt{SPQR}}$ *in time* $O(|G_\nu|)$ *and* `Merge`$_{\texttt{SPQR}}$ *in time* $O(\deg(u))$ *while marking non-planar graphs as such. Queries for one of the two possible rotations of vertices in planar triconnected skeletons take constant time.*

By using union-find to keep track of which rigid skeleton vertices belong to, we can also efficiently check whether a pair of vertices is triconnected and provide synchronized rotation information for all vertices in a rigid. Note that the use of union-find leads to the inverse Ackerman function being incorporated as overhead factor to every operation.

4 Application to Synchronized Planarity

We use our datastructure to improve the runtime of the algorithm for solving Synchronized Planarity by Bläsius et al. [2] from $O(m^2)$ to $O(m \cdot \Delta)$, where Δ is the maximum pipe degree (i.e. the maximum degree of a vertex with synchronization constraints that enforce its rotation to be the same as that of another vertex). The algorithm spends a major part of its runtime on computing embedding trees, which describe all possible rotations of a single vertex in a planar graph. Once the embedding trees are available, each of the at most $O(m)$ executed operations runs in time linear in the degree of the pipe it is applied on, that is in $O(\Delta)$ [2]. The quadratic runtime thus only stems from the linear re-computation of embedding trees before each operation. As the SPQR-tree describes all embeddings of the whole graph, it can be used to efficiently derive embedding trees of single vertices. The applied operations might change the underlying graph by expanding vertices, thus one would still need to spend linear time on re-computing SPQR-trees each iteration. Our dynamic data structure can now be used to reduce the runtime of solving Synchronized Planarity by once generating SPQR-trees upfront, maintaining them throughout all applied operations, and deriving any needed embedding tree from an SPQR-tree. The full version provides more backgrounds on the problem, the operations needed to solve it, how embedding trees can be derived from SPQR-trees and how we modify individual operations.

▶ **Theorem 4.1.** Synchronized Planarity *can be solved in time in* $O(m \cdot \Delta)$.

▶ **Corollary 4.2.** Clustered Planarity *can be solved in time in* $O(n + d \cdot \Delta)$, *where* d *is the total number of crossings between cluster borders and edges and* Δ *is the maximum number of edge crossings on a single cluster border.*

References

1 Patrizio Angelini, Thomas Bläsius, and Ignaz Rutter. Testing mutual duality of planar graphs. *International Journal of Computational Geometry & Applications*, 24(4):325–346, 2014. arXiv:1303.1640, doi:10.1142/S0218195914600103.

2 Thomas Bläsius, Simon D. Fink, and Ignaz Rutter. Synchronized planarity with applications to constrained planarity problems. In *Proceedings of the 29th Annual European Symposium on Algorithms (ESA'21)*, volume 204 of *LIPIcs*, pages 19:1–19:14, 2021. doi:10.4230/LIPIcs.ESA.2021.19.

3 G. Di Battista and R. Tamassia. On-line maintenance of triconnected components with SPQR-trees. *Algorithmica*, 15(4):302–318, 1996. doi:10.1007/bf01961541.

4 Giuseppe Di Battista and Roberto Tamassia. On-line planarity testing. *SIAM Journal on Computing*, 25(5):956–997, 1996. doi:10.1137/s0097539794280736.

5 David Eppstein, Zvi Galil, Giuseppe F. Italiano, and Thomas H. Spencer. Separator based sparsification. *Journal of Computer and System Sciences*, 52(1):3–27, 1996. doi:10.1006/jcss.1996.0002.

6 Carsten Gutwenger. *Application of SPQR-trees in the planarization approach for drawing graphs.* PhD thesis, 2010. URL: https://eldorado.tu-dortmund.de/bitstream/2003/27430/1/diss_gutwenger.pdf.

7 Jacob Holm and Eva Rotenberg. Fully-dynamic planarity testing in polylogarithmic time. In Konstantin Makarychev, Yury Makarychev, Madhur Tulsiani, Gautam Kamath, and Julia Chuzhoy, editors, *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing (STOC'20)*, volume abs/1911.03449, pages 167–180. ACM, 2020. arXiv:1911.03449, doi:10.1145/3357713.3384249.

8 Jacob Holm and Eva Rotenberg. Worst-case polylog incremental SPQR-trees: Embeddings, planarity, and triconnectivity. In *Proceedings of the 31st Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'20)*, pages 2378–2397. SIAM, 2020. `doi:10.1137/1.9781611975994.146`.

9 Rene Weiskircher. *New applications of SPQR-trees in graph drawing*. PhD thesis, Universität des Saarlandes, 2002. `doi:10.22028/D291-25752`.

Towards Space Efficient Two-Point Shortest Path Queries in a Polygonal Domain

Sarita de Berg[1], Tillmann Miltzow[1], and Frank Staals[1]

1 Department of Information and Computing Sciences, Utrecht University, The Netherlands
s.deberg@uu.nl, t.miltzow@uu.nl, f.staals@uu.nl

Abstract

We devise a data structure that can answer shortest path queries for two query points in a polygonal domain P on n vertices. For any $\varepsilon > 0$, the space complexity of the data structure is $O(n^{10+\varepsilon})$ and queries can be answered in $O(\log n)$ time. This is the first improvement upon a conference paper by Chiang and Mitchell [8] from 1999. They present a data structure with $O(n^{11})$ space complexity. Furthermore, our main result can be extended to include a space-time trade-off. Specifically, we devise data structures with $O(n^{10+\varepsilon}/\ell^{5+O(\varepsilon)})$ space complexity and $O(\ell \log n)$ query time for any integer $1 \leq \ell \leq n$.

Related Version A full version of the paper is available at `https://arxiv.org/abs/2303.00666`.

1 Introduction

In the two-point shortest path problem, we are given a polygonal domain P with n vertices, and we wish to store P so that given two query points $s, t \in P$ we can compute their *geodesic distance* $d(s, t)$, i.e. the length of a shortest path fully contained in P, in $O(\log n)$ time.

The main motivation to study the two-point shortest path problem is that it is a very natural problem. It is central in computational geometry, and forms a basis for many other problems. The problem was solved optimally for simple polygons (polygonal domains without holes) by Guibas and Hershberger [13], and turned out to be a key ingredient to solve many other problems in simple polygons. A few noteworthy examples are data structures for geodesic Voronoi diagrams [20], furthest point Voronoi diagram [25], k-th nearest neighbor search [1, 11], and more [12, 19]. In real life situations, the environment is often less restricted than a simple polygon. For example, consider a boat in the sea surrounded by a number of islands (Figure 2). Finding the fastest route to an emergency, such as a sinking boat, corresponds to finding the shortest path among obstacles, i.e. in a polygonal domain. This is just one of many examples where finding the shortest path in a polygonal domain is a natural model of a real life situation, which makes it an interesting problem to study.

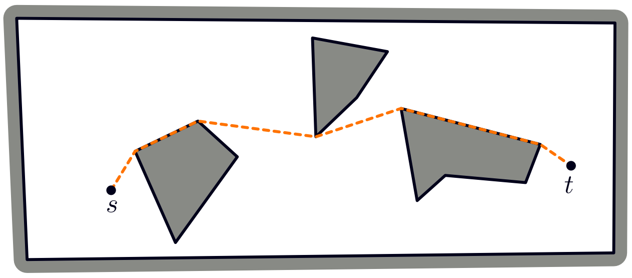


Figure 1 Given P and the query points s, t we want to compute the shortest path efficiently.

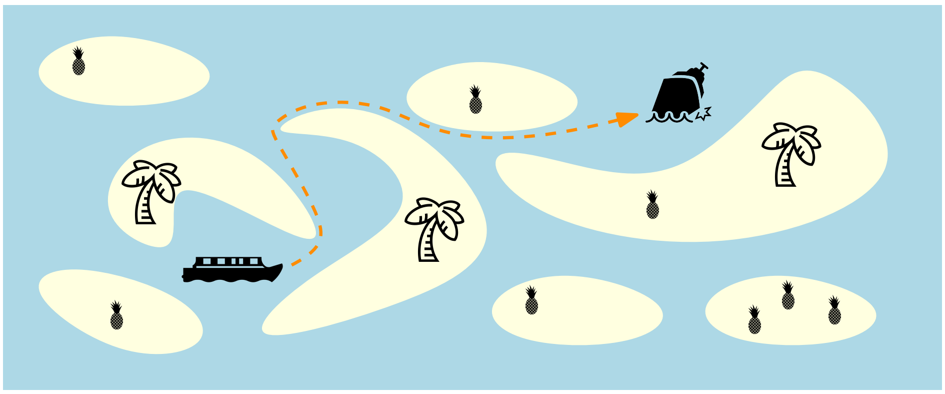


Figure 2 Finding the shortest path among islands for a boat to an emergency.

Related Work. Chiang and Mitchell [8] announced a data structure for the two-point shortest path problem in polygonal domains at SODA 1999. They use $O(n^{11})$ space and achieve a query time of $O(\log n)$. They also present another data structure that uses "only" $O(n^{10} \log n)$ space but $O(\log^2 n)$ query time. Since then, there have been no improvements on the two-point shortest path problem in its general form. Instead, related and restricted versions were considered. We briefly discuss the most relevant ones.

As mentioned before, when the domain is restricted to a simple polygon, there exists an optimal linear size data structure with $O(\log n)$ query time by Guibas and Hershberger [13].

By parameterizing the query time by the number of holes h, Guo, Maheshwari, and Sack [15] manage to build a data structure that uses $O(n^2)$ space and has query time $O(h \log n)$.

Bae and Okamoto [3] study the special case where both query points are restricted to lie on the boundary of the polygonal domain. They present a data structure of size $O(n^4 \lambda_{66}(n)) \approx O(n^5)$ that can answer queries in $O(\log n)$ time.

When we consider the algorithmic question of finding the shortest path between two (fixed) points in a polygonal domain, the state-of-the-art algorithms build the so-called shortest path map from the source s [14, 16]. Hershberger and Suri presented such an $O(n)$ space data structure that can answer shortest path queries from a fixed point s in $O(\log n)$ time [13]. The construction takes $O(n \log n)$ time and space. This was recently improved by Wang [26] to run in $O(n + h \log h)$ time and to use only $O(n)$ working space.

Two other relaxations that were considered are approximation [5, 23], and using the L_1-norm [6, 7, 24].

Results. Our main result is the first improvement in more than two decades that achieves optimal $O(\log n)$ query time.

▶ **Theorem 1.1** (Main Theorem). *Let P be a polygonal domain with n vertices. For any constant $\varepsilon > 0$, we can build a data structure in $O(n^{10+\varepsilon})$ space and expected time that can answer two-point shortest path queries in $O(\log n)$ time. The shortest path of k vertices can be returned in additional $O(k)$ time.*

One of the main downsides of the two-point shortest path data structure is the large space complexity. One strategy to mitigate the space complexity is to allow for a larger query time. For instance, Chiang and Mitchell presented a myriad of different space-time trade-offs. One of them being $O(n^{5+10\delta+\varepsilon})$ space with $O(n^{1-\delta} \log n)$ query time for $0 < \delta \leq 1$. Our methods allow naturally for such a trade-off. We summarize our findings in the following theorem.

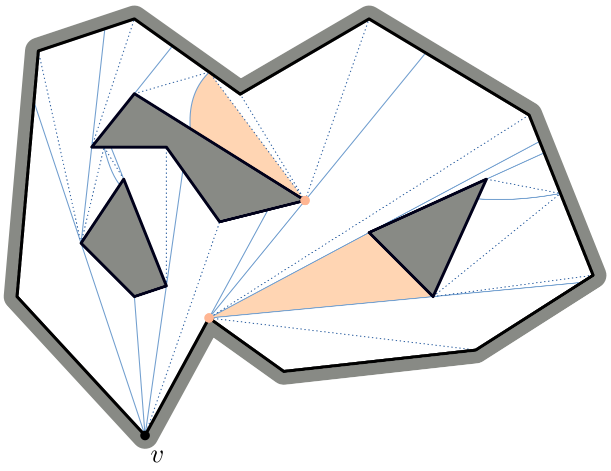


Figure 3 The augmented shortest path map of a vertex v. The shortest path map edges are solid, and the additional edges in the augmented shortest path map are dotted. Each region is bounded by three curves, of which at least two are line segments. Two regions and their apices are highlighted.

▶ **Theorem 1.2.** *Let P be a polygonal domain with n vertices. For any constant $\varepsilon > 0$ and integer $1 \leq \ell \leq n$, we can build a data structure in $O(n^{10+\varepsilon}/\ell^{5+O(\varepsilon)})$ space and expected time that can answer two-point shortest path queries in $O(\ell \log n)$ time.*

For example, for $\ell = n^{4/5}$ we obtain an $O(n^{6+\varepsilon})$ size data structure with query time $O(n^{4/5} \log n)$, which improves the $O(n^{7+\varepsilon})$ data structure with similar query time of [8].

Organization. In Section 2, we give an overview of our main data structure. A full version of the paper is available [10].

2 Global Approach

Direct Visibility. As a first step, we build the visibility complex as described by Pocchiola and Vegter [21]. It allows us to query in $O(\log n)$ time if s and t can see each other. If so, the line segment connecting them is the shortest path. The visibility complex uses $O(n^2)$ space and can be built in $O(n^2)$ time. So, in the remainder we assume that s and t cannot see each other, hence their shortest path will visit at least one vertex of P.

Augmented Shortest Path Maps. In our approach, we build a data structure on the regions provided by the *augmented shortest path maps* of all vertices of P. The shortest path map of a point $p \in P$ is a partition of P into maximal regions, such that for every point in a region R the shortest path to p traverses the same vertices of P [17]. To obtain the augmented shortest path map $SPM(p)$, we connect each boundary vertex of R with the apex v_R of the region, i.e. the first vertex on the shortest path from any point in R towards p. See Figure 3 for an example. All regions in $SPM(p)$ are "almost" triangles; they are bounded by three curves, two of which are line segments, and the remaining is either a line segment or a piece of a hyperbola. The (augmented) shortest path map has complexity $O(n)$ [17]. Let $\mathcal{T}$ be the multi-set of *all* augmented shortest path regions of *all* the vertices of P. As there are n vertices in P, there are $O(n^2)$ regions in $\mathcal{T}$.

Because we are only interested in shortest paths that contain at least one vertex, the shortest path between two points $s, t \in P$ consists of an edge from s to some vertex v of

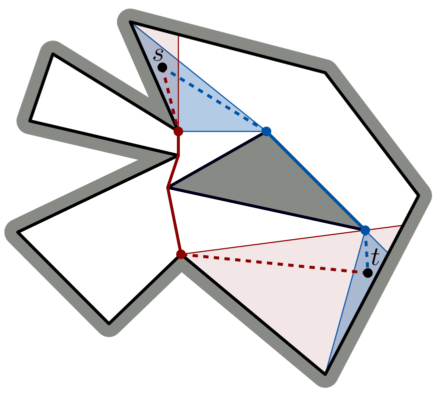


Figure 4 Two pairs of relevant regions in red and blue with the path whose length is $f_{ST}(s,t)$.

P that is visible from s, a shortest path from v to a vertex u (possibly equal to v) that is visible from t, and an edge from u to t. For two regions $S, T \in \mathcal{T}$ with $s \in S$ and $t \in T$, we define $f_{ST}(s,t) = ||sv_S|| + d(v_S, v_T) + ||v_T t||$. The distance $d(s,t)$ between s and t is realised by this function when $v_S = v$ and $v_T = u$. As for any pair S, T with $s \in S$ and $t \in T$ the function $f_{ST}(s,t)$ corresponds to the length of some path between s and t in P, we can obtain the shortest distance by taking the minimum over all of these functions, see Figure 4. In other words, if we denote by $\mathcal{T}_p$ all regions that contain a point $p \in P$, we have

$$d(s,t) = \min\{f_{ST}(s,t) : S \in \mathcal{T}_s, T \in \mathcal{T}_t\}.$$

Lower Envelope. Given two multi-sets $\mathcal{A}, \mathcal{B} \subseteq \mathcal{T}$, we can construct a data structure of size $O(\min\{|\mathcal{A}|, |\mathcal{B}|, n\}^{6+\varepsilon})$ that we can query at any point (s,t) with $s \in \bigcap \mathcal{A}$ and $t \in \bigcap \mathcal{B}$ to find $\min\{f_{ST}(s,t) : S \in \mathcal{A}, T \in \mathcal{B}\}$ in $O(\log(\min\{|\mathcal{A}|, |\mathcal{B}|, n\}))$ time as follows. We refer to this as the Lower Envelope data structure.

The functions f_{ST} are four-variate algebraic functions of constant degree. Each such function gives rise to a surface in $\mathbb{R}^5$, which is the graph of the function f. Koltun [18] shows that the vertical decomposition of m such surfaces in $\mathbb{R}^5$ has complexity $O(m^{6+\varepsilon})$, and can be stored in a data structure of size $O(m^{6+\varepsilon})$ so that we can query the value of the lower envelope, and thus $d(s,t)$, in $O(\log m)$ time. We limit the number of functions $f_{ST}(s,t)$ by using an observation of Chiang and Mitchell [8]. They note that we do not need to consider all pairs $S \in \mathcal{A}, T \in \mathcal{B}$, but only $\min\{|\mathcal{A}|, |\mathcal{B}|, n\}$ *relevant* pairs. Two regions form a relevant pair, if they belong to the same augmented shortest path map $SPM(v)$, of some vertex v. (To be specific, if v is any vertex on the shortest path from s to t, then the minimum is achieved for S and T in the shortest path map of v.) We thus obtain a Lower Envelope data structure by constructing the vertical decomposition of these $\min\{|\mathcal{A}|, |\mathcal{B}|, n\}$ functions.

Naively, to build a data structure that can answer shortest-path queries for any pair of query points s, t, we would need to construct this data structure for all possible combinations of $\mathcal{T}_s$ and $\mathcal{T}_t$. The overlay of the n augmented shortest path maps has worst-case complexity $\Omega(n^4)$ [8], which implies that we would have to build $\Omega(n^8)$ of the Lower Envelope data structures. Indeed, this results in an $O(n^{14+\varepsilon})$ size data structure, and is one of the approaches Chiang and Mitchell consider [8]. Next, we describe how we use cuttings to reduce the number of Lower Envelope data structures we construct.

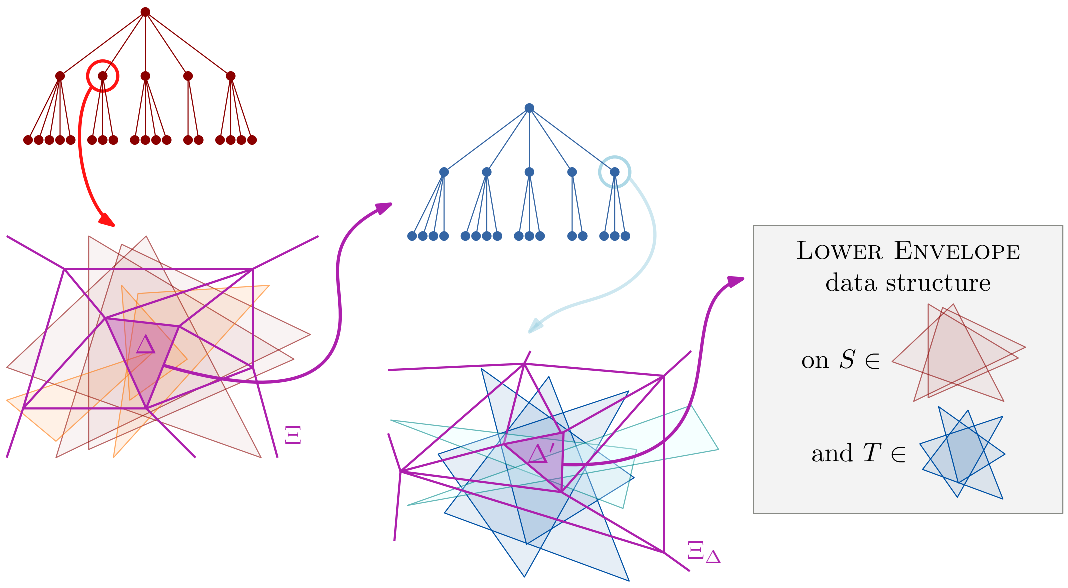


Figure 5 Overview of our data structure. The first level cutting tree (red) is built by recursively constructing a cutting Ξ on the (orange) regions that intersect a cell Δ (purple). For each cell Δ, we store a second level cutting tree (blue). For each cell Δ' in Ξ_Δ, we build a LOWER ENVELOPE data structure on all regions that fully contain Δ (dark red) and Δ' (dark blue).

Cutting Trees. Now, we explain how to determine $\mathcal{T}_s$ more efficiently using cuttings and cutting-trees. Suppose we have a set $\mathcal{A}$ of N (not necessarily disjoint) triangles in the plane. A $1/r$-cutting Ξ of $\mathcal{A}$ is then a subdivision of the plane into constant complexity cells, for example triangles, such that each cell in Ξ is intersected by the boundaries of at most N/r triangles in $\mathcal{A}$ [4]. There can thus still be many triangles that fully contain a cell, but only a limited number whose boundary intersects a cell. In our case, the regions in $\mathcal{T}$ are *almost* triangles, called *Tarski cells* [2]. As we explain in the appendix, we can always construct such a cutting with only $O(r^2)$ cells for these types of regions efficiently.

Let Ξ be a $1/r$-cutting of $\mathcal{T}$. For $s \in \Delta \in \Xi$ the regions $R \in \mathcal{T}$ that fully contain Δ also contain s. To be able to find the remaining regions in $\mathcal{T}_s$, we recursively build cuttings on the N/r regions whose boundary intersects Δ. This gives us a so-called cutting tree. The set $\mathcal{T}_s$ is then the disjoint union of all regions obtained in a root to leaf path in the cutting tree.

The Multi-Level Data Structure. Our data structure is essentially a nested cutting tree, as in [9]. See Figure 5 for an illustration. The first level is a cutting tree that is used to find the regions that contain s, as described before. For each cell $\Delta \in \Xi$ in a cutting Ξ, we construct another cutting tree to find the regions containing t. Let $\mathcal{A}$ be the set of regions fully containing Δ and $|\mathcal{A}| = k$, then the second-level cutting Ξ_Δ is built on the $O(kn)$ candidate relevant regions. See Figure 6. We process the regions intersected by a cell in Ξ_Δ recursively to obtain a cutting tree. Additionally, for each cell $\Delta' \in \Xi_\Delta$, we construct the LOWER ENVELOPE data structure on the sets $\mathcal{A}, \mathcal{B}$, where $\mathcal{B}$ is the set of regions that fully contain Δ'. This allows us to obtain min $f_{ST}(s,t)$ for $S \in \mathcal{A}$ and $T \in \mathcal{B}$ efficiently.

Queries. To query our data structure with two sites s, t, we first locate the cell Δ_s containing s in the cutting Ξ at the root. We compute min $f_{ST}(s,t)$ for all regions S that intersect Δ_s, but do not fully contain Δ_s, by recursively querying the child node corresponding to Δ_s. To

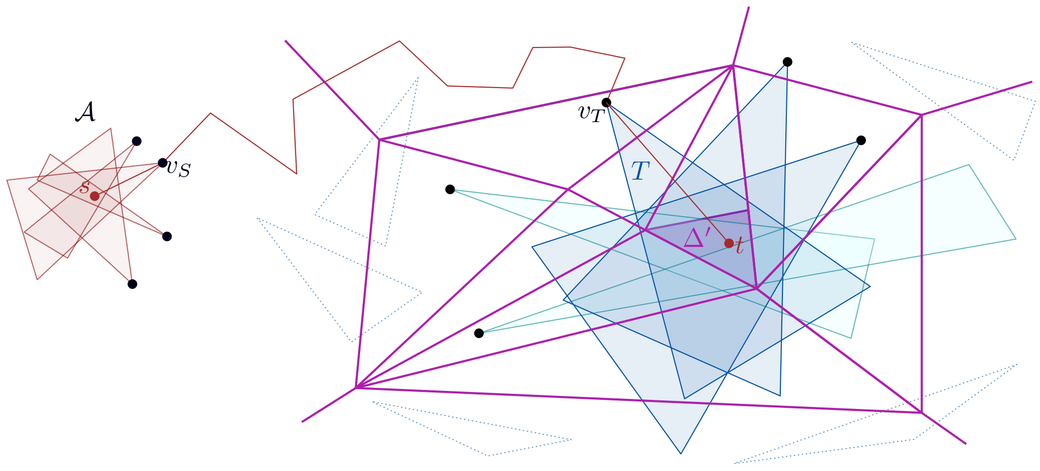


Figure 6 A sketch of the subproblem considered here, computing $\min_{S\in\mathcal{A},T\in\mathcal{T}} f_{ST}(s,t)$. We build a $1/r$-cutting Ξ_Δ (shown in purple) on the set of relevant regions in $\mathcal{T}$ (blue). The regions $\mathcal{T}_t \subseteq \mathcal{T}$ either fully contain the cell $\Delta' \in \Xi_\Delta$ of the cutting that contains t (dark blue), or their boundaries intersect Δ' (light blue).

compute $\min f_{ST}(s,t)$ for all S that fully contain Δ_s, we query its associated data structure. To this end, we locate the cell Δ_t containing t in Ξ_{Δ_s}, and use its lower envelope structure to compute $\min f_{ST}(s,t)$ over all S that fully contain Δ_s and all T that fully contain Δ_t. We recursively query the child corresponding to Δ_t to find $\min f_{ST}(s,t)$ over all regions T that intersect Δ_t.

Sketch of the Analysis. By choosing r as n^δ for some constant $\delta = O(\varepsilon)$, we can achieve that each cutting tree has only constant height. The total query time is thus $O(\log n)$. Next, we sketch the analysis to bound the space usage of the first-level cutting tree, under the assumption that a second-level cutting tree, including the Lower Envelope data structures, uses $O(n^2 \min\{k,n\}^{6+\varepsilon})$ space. The analysis for the second-level cutting tree is similar.

To bound the space usage, we analyze the space used by the *large* levels, where the number of regions is greater than n, and the *small* levels of the tree separately, see Figure 7. There are only $O(n^2)$ large nodes in the tree. For these $\min\{k,n\} = n$, so each stores a data structure of size $O(n^{8+\varepsilon})$. For the small nodes, the size of the second-level data structures decreases in each step, as k becomes smaller than n. Therefore, the space of the root of a small subtree, which is $O(n^{8+\varepsilon})$, dominates the space of the other nodes in the subtree. As there are $O(n^2)$ small root nodes, the resulting space usage is $O(n^{10+\varepsilon})$.

3 Concluding remarks

The Lower Envelope data structure we use is actually more powerful than we require: it allows us to perform point location queries in the vertical decomposition of the entire arrangement, while we are only interested in lower envelope queries. The (projected) lower envelope of m four-variate functions has a complexity of only $O(m^{4+\varepsilon})$ [22]. However, it is unclear if we can store this lower envelope in a data structure of size $O(m^{4+\varepsilon})$ while retaining the $O(\log m)$ query time. We are currently investigating if we can achieve such a bound using kinetic Voronoi diagrams. This would then immediately improve the space usage of our two-point shortest path data structure.

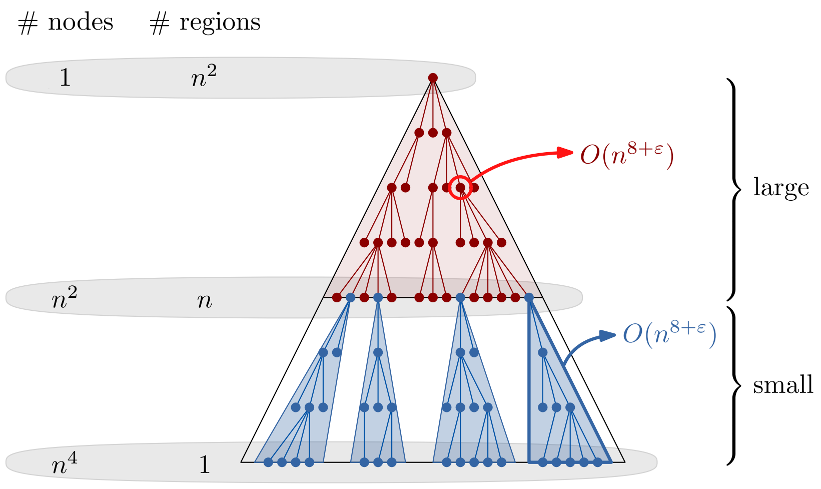


Figure 7 We analyze the large levels, built on $\geq n$ regions, and the small levels, built on $< n$ regions, separately. The total space usage is $O(n^{10+\varepsilon})$.

References

1 Pankaj K. Agarwal, Lars Arge, and Frank Staals. Improved dynamic geodesic nearest neighbor searching in a simple polygon. In *34th International Symposium on Computational Geometry, SoCG*, volume 99 of *LIPIcs*, pages 4:1–4:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018.

2 Pankaj K. Agarwal and Jirí Matoušek. On range searching with semialgebraic sets. *Discret. Comput. Geom.*, 11:393–418, 1994.

3 Sang Won Bae and Yoshio Okamoto. Querying two boundary points for shortest paths in a polygonal domain. *Comput. Geom.*, 45(7):284–293, 2012.

4 Bernard Chazelle. Cutting hyperplanes for divide-and-conquer. *Discret. Comput. Geom.*, 9:145–158, 1993.

5 Danny Z. Chen. On the all-pairs Euclidean short path problem. In *Proceedings of the 6th annual ACM-SIAM symposium on Discrete algorithms, SODA*, pages 292–301, 1995.

6 Danny Z. Chen, Rajasekhar Inkulu, and Haitao Wang. Two-point L1 shortest path queries in the plane. *Journal of Computational Geometry*, 7(1):473–519, 2016.

7 Danny Z. Chen, Kevin S. Klenk, and Hung-Yi T. Tu. Shortest path queries among weighted obstacles in the rectilinear plane. *SIAM Journal on Computing*, 29(4):1223–1246, 2000.

8 Yi-Jen Chiang and Joseph S. B. Mitchell. Two-point Euclidean shortest path queries in the plane. In *Proceedings of the 10th Annual ACM-SIAM Symposium on Discrete Algorithms, SODA*, pages 215–224, 1999.

9 Kenneth L. Clarkson. New applications of random sampling in computational geometry. *Discret. Comput. Geom.*, 2:195–222, 1987.

10 Sarita de Berg, Tillman Miltzow, and Frank Staals. Towards space efficient two-point shortest path queries in a polygonal domain. *CoRR*, abs/2303.00666, 2023.

11 Sarita de Berg and Frank Staals. Dynamic data structures for k-nearest neighbor queries. In *Proceedings of the 32nd International Symposium on Algorithms and Computation, ISAAC*, volume 212 of *LIPIcs*, pages 14:1–14:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021.

12 Patrick Eades, Ivor van der Hoog, Maarten Löffler, and Frank Staals. Trajectory visibility. In *Proceedings of the 17th Scandinavian Symposium and Workshops on Algorithm Theory,* *SWAT 2020*, volume 162 of *LIPIcs*, pages 23:1–23:22. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020.

13 Leonidas J. Guibas and John Hershberger. Optimal shortest path queries in a simple polygon. *J. Comput. Syst. Sci.*, 39(2):126–152, 1989.

14 Leonidas J. Guibas, John Hershberger, Daniel Leven, Micha Sharir, and Robert E. Tarjan. Linear-time algorithms for visibility and shortest path problems inside triangulated simple polygons. *Algorithmica*, 2(1):209–233, 1987.

15 Hua Guo, Anil Maheshwari, and Jörg-Rüdiger Sack. Shortest path queries in polygonal domains. In *Proceedings of the 4th International Conference on Algorithmic Aspects in Information and Management, AAIM*, volume 5034 of *Lecture Notes in Computer Science*, pages 200–211. Springer, 2008.

16 John Hershberger and Jack Snoeyink. Computing minimum length paths of a given homotopy class. *Computational geometry*, 4(2):63–97, 1994.

17 John Hershberger and Subhash Suri. An optimal algorithm for Euclidean shortest paths in the plane. *SIAM J. Comput.*, 28(6):2215–2256, 1999.

18 Vladlen Koltun. Almost tight upper bounds for vertical decompositions in four dimensions. *J. ACM*, 51(5):699–730, 2004.

19 Matias Korman, André van Renssen, Marcel Roeloffzen, and Frank Staals. Kinetic geodesic voronoi diagrams in a simple polygon. In *Proceedings of the 47th International Colloquium on Automata, Languages, and Programming, ICALP*, volume 168 of *LIPIcs*, pages 75:1–75:17. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020.

20 Eunjin Oh. Optimal algorithm for geodesic nearest-point voronoi diagrams in simple polygons. In *Proceedings of the 30th Annual ACM-SIAM Symposium on Discrete Algorithms, SODA*, pages 391–409. SIAM, 2019.

21 Michel Pocchiola and Gert Vegter. The visibility complex. *Int. J. Comput. Geom. Appl.*, 06(03):279–308, 1996. doi:10.1142/S0218195996000204.

22 Micha Sharir. Almost tight upper bounds for lower envelopes in higher dimensions. *Discret. Comput. Geom.*, 12:327–345, 1994.

23 Mikkel Thorup. Compact oracles for approximate distances around obstacles in the plane. In *Proceedings of the 15th Annual European Symposium, ESA*, volume 4698 of *Lecture Notes in Computer Science*, pages 383–394. Springer, 2007.

24 Haitao Wang. A divide-and-conquer algorithm for two-point L1 shortest path queries in polygonal domains. *J. Comput. Geom.*, 11(1):235–282, 2020.

25 Haitao Wang. An optimal deterministic algorithm for geodesic farthest-point voronoi diagrams in simple polygons. In *37th International Symposium on Computational Geometry, SoCG 2021*, volume 189 of *LIPIcs*, pages 59:1–59:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021.

26 Haitao Wang. Shortest paths among obstacles in the plane revisited. In *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms, SODA*, pages 810–821. SIAM, 2021.

Radon number of graph families[*]

Attila Jung[1]

1 **Department of Computer Science, Eötvös Loránd University, Budapest**
jungattila@gmail.com

Abstract

Motivated by Bukh's counterexample to Eckhoff's Partition Conjecture, we define Radon numbers for families of isomorphism classes of graphs and completely characterize families with Radon number at most four in terms of small forbidden subgraphs.

1 Introduction

Radon's Lemma [7] states that whenever we have $X \subset \mathbb{R}^d$ with $|X| \geq d+2$, we can partition X into two disjoint subsets $X^+ \cup X^- = X$ such that $\text{conv}(X^+) \cap \text{conv}(X^-) \neq \emptyset$, where $\text{conv}(S)$ is the convex hull of $S \subset \mathbb{R}^d$. One of its generalizations is Tverberg's Theorem [8], which can be stated as follows. For any $k \geq 2$ if we have $X \subset \mathbb{R}^d$ with $|X| \geq (d+1)(k-1)+1$, then we can partition X into disjoint subsets $X_1 \cup \ldots \cup X_k$ such that $\bigcap_i \text{conv}(X_i) \neq \emptyset$. Eckhoff's Partition Conjecture [2, 3] states that Tverberg's Theorem is a purely combinatorial consequence of Radon's Lemma. For an exact statement, we need to define abstract convexity spaces and their generalized Radon numbers. For an overview of convexity spaces and their invariants see the book by van de Vel [9].

A set X and a family of its subsets $\mathcal{C}$ forms a convexity space if X and $\emptyset$ are members of $\mathcal{C}$, $\mathcal{C}$ is closed under arbitrary intersections, and under union of nested sets. In this sense $\mathbb{R}^d$ and the family of all the convex sets in $\mathbb{R}^d$ forms a convexity space. The convex hull of a subset $S \subseteq X$ in a convexity space is $\text{conv}(S) = \bigcap_{S \subset C \in \mathcal{C}} C$. The kth generalized Radon number of a convexity space $(X, \mathcal{C})$ is the smallest r_k such that any subset $S \subseteq X$ with $|S| = r_k$ can be partitioned into k nonempty disjoint subsets $S = \uplus_{i=1}^k S_i$ with $\bigcap_{i=1}^k \text{conv}(S_i) \neq \emptyset$. If no such number exists, let $r_k = \infty$. We will call $r = r_2$ simply the (not generalized) Radon number. Calder [2] and Eckhoff [3] conjectured $r_k \leq (r-1)(k-1)+1$ for every convexity space.

Upper bounds on r_k can be used to prove the existence of weak epsilon-nets [4]. It was confirmed by Jamison [5] that the conjecture holds if $r_2 = 3$. In general, the best upper bound of r_k in terms of k and $r = r_2$ is $r_k \leq (2k)^{\log_2 r}$ by Jamison [5] and $r_k \leq kr^{r^{r^{\log_2 r}}}$ by Pálvölgyi [6]. However, after being open for more than thirty years, the conjecture itself was refuted by Bukh [1], as he constructed convexity spaces with $r = 4$ and $r_k \geq 3k-1$.

Bukh first constructs a violating subset S, which shows that $r_k \geq 3k-1$, and decides which convex hulls of subsets of S intersect. Every intersection can be described using a hypergraph with the base set S, with the resulting hypergraph family satisfying certain natural conditions. After adding a new point to every required intersection, the (not generalized) Radon number of the resulting convexity space depends on the following property.

▸ **Definition 1.1.** A family $\mathcal{H}$ of hypergraphs has Radon number at most r, if for every $H_1, \ldots, H_r \in \mathcal{H}$, there is a partition $I \uplus J = [r]$ into nonempty subsets such that $\left(\bigcap_{i \in I} H_i\right) \bigcup \left(\bigcap_{j \in J} H_j\right) \in \mathcal{H}$.

[*] Research was supported by the Rényi Doctoral Fellowship and by the NKFIH grant FK132060.

To see how this property arises from a convexity space with bounded Radon number, let H_x, H_y, H_z and H_w be collections of all the convex sets containing the points x, y, z and w respectively. In this case $\operatorname{conv}(\{x,y\})$ is the intersection of all the convex sets in $H_x \cap H_y$ and $\operatorname{conv}(\{x,y\}) \cap \operatorname{conv}(\{z,w\})$ is the intersection of all the convex sets in $(H_x \cap H_y) \cup (H_z \cap H_w)$. In this paper, we investigate the property described in Definition 1.1 in the case when $\mathcal{H}$ is a family of graphs with the hope that it will contribute to a better understanding of Radon numbers of convexity spaces.

The paper is organized as follows. In Section 2 we define the Radon number for graph families and give some general results. In Section 3 we show that graph families with Radon number at most three are trivial, and in Section 4 we state and prove our main result, Theorem 4.1, which characterizes graph families with Radon number four. Section 5 contains open questions. Due to space restrictions, some proofs will only be available in the final version of the paper.

2 Radon number of graph families

In the rest of the paper, we will consider graph families which are closed under any isomorphism of any graphs in the family and are also closed under taking subgraphs. Since the considered graph families are closed under isomorphisms of individual graphs, we have to fix a vertex set as the domain of these isomorphisms. In this paper, we assume that the underlying vertex set is finite but large enough. We only consider simple graphs and identify graphs with their edge sets. We will denote the number of edges in a graph G by $|G|$.

▸ **Definition 2.1.** A family $\mathcal{G}$ of graphs has Radon number at most r, if no matter how we choose $G_1, \dots, G_r \in \mathcal{G}$, there is a partition $I \uplus J = [r]$ into nonempty subsets such that $\left(\bigcap_{i\in I} G_i\right) \bigcup \left(\bigcap_{j\in J} G_j\right) \in \mathcal{G}$.

We will call $\left(\bigcap_{i\in I} G_i\right) \bigcup \left(\bigcap_{j\in J} G_j\right)$ a Radon-major of $G_1, \dots, G_r$. The smallest number $r(\mathcal{G})$ such that $\mathcal{G}$ has Radon number at most r is the Radon number of the family $\mathcal{G}$.

▸ **Example 2.2.** If $\mathcal{G}$ consists of all the n-cliques and their subgraphs, then $r(\mathcal{G}) = n+1$.

Proof. First, we show that $r(\mathcal{G}) \le n+1$. Let $G_1, \dots, G_n$ be cliques on n vertices and consider the sequence $\cap_{i=1}^{j} V(G_i)$ for $1 \le j \le n+1$, where $V(G_i)$ is the vertex set of G_i. If there exists a j with $\cap_{i=1}^{j} V(G_i) = \cap_{i=1}^{j+1} V(G_i)$, then $\cap_{i=1}^{j} G_i \subset G_{j+1}$ and thus $G = (\cap_{i=1}^{j} G_i) \cup (\cap_{i=j+1}^{n+1} G_i) \subset G_{j+1}$, which implies $G \in \mathcal{G}$. Otherwise we have $|\cap_{i=1}^{n} V(G_i)| \le 1$, and thus $(\cap_{i=1}^{n} G_i) \cup G_{n+1} = G_{n+1} \in \mathcal{G}$.

To show that $r(\mathcal{G}) > n$, let $G_1, \dots, G_n$ be different n-cliques on base set $[n+1]$. For every partition $I \uplus J = [n+1]$, the graph $G = (\cap_{i\in I} G_i) \cup (\cap_{j\in J} G_j)$ has $n+1$ vertices, but none of them is isolated. Thus, G is not a subgraph of an n-clique. ◂

Our main tool in considering Radon numbers of graph families will be an analog of Helly's Theorem. We need two preliminary definitions to state it.

▸ **Definition 2.3.** The h-Helly closure $\mathcal{H}_h(\mathcal{G})$ of a graph family $\mathcal{G}$ consists of all graphs H for which every at most h-edge subgraph of H is an element of $\mathcal{G}$.

As an example, if $\mathcal{G}$ consists of all the stars with at most h edges, then its h-Helly closure consists of all the stars with an arbitrary number of edges.

▸ **Definition 2.4.** We say that a graph family $\mathcal{G}$ implies a graph G under the condition $r(\mathcal{G}) \le t$ if every graph family $\mathcal{G}' \supseteq \mathcal{G}$ with $r(\mathcal{G}') \le t$ contains G as well. In notation $\mathcal{G} \xrightarrow{r\le t} G$. We also write $\mathcal{G} \xrightarrow{r\le t} \mathcal{F}$ for a graph family $\mathcal{F}$, if $\mathcal{G} \xrightarrow{r\le t} F$ for every $F \in \mathcal{F}$.

The following is an analog of Helly's Theorem for graph families.

▸ **Lemma 2.5** (Helly Property). *For every graph family $\mathcal{G}$ and every integer $h > 0$ we have*

$$\mathcal{G} \xrightarrow{r \leq h+1} \mathcal{H}_h(\mathcal{G}).$$

Proof. The proof of Lemma 2.5 mimics Radon's proof [7] of Helly's Theorem. Let $\mathcal{G}'$ be a family containing $\mathcal{G}$ and satisfying $r(\mathcal{G}') \leq h + 1$. We will prove $H \in \mathcal{G}'$ for every $H \in \mathcal{H}_h(\mathcal{G})$ by induction on the number of edges in H. If H has h elements, then it is equal to some $G \in \mathcal{G}$ and hence a member of $\mathcal{G}'$. For the induction step, assume that H has more than h edges and that every proper subgraph of H is a member of $\mathcal{G}'$. Let $H_1, \dots, H_{h+1}$ be different subgraphs of H, each containing exactly $|H| - 1$ edges. For every $I \uplus J = [h+1]$, we have $\left(\bigcap_{i \in I} H_i\right) \bigcup \left(\bigcap_{j \in J} H_j\right) = H$, showing that $\mathcal{G}'$ must contain H as well. ◂

▸ **Lemma 2.6.** *If $r(\mathcal{G}) \leq t$ for a graph family $\mathcal{G}$, and a graph family $\mathcal{G}'$ consists of graphs with at most $t - 2$ edges, then $r(\mathcal{G} \cup \mathcal{G}') \leq t$ as well.*

Proof. Let $G_1, \dots, G_t \in \mathcal{G} \cup \mathcal{G}'$. If there is any G_i with at most $t - 2$ edges, then there are two different $I, I' \subset [t]$ with $|I| = |I'| = t - 1$, $i \in I \cap I'$ and $\cap_{j \in I} G_j = \cap_{j \in I'} G_j$ by the pigeonhole principle. In particular, $\cap_{j \in I} G_j \subset G_k$ if $\{k\} = [t] \setminus I$ and $\cap_{j \in I} G_j \cup G_k \subseteq G_k \in \mathcal{G} \cup \mathcal{G}'$.

Otherwise $G_1, \dots, G_t \in \mathcal{G}$ and there exists $I \uplus J = [t]$ with $(\cap_{i \in I} G_i) \cup (\cap_{j \in J} G_j) \subset G \in \mathcal{G}$. ◂

▸ **Corollary 2.7.** *Every graph family $\mathcal{G}$ with $r(\mathcal{G}) = t$ is a union of a family of graphs with at most $(t-2)$ edges and a $(t-1)$-Helly closure of a family of graphs with exactly $t - 1$ edges.*

3 Graph families with Radon number at most 3 are trivial

▸ **Claim 3.1.** *If $r(\mathcal{G}) = 2$ for a graph family, then either $\mathcal{G}$ is the empty family or $\mathcal{G}$ contains the empty graph with no edges or $\mathcal{G}$ contains all the graphs.*

Proof. Graph families with Radon number 2 are not only closed under isomorphism of graphs and taking subgraphs (as all the considered families), but also under union of graphs. ◂

▸ **Claim 3.2.** *If $r(\mathcal{G}) = 3$ for a graph family $\mathcal{G}$, then $\mathcal{G}$ consists of all the graphs with exactly one edge.*

Proof. If $\mathcal{G}$ contains only graphs with exactly one edge, then $r(\mathcal{G}) = 3$. Suppose $\mathcal{G}$ contains one of the two different non-isomorphic simple graphs with exactly two edges, a path (denoted from now by the symbol $\wedge$), or a graph consisting of two disjoint edges (denoted by $=$). We show that any of them contains the other in its 2-Helly closure, thus $\mathcal{G}$ must contain all the graphs.

If we have $\wedge \in \mathcal{G}$, then every triangle is a member of $\mathcal{G}$ by Lemma 2.5. Let G_i be the triangle with vertices $\{i\} \setminus \{1, 2, 3, 4\}$ and consider G_1, G_2, G_3. Every Radon-major of them is isomorphic to $G_1 \cup (G_2 \cap G_3)$, which contains two disjoint edges. Thus, $=$ must also be a member of $\mathcal{G}$, since $\mathcal{G}$ is closed under taking subgraphs.

If have $= \in \mathcal{G}$, we can use $G_1 = \{12, 45, 78\}$, $G_2 = \{12, 56, 89\}$ and $G_3 = \{23, 45, 89\}$ to show that $\wedge$ must also be a member of $\mathcal{G}$ (we denote edges of the form $\{i, j\}$ simply by ij). ◂

4 Graph families with Radon number 4

We will use the following symbols for the five isomorphism classes of graphs with exactly three edges: $\equiv, \text{⅄}, \triangle, \bumpeq, \sqcap$. The first denotes a graph with three disjoint edges, the second a star with three edges, the third a triangle, the fourth a disjoint union of a path with two edges and an edge, and the fifth a path with three edges.

▸ **Theorem 4.1.** *If a graph family $\mathcal{G}$ with Radon number at most 4 contains any of the four families*

$$\{\equiv\}, \{\text{⅄}, \bumpeq\}, \{\text{⅄}, \triangle, \sqcap\}, \{\triangle, \bumpeq, \sqcap\},$$

then it contains all the graphs. If a graph family contains none of the above-listed families, then its 3-Helly closure has Radon number at most 4.

We prove the first part of Theorem 4.1 in Subsection 4.1 and we sketch the proof of the second part in Subsection 4.2.

4.1 Implications

Proof of the first part of Theorem 4.1. If $\mathcal{G}$ contains one of the listed four families and has Radon number at most 4, then it must contain the 3-Helly closure of that family by Lemma 2.5. We will show that much more is true, and $\mathcal{G}$ must contain all other graphs with at most three edges under the assumption that $r(\mathcal{G}) \le 4$. In the proofs, we will repeatedly find four graphs in the 3-Helly closure of the previously guaranteed graphs in $\mathcal{G}$ with the property that all the Radon-majors of them contain at least one new 3-edge subgraph.

Consider the seven different 2-partitions of $\{1,2,3,4\}$, namely 1|234, 2|134, 3|124, 4|123, 12|34, 13|24 and 14|23. Let $I_1, J_1, \dots, I_7, J_7$ be the subsets of these partitions in the previous order. For example $I_1 = \{1\}$ or $J_7 = \{2,3\}$.

If $\equiv$ is in $\mathcal{G}$, consider seven vertex-disjoint copies of $\wedge$, each corresponding to a different 2-partition of $\{1,2,3,4\}$. Name the two edges of the kth copy as $\{e_i^{(k)}, e_j^{(k)}\}$ and let $G_1, \dots, G_4$ be such that $G_\ell = \{e_i^{(k)} : \ell \in I_k\} \cup \{e_j^{(k)} : \ell \in J_k\}$. All the G_is are part of the 3-Helly closure of $\{\equiv\}$ and every Radon-major of them contains $\bumpeq$ as a subgraph, thus $\{\equiv\} \xrightarrow{r\le4} \bumpeq$. Similar constructions show that $\{\equiv, \bumpeq\} \xrightarrow{r\le4} \triangle, \text{⅄}$ and $\sqcap$. Thus, if $r(\mathcal{G}) \le 4$ and $\equiv \in \mathcal{G}$, then $\mathcal{G}$ contains all the graphs with at most three edges. But every graph is in their 3-Helly closure, so $\mathcal{G}$ must contain all the graphs by Lemma 2.5.

If we have $\{\text{⅄}, \bumpeq\} \subset \mathcal{G}$, then the graphs $G_\ell = \{\{0,k\} : \ell \in I_k\} \cup \{\{k,8\} : \ell \in J_k\}$ for $\ell \in \{1,2,3,4\}$ show that $\{\text{⅄}, \bumpeq\} \xrightarrow{r\le4} \sqcap$. Now the graphs $G_i : i \in \{1,2,3,4\}$ with G_i being the complete bipartite graph between $\{i,5\}$ and $\{i,5\} \setminus \{1,\dots,5\}$ show that $\{\text{⅄}, \bumpeq, \sqcap\} \xrightarrow{r\le4} \triangle$. From here four different five-clique on $\{1,2,\dots,6\}$ shows that $\{\text{⅄}, \triangle, \sqcap, \bumpeq\} \xrightarrow{r\le4} \equiv$.

Four different four-cliques on $\{1,2,3,4,5\}$ shows us that $\{\text{⅄}, \triangle, \sqcap\} \xrightarrow{r\le4} \bumpeq$ and from there we have seen $\{\text{⅄}, \triangle, \sqcap, \bumpeq\} \xrightarrow{r\le4} \equiv$.

Finally, if we have $\{\triangle, \bumpeq, \sqcap\} \subset \mathcal{G}$, then we can take G_1 to be a five cycle with vertices $1,2,3,4,5$, G_2 a disjoint union of two triangles with vertices $1,2,3$ and $4,5,6$, G_3 a path with five vertices $2,3,4,6,5$ and G_4 another path with five vertices $1,3,4,5,6$, showing $\{\triangle, \bumpeq, \sqcap\} \xrightarrow{r\le4} \text{⅄}$. We are done, since we have seen that $\{\text{⅄}, \triangle, \sqcap, \bumpeq\} \xrightarrow{r\le4} \equiv$. ◂

4.2 Radon-closed families

To show that a certain graph family $\mathcal{G}$ has Radon number 4, we have to exclude the possibility of the existence of $G_1, \dots, G_4 \in \mathcal{G}$ without a Radon-major in $\mathcal{G}$. The following claims show some necessary properties of such quadruples of graphs. Due to space restrictions, the proofs of corollaries will be available only in the final version of this paper.

▸ **Claim 4.2.** *If $G_1, \dots, G_4 \in \mathcal{G}$ has no Radon-major in $\mathcal{G}$, then the 3-wise intersections of them must be nonempty and distinct.*

Proof. If a three-wise intersection is empty, for example $G_1 \cap G_2 \cap G_3 = \varnothing$, then G_4 is a Radon major, as $(G_1 \cap G_2 \cap G_3) \cup G_4 = G_4$.

If two of the three-wise intersections are equal, for example $G_1 \cap G_2 \cap G_3 = G_2 \cap G_3 \cap G_4$, then G_4 is a Radon-major, as $(G_1 \cap G_2 \cap G_3) \cup G_4 = (G_2 \cap G_3 \cap G_4) \cup G_4 = G_4$. ◂

▸ **Claim 4.3.** *If $G_1, \dots, G_4$ are arbitrary graphs, then $\bigcup_{i=1}^{4}(\bigcap_{j \neq i} G_j) \in \mathcal{H}_3(\{G_1, \dots, G_4\})$.*

Proof. If we choose 3 edges from the union of the 3-wise intersections, there will be a G_i among $G_1, \dots, G_4$, which contains all the 3 edges. ◂

▸ **Corollary 4.4.** *If $\mathcal{G}$ is one of $\{\text{⅄}\}, \{\text{⅄}, \sqcap\}$ or $\{\text{⅄}, \triangle\}$, then $r(\mathcal{H}_3(\mathcal{G})) = 4$.*

Given four graphs $G_1, \dots, G_4$ and subset $I \subset \{1,2,3,4\}$, we will say that an edge $e \in \cup_{i=1}^{n} G_i$ is I-type, if $e \in \bigcap_{i \in I} G_i$ but for every $J \supsetneq I$ we have $e \notin \bigcap_{j \in J} G_j$.

▸ **Claim 4.5.** *If $G_1, \dots, G_4 \in \mathcal{H}$ has no Radon-major in a 3-Helly closure $\mathcal{H}$ of a graph family, then there is either a $\{1,2\}$-type edge or a $\{3,4\}$-type edge in $\cup_i G_i$.*

Proof. If there is no $\{1,2\}$-type edge, then $G_1 \cap G_2 = (G_1 \cap G_2 \cap G_3) \cup (G_1 \cap G_2 \cap G_4) \in \mathcal{H}$ by Claim 4.3. We must have an edge e in $(G_3 \cap G_4) \setminus ((G_1 \cap G_3 \cap G_4) \cup (G_2 \cap G_3 \cap G_4))$, otherwise $(G_1 \cap G_2) \cup (G_3 \cap G_4) \in \mathcal{H}$ would be a Radon-major. ◂

▸ **Corollary 4.6.** *If $\mathcal{G}$ is one of $\{\bumpeq, \triangle\}$ or $\{\bumpeq, \sqcap\}$, then $r(\mathcal{H}_3(\mathcal{G})) = 4$.*

▸ **Claim 4.7.** *If all $G \in \mathcal{H}_3(\mathcal{G})$ has $|G| \leq 4$, then $r(\mathcal{H}_3(\mathcal{G})) \leq 4$.*

Proof. Let $G_1, \dots, G_4 \in \mathcal{H}$ and consider first the 3-wise intersections of them. If $G_1, \dots, G_4$ have no Radon-major in $\mathcal{H}$, then by Claim 4.2 every G_i must contain at least 3 edges, one in each distinct 3-wise intersection where G_i is one of the intersecting graphs.

By Claim 4.5 at least three of the 2-wise intersections must be of size at least three. It follows from the pigeon-hole principle that at least one of the G_is have at least five edges. ◂

▸ **Corollary 4.8.** *If $\mathcal{G}$ is one of $\{\sqcap\}, \{\bumpeq\}, \{\triangle\}$ or $\{\sqcap, \triangle\}$, then $r(\mathcal{H}_3(\mathcal{G})) = 4$.*

Proof of the second part of Theorem 4.1. If $\mathcal{G}$ does not contain any graph with at least three edges, then $r(\mathcal{G}) \leq 4$ by Lemma 2.6.

If $\mathcal{G}$ does contain graphs with at least three edges, $r(\mathcal{G}) = 4$ and $\mathcal{G}$ does not have $\{\equiv\}, \{\text{⅄}, \bumpeq\}, \{\text{⅄}, \triangle, \sqcap\}$ or$\{\triangle, \bumpeq, \sqcap\}$ as subfamilies, then its 3-Helly closure equals to the 3-Helly closure of one of the remaining nonempty subfamilies of the five isomorphism classes of graphs with three edges. These subfamilies are exactly the families listed in Corollaries 4.8, 4.4 and 4.6 and the Radon number of each of their 3-Helly closures is 4. ◂

5 Open questions

We chose the following two questions with the hope that investigating them will bring us closer to improving Bukh's counterexample to Eckhoff's Conjecture. The next step towards the general setting of hypergraphs might be to consider families of 3-uniform hypergraphs with the straightforward generalization of the definition of Radon numbers for graph families.

▸ **Problem 5.1.** *Characterize families of* 3*-uniform hypergraphs with Radon number* 4*.*

One might also try to find analogs of Bukh's construction with larger Radon numbers. The first step might be to consider graph families with Radon number 5.

▸ **Problem 5.2.** *Characterize graph families* $\mathcal{G}$ *with Radon number* $r(\mathcal{G}) = 5$*.*

Acknowledgement. The author would like to thank Dömötör Pálvölgyi for a lot of interesting and encouraging discussions during the research.

References

1 Boris Bukh. Radon partitions in convexity spaces. *arXiv preprint arXiv:1009.2384*, 2010.

2 JR Calder. Some elementary properties of interval convexities. *Journal of the London Mathematical Society*, 2(3):422–428, 1971.

3 Jürgen Eckhoff. Radon's theorem revisited. In *Contributions to geometry*, pages 164–185. Springer, 1979.

4 Andreas F Holmsen and Donggyu Lee. Radon numbers and the fractional helly theorem. *Israel Journal of Mathematics*, 241(1):433–447, 2021.

5 Robert Jamison. Partition numbers for trees and ordered sets. *Pacific Journal of Mathematics*, 96(1):115–140, 1981.

6 Dömötör Pálvölgyi. Radon numbers grow linearly. *Discrete & Computational Geometry*, 68(1):165–171, 2022.

7 Johann Radon. Mengen konvexer körper, die einen gemeinsamen punkt enthalten. *Mathematische Annalen*, 83(1):113–115, 1921.

8 Helge Tverberg. A generalization of radon's theorem. *Journal of the London Mathematical Society*, 1(1):123–128, 1966.

9 Marcel LJ van De Vel. *Theory of convex structures.* Elsevier, 1993.

Proving non-realizability with grass-plucker3*

Julian Pfeifle

Universitat Politècnica de Catalunya
`julian.pfeifle@upc.edu`

Abstract

One of the great challenges of convexity is being able to decide whether or not a given simplicial complex that *could* be realizable as a convex set actually *can* be realized as the convex hull of a finite point set or not. Recently there have been two algorithmic approaches for this, namely slack realization spaces and Plücker trees. None of these are "better" that the other in every case: the latter is usually faster and can tackle more problems and bigger instances, but the former can still solve some problems that the other one can't. This extended abstract reports on the recent improvement of Plücker trees to *Plücker cubes*, which work towards closing this gap.

1 Introduction

High-dimensional objects can be counter-intuitive: some manifolds cannot be triangulated; some manifolds have the homology of a sphere but are not spheres; some spheres have vertex links that are not spheres. We don't even understand the boundary between combinatorics, topology and geometry: Given a triangulated combinatorial manifold homeomorphic to a sphere, does there exist a convex polytope whose boundary is that triangulation? We know that in the overwhelming majority of cases, the answer is "no" [1, 8], but we cannot yet effectively (never mind efficiently) decide any given concrete instance.

The two most sucessful general approaches known so far are the classical *Plücker embedding of the Grassmannian* [2] approach going back to Bokowski and Sturmfels, and the more recent *slack realization spaces* [6, 7] by Gouveia, Macchia and Wiebe. Early implementations of the Plücker embedding approach used linear programming and special combinations of Grassmann-Plücker polynomials (1) called "biquadratic final polynomials", while the most recent grass-plucker algorithm [9], released in `polymake 4.7` [5], uses a combinatorial search to combine such polynomials in a more general way. In contrast, the slack realization space approach also uses linear programming [7], and has been implemented in Macaulay 2.

While generally the grass-plucker algorithm can process larger instances and is faster, the slack realization space approach can still prove several more instances of non-realizability, largely because it allows to successively specialize certain orientations in an ad-hoc decision tree. The grass-plucker3 algorithm presented in this paper systematically incorporates this feature into the Plücker embedding approach, under the name of *Plücker cubes.*

2 An example

Let's input the *prismatoid #375* [3] into our algorithm grass-plucker3 and describe its output.

$\Sigma = \Pi_{375}$ is a simplicial sphere of dimension 4 with 15 vertices, which we label 0,...,9, a,b,c,d,e for convenience, and the following list of 101 facets:

01234, 0124b, 0128b, 0128d, 0129d, 013ab, 0134b, 0136a, 0136c, 0138c, 0138d, 0139d, 016ab, 016bc, 018bc, 023ab, 0234b, 0235a, 0259a, 026ab, 0268b, 02689, 0269a, 0289d, 0359a, 0369a,

* Supported by the grant PID2019-106188GB-I00 from the Spanish Ministry of Education (MEC).

0369c, 038ce, 0389d, 0389e, 039ce, 06bce, 068be, 0689e, 069ce, 08bce, 1234d, 1239d, 124bd, 128bd, 134bd, 136ce, 1368a, 1368e, 138ab, 138bd, 138ce, 16abc, 168ac, 168ce, 18abc, 2345b, 23456, 23467, 23478, 2348d, 235ab, 2356c, 2357c, 2367c, 2378e, 2379e, 2389d, 2389e, 245bd, 24568, 2458d, 24678, 256ab, 2568b, 2569a, 2569c, 2579c, 258bd, 267ce, 2678e, 2689e, 269ce, 279ce, 3456b, 3467b, 347bd, 3478d, 356ab, 3569a, 3569c, 3579c, 367ab, 367ce, 3678a, 3678e, 378ab, 378bd, 379ce, 4568b, 458bd, 467bd, 4678d, 67abd, 678ad, 78abd.

This list of facets is the entire input to the problem! The task is to determine the realizability of Σ as a convex polytope without recourse to any other information.

In this case, our algorithm proves that Σ is in fact *not* realizable as a convex polytope, and it does so by exhibiting the "Plücker cube" depicted in Figure 1:

$$\begin{tikzcd}
+[A][M]+[D]^?[V]+[K][E]=0 \arrow[rrr, dotted, dash] \arrow[d, dash] & & & +[A][M]+[D]^?[V]+[K][E]=0 \arrow[d, dash] \\
-[K][B]^?-[L][D]^?+[M][F]^?=0 \arrow[dr, dash] & & & +[A][B]^?-[C][D]^?+[E][F]^?=0 \arrow[dl, dash] \\
& {[B]^?=-,\ [F]^?=+} \arrow[r, dash] \arrow[dd, dash] \arrow[ddr, phantom, "\{[B]^?,[F]^?\}"] & {[B]^?=+,\ [F]^?=+} \arrow[dd, dash] & \\
& & & \\
& {[B]^?=-,\ [F]^?=-} \arrow[r, dash] \arrow[dl, dash] & {[B]^?=+,\ [F]^?=-} \arrow[dr, dash] & \\
-[N][F]^?+[Q][C]-[B]^?[P]=0 & & & -[G][H]^?+[I][B]^?-[J][F]^?=0 \arrow[d, dash] \\
& & & +[R][H]^?+[O][S]+[T][U]=0
\end{tikzcd}$$

Figure 1 A Plücker cube (in this case, a square) proving the non-realizability of Π_{375}. Notice the difference between the *polynomials* with 3 terms that must equal 0 in any realization (explained in item 2 below) and the *sign choices* for the blue monomials in the inner square (see item 3).

This diagram is read as follows: Since the sphere $\Sigma = \Pi_{375}$ has dimension $d = 4$, each 4-simplex facet has $d+1 = 5$ vertices, and each solid $d+2 = 6$ vertices, see Table 1 below.

(In the case $d = 2$, each triangular facet of a simplicial 2-sphere has $d+1 = 3$ vertices, while the solid tetrahedra that "live in the space $\mathbb{R}^{d+1}$" have $d+2 = 4$ vertices.)

$A = 234675$	$B = 23578b$	$C = 234785$	$D = 23567b$	$E = 234b57$	$F = 235678$
$G = 237e85$	$H = 23678b$	$I = 367e82$	$J = 237e8b$	$K = 236c75$	$L = 2357c8$
$M = 2357cb$	$N = 234b58$	$O = 3678a2$	$P = 245683$	$Q = 2568b3$	$R = 23467a$
$S = 346b72$	$T = 367ab2$	$U = 246783$	$V = 2357c4$		

Table 1 The solids involved in Figure 1, where we have labeled the 15 vertices by 0,...,9,a,...,e.

In Figure 1, we don't actually consider the solids from Table 1 themselves, but we moreover *orient them positively* using the orientation of the sphere Σ whenever we can (see item 1 below). This orientation is denoted by square brackets $[X]$ around the letter X denoting the solid. Whenever the orientation could be either way we indicate this by appending a small question mark to the square brackets, like $[X]^?$; this is explained in items 3 and 4 below.

We now discuss each feature of Figure 1 in turn.

1. *The "black" solids:* All solids in Figure 1 except for B, F and D, H are *oriented positively*: in any convex realization of Σ in $\mathbb{R}^{d+1}$, the determinant of the $(d+2)\times(d+2)$ matrix whose columns contain the homogeneous coordinates of the vertices in each black solid, listed in the given order, is positive. This can be done because each black solid $S = F \cup \{v_0\}$ contains the vertex set of a facet F of Σ, and using a basis for the top-dimensional homology of Σ (which has rank 1 and only depends on the combinatorics of the sphere, not on any realization), we can permute the vertices in F, i.e., reorient F, to achieve that the extra vertex v_0 lies on the "positive" side of the hyperplane spanned by F.
   For example, the solid $A = 234675$ contains the facets $f = 23456$ and $f' = 23467$, which both turn out to be oriented positively in the homology basis. Appending the extra vertex $v_0 = 7$ to f and adjacent-swapping 5 twice towards the end yields $[A]$ and preserves the positive orientation, as does appending the extra vertex $v_0' = 5$ to f'.
   On the other hand, the solid $K = 236\mathrm{c}75$ contains the positive facet $f = 2357\mathrm{c}$ and the negative facet $f' = 2367\mathrm{c}$. Appending $v_0 = 6$ to f one reaches $[K]$ with an even number of adjacent swaps, preserving the orientation, while after appending $v_0' = 5$ to f' one needs one adjacent swap to make the solid positively oriented.
2. *The polynomials:* Each polynomial in Figure 1 is a *Grassmann–Plücker polynomial* (in this paper, a *GP polynomial*)

$$\Gamma(I,J) \;:=\; \sum_{k=1}^{d+2} (-1)^k \,[i_1,\dots,i_d,j_k]\,[j_1,\dots,\widehat{j_k},\dots,j_{d+2}] \qquad (1)$$

   for any two index sets $I \in \binom{[n]}{d}$ and $J \in \binom{[n]}{d+2}$, where as usual the hat notation $\widehat{j_k}$ means omission of the k-th index from J. Each monomial has $d+1$ entries and is therefore a solid of Σ.
   It turns out that *every GP polynomial is zero in any convex realization of* Σ, and these relations are called *Grassmann-Plücker relations*, or in this paper, *GP relations*:

$$\Gamma(I,J) \;=\; 0 \qquad \text{for all } I \in \binom{[n]}{d},\; J \in \binom{[n]}{d+2}.$$

   The reason why the signs in the GP polynomials of Figure 1 do not alternate as nicely as (1) suggests is because we have permuted some solids to make them positively oriented.
3. *The sign choices for the "blue" solids:* In contrast, the solids $[B]^?$ and $[F]^?$ do *not* contain a (boundary) facet of Σ, so their orientations can take either sign depending on the concrete realization of Σ, cf. Figure 2. The $2^2 = 4$ *vertices* $[B]^? = \pm$, $[F]^? = \pm$ of the 2-dimensional Plücker cube account for all combinations of signs of these *undetermined* solids.
4. *The "red" solids:* The orientations of the solids $[D]^?$ and $[H]^?$ are also not determined by the combinatorics of Σ, and again for every sign with which these solids appear in a GP polynomial, there is a GP polynomial in which it appears with the opposite sign.

Some more comments on the notation:

- The presentation of the black solids in Table 1 has been chosen so that the first five indices describe a positively oriented facet, and the last index a vertex not on it.
- By checking against the list of facets, one may check that no red solid and no blue solid contains any facet, compare Figure 2 below. To make this visually explicit, we append a small question mark to these solids: $[B]^?$, $[F]^?$, $[B]^?$, $[H]^?$.
- The only reason for distinguishing blue and red solids is to keep track of which signs (the blue ones) are being exhaustively enumerated, i.e. all sign combinations appear in the Plücker cube, and which ones (the red ones) only appear opportunistically.

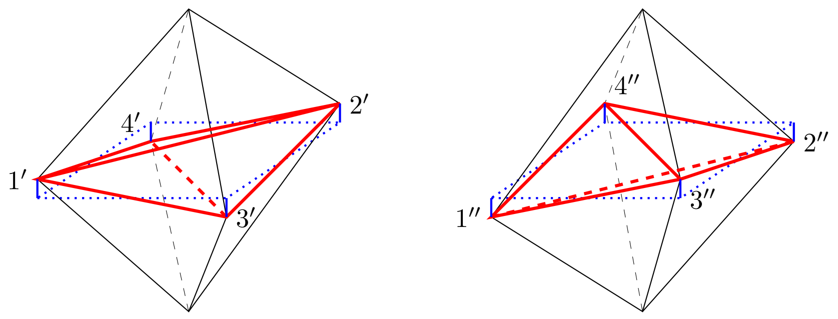


Figure 2 If a solid tetrahedron determined by the vertices of the sphere (in this case, an octahedron) does not contain a boundary facet, its orientation can be positive or negative.

- The positive orientation $[X]$ of a solid X is actually one of the two equivalence classes of permutations of the vertices of X with the same parity. Which class we are really talking about in any particular case is determined by the orientation of the boundary of Σ given by the top homology class. We represent undetermined solids by the lexicographical ordering of its vertices.
- The fact that all terms connected to the central square in Figure 1 and containing red solids are negative is purely coincidental.

2.1 What is the actual point of the whole diagram?

Consider the undetermined solids $[B]^? = [23578b]^?$ and $[F]^? = [235678]^?$. For each of the $2^2 = 4$ possible sign patterns that can occur, the monomials containing these solids in the GP polynomial at the corresponding vertex of the cube are positive.

For example, if $[B]^? = +$ and $[F]^? = -$, then the monomials containing these solids in the following GP relation are positive:

$$-[G][H]^? + [I][B]^? - [J][F]^? \;=\; 0. \tag{2}$$

- If, in this realization of Σ with $[B]^? = +$ and $[F]^? = -$ that we are considering, it also happens that $[H]^? = -$, then *all terms* in the GP polynomial in (2) are *positive* — but that cannot be, because any GP polynomial *vanishes* for any realization of Σ.
- On the other hand, if $[H]^? = +$, then the GP polynomial

  $$+[R][H]^? + [O][S] + [T][U]$$

  connected to it in the lower right of Figure 1 is positive, which is also impossible.

In this way, Figure 1 provides a GP polynomial contradicting realizability for any combination of signs of the participating undetermined solids. Notice that the GP polynomial $+[A][M] + [D]^?[V] + [K][E]$ is used in two distinct vertices.

3 The grass-plucker3 algorithm

On a high level, we execute the following steps:

Enumeration of GP polynomials: The $\Gamma(I,J)$'s are at the heart of all subsequent data structures, and generating and storing them efficiently is vital. Even though the user can limit their number, for example by specifying the maximal number of terms or the maximal number of undetermined solids, control of memory consumption is still a critical issue. For this reason, the present implementation limits the maximum number of vertices of Σ to 32, and stores each $\Gamma(I,J)$ in implicit form as a single 64-bit integer whose high and low 32 bits contain a bitset representation of I and J, respectively.[1]

Identification of potential leaves: Some spheres have a positive GP polynomial with *no* undetermined solids, which directly proves their non-realizability.
The next step up are GP polynomials with exactly one undetermined solid, and they are special and precious: They make up leaves of the resulting decision tree.
We maintain an ordered list for leaves and another ordered list for non-leaf decision trees, and refer to both leaves and trees via their indices (offsets) in those lists.
Next, we represent each undetermined solid S as a bitset in a signed integer dubbed *sush* (for *"signed undetermined solid hash"*), and maintain two `map` data structures that operate on such a sush S: `leaf_of_sush` takes S to the index of *one* leaf, i.e. a GP polynomial in which S occurs as the only undetermined solid with the specified sign, while `nonleaf_trees_of_sush` takes S to *all* indices of the trees in which S is unpaired.
Whenever we encounter a new GP polynomial Γ with exactly one undetermined solid S, we check whether `leaf_of_sush[-S]` already exists (in which case we have found a certificate consisting of two GP polynomials), and whether `nonleaf_trees_of_sush[-S]` exists (in which case we "close off" the unpaired solid $-S$ in those trees, and hopefully come closer to having a tree with no unpaired solids).

Enumeration of cubes, and conversion to trees: If no certificate was found yet, we enumerate all tuples $C = \{S_1, S_2, \dots, S_k\}$ of "blue" sushes such that for each assignment of signs to the S_i, there exists a GP polynomial in which the S_i occur with those signs — in Section 2, we would find $C = \{[B]^?, [F]^?\}$, among others. In general, there are many such GP polynomials for each "cube vertex", i.e., each selection of signs, and we choose from among them in a compatible way according to some criteria: For instance, we will favor GP polynomials with fewer "red" undetermined solids that are not in C, and if a given "red" undetermined solid (such as $[D]^?$ in Section 2) occurs in more than one vertex, we make sure it always occurs with the same sign. Each choice yields a tree with "special" nodes corresponding to the cube vertices, and "usual" nodes corresponding to the chosen GP polynomials. Of course, we immediately check if each obtained tree can be completed using `leaf_of_sush`, and this yields the result in Section 2.

Joining trees: If still no certificate was found, we have accumulated a collection $\mathcal{T}$ of trees, and a collection $\mathcal{U}$ of unmatched signed undetermined solids in those trees. We next pair the trees by iterating over $\mathcal{U}$ in a convenient order, for example by prioritizing the sushes that yield new leaves when joining trees among them. For the sake of conserving memory, it is absolutely crucial that these trees be represented in an implicit way by only storing the indices of trees that are joined. We therefore work with an ordered list of "explicit trees", which store all trees found in the previous steps, and an ordered list of "implicit trees", which are small data structures storing little more than the indices of the parent trees and the sush along which the join was made, and the currently unpaired sushes.

[1] Actually, we store $\pm\Gamma(I,J)$ as a *signed* 64 bit integer because we need to consider both versions of the GP polynomial, and therefore only allow up to 31 vertices; so one bit goes unused. A way around this limitation is to use general bitsets to store the index sets.

End: If after iterating over all of $\mathcal{U}$ no certificate has been found, the algorithm gives up.

4 Discussion and Outlook

Some portions of the algorithm are still under active development, especially the efficient generation of GP polynomials in the first step, and the order in which the sushes in $\mathcal{U}$ are selected in the last step. Nevertheless, the current version already resides in the source tree for the upcoming version `4.8` of `polymake`, which means it will become available once that release is published.

The largest benchmark instance that is as yet just out of reach, and which the algorithm will hopefully manage to process after the current phase of development, is an 11-dimensional sphere with 24 vertices and 5778 facets whose realizability status is unknown at present.

Another interesting benchmark example is the sphere with the evocative name `f374225`, which Firsching [4] conjectured to be non-realizable. However, neither Gouveia, Macchia & Wiebe's algorithm nor our grass-plucker3 can as yet certify this.

A further family of testcases to be examined is the entire collection of Criado & Santos's topological prismatoids [3], for which Gouveia, Macchia & Wiebe can currently certify more instances of non-realizability.

5 Acknowledgements

The author is indebted to the three anonymous referees for their in-depth reading of the preliminary version and their many detailed and pertinent comments. Thank you!

References

1 Karim Adiprasito and Arnau Padrol. The universality theorem for neighborly polytopes. *Combinatorica*, 37:129–136, 2017.

2 Jürgen Bokowski and Bernd Sturmfels. *Computational synthetic geometry*, volume 1355. Berlin etc.: Springer-Verlag, 1989.

3 Francisco Criado and Francisco Santos. Topological Prismatoids and Small Simplicial Spheres of Large Diameter. *Experimental Mathematics*, 2019. `https:doi.org/10.1080/10586458.2019.1641766`.

4 Moritz Firsching. Realizability and inscribability for simplicial polytopes via nonlinear optimization. *Math. Program.*, 166(1-2 (A)):273–295, 2017.

5 Ewgenij Gawrilow and Michael Joswig. `polymake`: a framework for analyzing convex polytopes. In *Polytopes—combinatorics and computation (Oberwolfach, 1997)*, volume 29 of *DMV Sem.*, pages 43–73. Birkhäuser, Basel, 2000.

6 João Gouveia, Antonio Macchia, Rekha R. Thomas, and Amy Wiebe. The slack realization space of a polytope. *SIAM J. Discrete Math.*, 33(3):1637–1653, 2019.

7 Antonio Macchia and Amy Wiebe. Slack ideals in macaulay2, 2020. `arXiv:2003.07382`.

8 Arnau Padrol. Many neighborly polytopes and oriented matroids. *Discrete Comput. Geom.*, 50(4):865–902, 2013. `doi:10.1007/s00454-013-9544-7`.

9 Julian Pfeifle. Positive Plücker tree certificates for non-realizability. *Experimental Math.*, pages 1–17, 2022. `https://doi.org/10.1080/10586458.2021.1994487`.

On the geometric thickness of 2-degenerate graphs

Rahul Jain[1], Marco Ricci[1], Jonathan Rollin[1], and André Schulz[1]

1 **FernUniversität in Hagen, Germany**
{rahul.jain,marco.ricci,jonathan.rollin,andre.schulz}@fernuni-hagen.de

Abstract

A graph is 2-degenerate if every subgraph contains a vertex of degree at most 2. We show that every 2-degenerate graph can be drawn with straight lines such that the drawing decomposes into 4 plane forests. Therefore, the geometric arboricity, and hence the geometric thickness, of 2-degenerate graphs is at most 4. On the other hand, we show that there are 2-degenerate graphs that do not admit any straight-line drawing with a decomposition of the edge set into 2 plane graphs. That is, there are 2-degenerate graphs with geometric thickness, and hence geometric arboricity, at least 3. This answers two questions posed by Eppstein [Separating thickness from geometric thickness. In *Towards a Theory of Geometric Graphs*, vol. 342 of *Contemp. Math.*, AMS, 2004].

Related Version *Full Version*: `https://arxiv.org/abs/2302.14721` [14]

1 Introduction

A graph is planar if it can be drawn without crossings on a plane. Planar graphs exhibit many nice properties, which can be exploited to solve problems for this class more efficiently compared to general graphs. However, in many situations, graphs cannot be assumed to be planar even if they are sparse. It is therefore desirable to define graph classes which extend planar graphs. Several approaches for extending planar graphs have been established over the last years [3, 12]. Often these classes are defined via drawings, for which the types of crossings and/or the number of crossings are restricted. A natural way to describe how close a graph is to being a planar graph is provided by the graph parameter *thickness*. The thickness of a graph G is the smallest number $\theta(G)$ such that the edges of G can be partitioned into $\theta(G)$ planar subgraphs of G. Related graph parameters are *geometric thickness* and *book thickness*. Geometric thickness was introduced by Kainen under the name *real linear thickness* [15]. The geometric thickness $\bar{\theta}(G)$ of a graph G is the smallest number of colors that is needed to find an edge-colored geometric drawing (i.e., one with edges drawn as straight-line segments) of G with no monochromatic crossings. For the book thickness $\mathsf{bt}(G)$, we only consider geometric drawings with vertices in convex position.

An immediate consequence from the definitions of thickness, geometric thickness and book thickness is that for every graph G we have $\theta(G) \leq \bar{\theta}(G) \leq \mathsf{bt}(G)$. Eppstein shows that the three thickness parameters can be arbitrarily "separated". Specifically, for any number k there exists a graph with geometric thickness 2 and book thickness at least k [9] as well as a graph with thickness 3 and geometric thickness at least k [10]. The latter result is particularly notable since any graph of thickness k admits a k-edge-colored drawing of G with no monochromatic crossings if edges are not required to be straight lines. This follows from a result by Pach and Wenger [20], stating that any planar graph can be drawn without crossings on arbitrary vertex positions with polylines.

Related to the geometric thickness is the *geometric arboricity* $\bar{\mathsf{a}}(G)$ of a graph G, introduced by Dujmović and Wood [5]. It denotes the smallest number of colors among all edge-colored geometric drawings of G without monochromatic crossings where every color class is acyclic. As every such plane forest is a plane graph, we have $\bar{\theta}(G) \leq \bar{\mathsf{a}}(G)$. Moreover, every plane graph can be decomposed into three forests [22], and therefore $3\bar{\theta}(G) \geq \bar{\mathsf{a}}(G)$.

Bounds on the geometric thickness are known for several graph classes. Due to Dillencourt et al. [4] we have $\frac{n}{5.646} + 0.342 \leq \bar{\theta}(K_n) \leq \frac{n}{4}$ for the complete graph K_n. Graphs with bounded degree can have arbitrarily high geometric thickness. In particular, as shown by Barárt et al. [1], there are d-regular graphs with n vertices and geometric thickness at least $c\sqrt{d}n^{1/2-4/d-\varepsilon}$ for every $\varepsilon > 0$ and some constant c. However, due to Duncan et al. [7], if the maximum degree of a graph is 4, its geometric thickness is at most 2. For graphs with treewidth t, Dujmović and Wood [5] showed that the maximum geometric thickness is $\lceil t/2 \rceil$. Hutchinson et al. [13] showed that graphs with n vertices and geometric thickness 2 can have at most $6n - 18$ edges. As shown by Durocher et al. [8], there are n-vertex graphs for any $n \geq 9$ with geometric thickness 2 and $2n - 19$ edges. In the same paper, it is proven that it is NP-hard to determine if the geometric thickness of a given graph is at most 2. Computing thickness [16] and book thickness [2] are also known to be NP-hard problems. For bounds on the thickness for several graph classes, we refer to the survey of Mutzel et al. [17]. An overview on bounds for book thickness is given on the webpage of Pupyrev [21].

A graph G is *d-degenerate* if every subgraph contains a vertex of degree at most d. So we can repeatedly find a vertex of degree at most d and remove it, until no vertices remain. The reversal of this vertex order (known as a *degeneracy order*) yields a construction sequence for G that adds vertex by vertex and each new vertex is connected to at most d previously added vertices (called its *predecessors*). Adding a vertex with exactly two predecessors is also known as a Henneberg 1 step [11]. In particular, any 2-degenerate graph is a subgraph of a so-called Laman graph, however not every Laman graph is 2-degenerate. Laman graphs are the generically minimal rigid graphs and they are exactly those graphs constructable from a single edge by some sequence of Henneberg 1 and Henneberg 2 steps (the latter step consists of subdividing an arbitrary existing edge and adding a new edge between the subdivision vertex and an arbitrary, yet non-adjacent vertex). All d-degenerate graphs are (d, ℓ)-sparse, for any $\binom{d+1}{2} \geq \ell \geq 0$, that is, every subgraph on n vertices has at most $dn - \ell$ edges.

Our Results. In this paper, we study the geometric thickness of 2-degenerate graphs. Due to the Nash-Williams theorem [18, 19], every 2-degenerate graph can be decomposed into 2 forests and hence has arboricity at most 2 and therefore thickness at most 2. On the other hand, as observed by Eppstein [9], 2-degenerate graphs can have unbounded book thickness. Eppstein's examples of graphs with thickness 3 and arbitrarily high geometric thickness are 3-degenerate graphs [10]. Eppstein asks whether the geometric thickness of 2-degenerate graphs is bounded by a constant from above and whether there are 2-degenerate graphs with geometric thickness greater than 2. The currently best upper bound of $O(\log n)$ follows from a result by Duncan for graphs with arboricity 2 [6]. We improve this bound and answer both of Eppstein's questions with the following two theorems.

▶ **Theorem 1.** *For each 2-degenerate graph G we have $\bar{\theta}(G) \leq \bar{\mathsf{a}}(G) \leq 4$.*

▶ **Theorem 2.** *There is a 2-degenerate graph G with $\bar{\mathsf{a}}(G) \geq \bar{\theta}(G) \geq 3$.*

We give proof ideas for these theorems in Sections 2 and 3, respectively.

2 The upper bound

In this section, we outline the proof of Theorem 1. We describe, for any 2-degenerate graph, a construction for a straight-line drawing such that the edges can be colored using four colors,

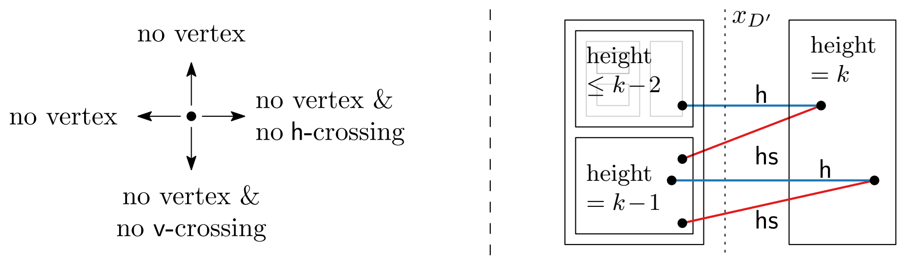


Figure 1 Left: For each vertex v in a feasible drawing, there are no other vertices on the vertical and the horizontal line through v. Moreover, v is h-open to the right and v-open to the bottom. Right: All vertices in the highest level (of height k) are placed to the right of all vertices of smaller height. Each vertex in that level is incident to one edge of color h and one edge of color hs.

avoiding monochromatic crossings and monochromatic cycles. This shows that 2-degenerate graphs have geometric arboricity, and hence geometric thickness, at most four.

For a graph G we denote its edge set with $E(G)$ and its vertex set with $V(G)$. Consider a 2-degenerate graph G with a given, fixed degeneracy order. We define the *height* of a vertex v in G as the length t of a longest path $u_0 \cdots u_t$ with $u_t = v$ such that for each i, with $1 \leq i \leq t$, the vertex u_{i-1} is a predecessor of u_i. The height of G is the largest height among its vertices. The set of vertices of the same height is called a *level* of G.

Our construction process embeds G level by level with increasing height. The levels are placed alternately either strictly below or strictly to the right of the already embedded part of the graph. If a level is placed below, then we use specific colors v and vs (short for "vertical" and "vertical slanted", respectively) for all edges between this level and levels of smaller height. Similarly, we use specific colors h and hs (short for "horizontal" and "horizontal slanted", respectively) if a level is placed to the right. See Figure 1 (right).

To make our construction work, we need several additional constraints to be satisfied in each step which we will describe next. For a point p in the plane, we use the notation $\mathsf{x}(p)$ and $\mathsf{y}(p)$ to refer to the x- and y-coordinates of p, respectively. Consider a drawing D of a 2-degenerate graph G together with a coloring of the edges with colors $\{\mathsf{h}, \mathsf{hs}, \mathsf{v}, \mathsf{vs}\}$. For the remaining proof, we assume that each vertex of G has either 0 or exactly 2 predecessors. If not, we add a dummy vertex without predecessors to the graph and make it the second predecessor of all those vertices which originally only had 1 predecessor. Let k denote the height of G. We say that D is *feasible* if it satisfies the following constraints:

(C1) For each vertex in G the edges to its predecessors are colored differently. If $k > 0$, then each vertex of height k in G is incident to edges of colors h and hs only.

(C2) There exists some $x_D \in \mathbb{R}$ such that for each vertex $v \in V(G)$ we have $\mathsf{x}(v) > x_D$ if and only if v is of height k in G.

(C3) There is no monochromatic crossing.

(C4) No two vertices of G lie on the same horizontal or vertical line.

(C5) Each $v \in V(G)$ is h-*open to the right*, that is, the horizontal ray emanating at v directed to the right avoids all h-edges.

(C6) Each $v \in V(G)$ is v-*open to the bottom*, that is, the vertical ray emanating at v directed downwards avoids all v-edges.

These constraints are schematized in Figure 1. We now show how to construct a feasible drawing for G. We prove this using induction on the height of the graph. The base case $k = 0$ is trivial, as there are no edges in the graph. Assume that $k \geq 1$ and the theorem is true for all 2-degenerate graphs with height $k - 1$. Let H denote the subgraph of G induced by vertices with height less than k. By induction, there is a feasible drawing D of H.

As a first step, we reflect the drawing D at the straight line $y = -x$. Additionally, we swap the colors hs and vs as well as the colors h and v. Let D' denote the resulting drawing. From now on, all appearing coordinates of vertices refer to coordinates in D'. By construction, D' satisfies (C3–C6). Applying (C1) to D shows that in D' each vertex of height $k-1$ is incident to one edge of color v and one edge of color vs. Applying (C2) to D shows that there exists $y_{D'} \in \mathbb{R}$ such that for each vertex $v \in V(H)$ we have $\mathsf{y}(v) < y_{D'}$ in D' if and only if v is of height $k-1$.

As the second (and last) step, we place the points of height k of G such that the resulting drawing is feasible. We only give a rough description of this placement here and refer to the full version of this paper [14, Section 2] for a precise formulation. Let L_k denote the set of these vertices and let $x_{D'}$ denote the largest x-coordinate among all vertices in D'. We choose a sufficiently small, positive slope m such that for any distinct u, $v \in V(H)$ with $\mathsf{y}(u) < \mathsf{y}(v)$, the horizontal line through v and the straight line through u with slope m intersect at a point p with $\mathsf{x}(p) > x_{D'}$. For each vertex $w \in L_k$ let u and v be the two predecessors of w in H with $\mathsf{y}(u) < \mathsf{y}(v)$ and let p^w denote the intersection point of the straight line of slope m passing through u (called a slanted line) and the horizontal line passing through v. We place w at point p^w and connect w to v using an edge of color h and we connect w to u using an edge of color hs. Then (C1), (C2) and (C6) are clearly satisfied. However, this placement comes with some issues: Several vertices in L_k might have the same predecessors and, hence, are placed on the same point, new edges of the same color with a common endpoint in H overlap (along a horizontal or slanted line), and (C3–C5) might not be satisfied, yet. To address these issues, we use a small perturbation, moving each point $w \in L_k$ slightly to the bottom-right (along a straight line of slope $-1/m$ through p^w) such that all vertices $w \in L_k$ are placed at different distances to their respective point p^w. In the full version of this paper [14, Section 2] we describe such a perturbation which yields a feasible drawing of G. This eventually shows that the geometric arboricity, and hence the geometric thickness, of G is at most four.

3 The lower bound

In this section, we shall describe a 2-degenerate graph with geometric thickness at least 3. For a positive integer n let $G(n)$ denote the graph constructed as follows. Start with a vertex set Λ_0 of size n and for each pair of vertices from Λ_0 add one new vertex adjacent to both vertices from the pair. Let Λ_1 denote the set of vertices added in the last step. For each pair of vertices from Λ_1 add 89 new vertices, each adjacent to both vertices from the pair. Let Λ_2 denote the set of vertices added in the last step. For each pair of vertices from Λ_2 add one new vertex adjacent to both vertices from the pair. Let Λ_3 denote the set of vertices added in the last step. This concludes the construction. Observe that for each $i = 1, 2, 3$, each vertex in Λ_i has exactly two neighbors in Λ_{i-1}. Hence, $G(n)$ is 2-degenerate. We claim that for sufficiently large n the graph $G(n)$ has geometric thickness at least 3. Due to limited space we briefly sketch of our arguments here. A complete proof is provided in the full version of this paper [14, Section 3].

Consider a geometric drawing of $G(n)$, for large n, and assume that there is a partition of its edge set into two plane subgraphs $\mathbb{A}$ and $\mathbb{B}$. In the first step, we find a large, and particularly nice grid structure (called a tidy grid) formed by edges between Λ_0 and Λ_1 where many disjoint $\mathbb{A}$-edges cross many disjoint $\mathbb{B}$-edges. We additionally ensure that there is a large subset $\Lambda_1' \subseteq \Lambda_1$ spread out over many cells of this grid. Next, we consider

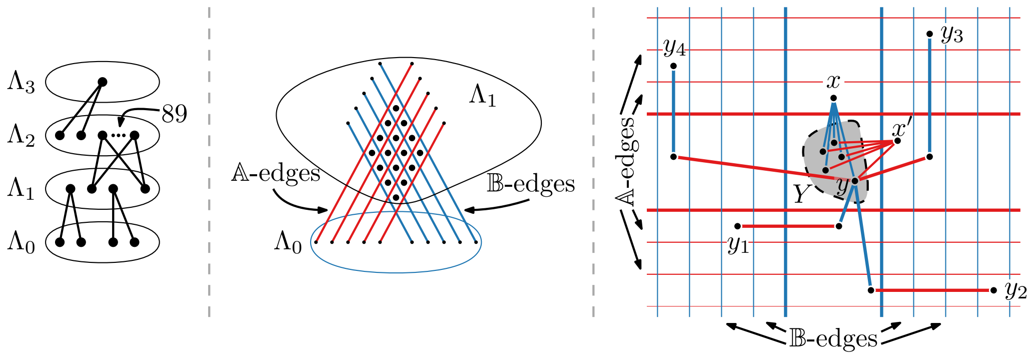


Figure 2 Left: Sketch of the graph $G(n)$. Middle: A tidy grid. Right: The situation leading to a contradiction in the proof of Theorem 2 with x, $x' \in \Lambda_1$, $Y \subseteq \Lambda_2$, and y_1, y_2, y_3, $y_4 \in \Lambda_3$.

the connections of vertices from Λ_1' via the edges towards Λ_2. We show that the drawing restrictions imposed by the surrounding grid edges force many of the edges between Λ_1' and Λ_2 to stay within the grid. In particular, this gives a large subset $\Lambda_2' \subseteq \Lambda_2$ spread out over many cells of the grid. Similarly to the previous argument, we then find many of the edges between Λ_2' and Λ_3 staying within the grid. We eventually arrive at a situation depicted in Figure 2 (right): A cell with a set Y of five vertices from Λ_2 with the same predecessors in Λ_1, such that for each $y \in Y$ there are four vertices $y_1, \dots, y_4 \in \Lambda_2$ (one from the bottom-left, one from the bottom-right, one from the top-right, and one from the top-left part of the grid) and for each i the common neighbor of y and y_i from Λ_3 lies in the grid. It turns out, that each $y \in Y$ either has an $\mathbb{A}$-edge to the left and an $\mathbb{A}$-edge to the right or it has a $\mathbb{B}$-edge to the top and a $\mathbb{B}$-edge to the bottom (using directions from Figure 2). As this is impossible to realize for all five vertices in Y simultaneously, the geometric thickness of $G(n)$ is at least 3.

References

1 János Barát, Jiří Matoušek, and David R. Wood. Bounded-degree graphs have arbitrarily large geometric thickness. *The Electronic Journal of Combinatorics*, 13:R3, 2006. `doi:10.37236/1029`.

2 Fan R. K. Chung, Frank T. Leighton, and Arnold L. Rosenberg. Embedding graphs in books: A layout problem with applications to VLSI design. *SIAM Journal on Algebraic Discrete Methods*, 8(1):33–58, 1987. `doi:10.1137/0608002`.

3 Walter Didimo, Giuseppe Liotta, and Fabrizio Montecchiani. A survey on graph drawing beyond planarity. *ACM Computing Surveys*, 52:1–37, January 2020. `arXiv:1804.07257`, `doi:10.1145/3301281`.

4 Michael B. Dillencourt, David Eppstein, and Daniel S. Hirschberg. Geometric thickness of complete graphs. *Journal of Graph Algorithms & Applications*, 4(3):5–17, 2000. `arXiv:math/9910185`, `doi:10.1007/s00454-007-1318-7`.

5 Vida Dujmović and David R. Wood. Graph treewidth and geometric thickness parameters. *Discrete & Computational Geometry*, 37:641–670, 2007. `arXiv:math/0503553`, `doi:10.1007/s00454-007-1318-7`.

6 Christian A. Duncan. On graph thickness, geometric thickness, and separator theorems. *Computational Geometry*, 44:95–99, February 2011. `doi:10.1016/j.comgeo.2010.09.005`.

7 Christian A. Duncan, David Eppstein, and Stephen G. Kobourov. The geometric thickness of low degree graphs. In *SCG '04: Proceedings of the Twentieth Annual Symposium on*

Computational Geometry, pages 340–346, New York, NY, USA, June 2004. Association for Computing Machinery. arXiv:math/0312056, doi:10.1145/997817.997868.

8 Stephane Durocher, Ellen Gethner, and Debajyoti Mondal. Thickness and colorability of geometric graphs. *Computational Geometry*, 56:1–18, 2016. doi:10.1016/j.comgeo.2016.03.003.

9 David Eppstein. Separating geometric thickness from book thickness. Preprint, 2001. arXiv:math/0109195.

10 David Eppstein. Separating thickness from geometric thickness. In János Pach, editor, *Towards a Theory of Geometric Graphs*, volume 342 of *Contemporary Mathematics*. American Mathematical Society, 2004. arXiv:math/0204252.

11 Lebrecht Henneberg. Die Graphische Statik der Starren Körper. In Felix Klein and Conrad Müller, editors, *Encyklopädie der Mathematischen Wissenschaften mit Einschluss ihrer Anwendungen: Vierter Band: Mechanik*, pages 345–434, Wiesbaden, Germany, 1908. B. G. Teubner Verlag. doi:10.1007/978-3-663-16021-2_5.

12 Seok-Hee Hong and Takeshi Tokuyama, editors. *Beyond Planar Graphs. Communications of NII Shonan Meetings*. Springer, 2020. doi:10.1007/978-981-15-6533-5.

13 Joan P. Hutchinson, Thomas C. Shermer, and Andrew Vince. On representations of some thickness-two graphs. *Computational Geometry*, 13(3):161–171, 1999. doi:10.1016/S0925-7721(99)00018-8.

14 Rahul Jain, Marco Ricci, Jonathan Rollin, and André Schulz. On the geometric thickness of 2-degenerate graphs. Preprint, 2023. arXiv:2302.14721.

15 Paul C. Kainen. Thickness and coarseness of graphs. *Abhandlungen aus dem Mathematischen Seminar der Universität Hamburg*, 39:88–95, 1973. doi:10.1007/BF02992822.

16 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.

17 Petra Mutzel, Thomas Odenthal, and Mark Scharbodt. The thickness of graphs: A survey. *Graphs and Combinatorics*, 14:59–73, 1998. doi:10.1007/PL00007219.

18 Crispin St.J. A. Nash-Williams. Edge-disjoint spanning trees of finite graphs. *Journal of the London Mathematical Society*, 36:445–450, 1961. doi:10.1112/jlms/s1-36.1.445.

19 Crispin St.J. A. Nash-Williams. Decomposition of finite graphs into forests. *Journal of the London Mathematical Society*, 39:12, 1964. doi:10.1112/jlms/s1-39.1.12.

20 János Pach and Rephael Wenger. Embedding planar graphs at fixed vertex locations. *Graphs and Combinatorics*, 17(4):717–728, 2001. doi:10.1007/PL00007258.

21 Sergey Pupyrev. Linear Graph Layouts. Accessed: 2022-11-30. URL: https://spupyrev.github.io/linearlayouts.html.

22 Walter Schnyder. Embedding planar graphs on the grid. In *SODA '90: Proceedings of the first annual ACM-SIAM Symposium on Discrete Algorithms*, pages 138–148. Society for Industrial and Applied Mathematics, January 1990. doi:10.5555/320176.320191.

Realizations of multiassociahedra via rigidity[*]

Luis Crespo Ruiz and Francisco Santos

Departamento de Matemáticas, Estadística y Computación, Universidad de Cantabria, 39005 Santander, Spain
luis.cresporuiz@unican.es, francisco.santos@unican.es

Abstract

Let $\Delta_k(n)$ denote the simplicial complex of $(k+1)$-crossing-free subsets of edges in $\binom{[n]}{2}$. Here $k, n \in \mathbb{N}$ and $n \geq 2k+1$. Jonsson (2005) proved that (neglecting the short edges that cannot be part of any $(k+1)$-crossing), $\Delta_k(n)$ is a shellable sphere of dimension $k(n-2k-1)-1$, and conjectured it to be polytopal. It was later noticed by Stump (2011) that the same result and question follows from the work of Knutson and Miller (2004) on subword complexes.

Despite considerable effort, the only values of (k, n) for which the conjecture is known to hold are $n \leq 2k+3$ (Pilaud and Santos, 2012) and $(2, 8)$ (Bokowski and Pilaud, 2009). Using ideas from rigidity theory we realize $\Delta_k(n)$ as a polytope for $(k, n) \in \{(2, 9), (2, 10), (3, 10)\}$. We also realize it as a simplicial fan for all $n \leq 13$ and arbitrary k, except the pairs $(3, 12)$ and $(3, 13)$.

Related Version arXiv:2212.14265

1 The multiassociahedron

Triangulations of the convex n-gon P $(n > 2)$ are the facets of an abstract simplicial complex with vertex set $\binom{[n]}{2}$ and defined by taking as simplices all the non-crossing sets of diagonals. This simplicial complex, ignoring the boundary edges $\{i, i+1\}$, is a polytopal sphere of dimension $n-4$ dual to the *associahedron*. (Here and all throughout the paper, indices for vertices of the n-gon are regarded modulo n).

A similar complex can be defined if we forbid crossings of more than a certain number k of edges (assuming $n > 2k+1$), instead of forbidding pairwise crossings.

▶ **Definition 1.1.** Two disjoint elements $\{i, j\}, \{k, l\} \in \binom{[n]}{2}$, with $i < j$ and $k < l$, of $\binom{[n]}{2}$ *cross* if $i < k < j < l$ or $k < i < l < j$. That is, if they cross when seen as diagonals of a convex n-gon. A k-*crossing* is a subset of k elements of $\binom{[n]}{2}$ such that every pair cross. A subset of $\binom{[n]}{2}$ is $(k+1)$-*free* if it doesn't contain any $(k+1)$-crossing. A k-*triangulation* is a maximal $(k+1)$-free set. We call $\Delta_k(n)$ the simplicial complex consisting of $(k+1)$-free sets of diagonals, whose facets are the k-triangulations.

Diagonals of length at most k (where length is measured cyclically) cannot participate in any $(k+1)$-crossing. Thus, it makes sense to define the reduced complex $\overline{\Delta}_k(n)$ obtained from $\Delta_k(n)$ by deleting them. We call $\overline{\Delta}_k(n)$ the *multiassociahedron* or k-associahedron. See [16, 17, 20] for additional information.

It was proved in [15, 9] that every k-triangulation of the n-gon has exactly $k(2n-2k-1)$ diagonals. That is, $\Delta_k(n)$ is pure of dimension $k(2n-2k-1)-1$. Jonsson [11] further proved that the reduced version $\overline{\Delta}_k(n)$ is a shellable sphere of dimension $k(n-2k-1)-1$, and conjectured it to be the normal fan of a polytope.

[*] Supported by grant PID2019-106188GB-I00 funded by MCIN/AEI/10.13039/501100011033, by FPU19/04163 of the Spanish Government, and by project CLaPPo (21.SI03.64658) of Universidad de Cantabria and Banco Santander

▶ Conjecture 1.2 ([11]). For every $n \geq 2k+1$ the complex $\overline{\Delta}_k(n)$ is a polytopal sphere. That is, there is a simplicial polytope of dimension $k(n-2k-1)-1$ and with $\binom{n}{2} - kn$ vertices whose lattice of proper faces is isomorphic to $\overline{\Delta}_k(n)$.

Conjecture 1.2 is easy to prove for $n \leq 2k+3$. $\overline{\Delta}_k(2k+1)$ is indeed a -1-sphere (the complex whose only face is the empty set). $\overline{\Delta}_k(2k+2)$ is the face poset of a $(k-1)$-simplex, and $\overline{\Delta}_k(2k+3)$ is (the polar of) the cyclic polytope of dimension $2k-1$ with n vertices (Lemma 8.7 in [17]). The only additional case for which Jonsson's conjecture is known to hold is $k=2$ and $n=8$ [2]. In some additional cases $\overline{\Delta}_k(n)$ has been realized as a complete simplicial fan, but it is open whether this fan is polytopal. This includes the cases $n \leq 2k+4$ [1], the cases $k=2$ and $n \leq 13$ [14] and the cases $k=3$ and $n \leq 11$ [1].

Interest in the polytopality of $\overline{\Delta}_k(n)$ also comes from cluster algebras and Coxeter combinatorics. Let $w \in W$ be an element in a Coxeter group W and let Q be a word of a certain length N. Assume that Q contains as a subword a reduced expression for w. The *subword complex* of Q and w is the simplicial complex with vertex set $[N]$ and with faces the subsets of positions that can be deleted from Q and still contain a reduced expression for w. Knutson and Miller [13, Theorem 3.7 and Question 6.4] proved that every subword complex is either a shellable ball or sphere, and they asked whether all spherical subword complexes are polytopal. It was later proved by Stump [20, Theorem 2.1] that $\overline{\Delta}_k(n)$ is a spherical subword complex for the Coxeter system of type A_{n-2k-1} and, moreover, it is *universal*: every other spherical subword complex of type A appears as a link in some $\overline{\Delta}_k(n)$ [18, Proposition 5.6]. Hence, Conjecture 1.2 is equivalent to a positive answer in type A to the question of Knutson and Miller.

2 Realizing a simplicial complex as a polytope

If Δ is a pure simplicial complex with vertex set V of dimension $D-1$ (so that its facets have size D) to realize it as a polytope we only need to find a vector configuration $\mathcal{V} = \{v_i\}_{i \in V} \subset \mathbb{R}^D$ on which Δ yields a *complete simplicial fan*, and then prove that the fan is a *regular triangulation* of $\mathcal{V}$. See [8, Section 9.5] for details.

We want to apply this to the complex $\overline{\Delta}_k(n)$, for which $V \subset \binom{[n]}{2}$ and $D = k(n-2k-1)$. This complex has two important properties. On the one hand, it is a *pseudo-manifold*. From the point of view of k-triangulations this is equivalent to the following property, which defines *flips* among k-triangulations:

▶ Proposition 2.1 (Flips [17, section 5]). For every edge f of a k-triangulation T with length greater than k, there is a unique edge $e \in \binom{[n]}{2}$ such that

$$T \triangle \{e, f\} := T \setminus \{f\} \cup \{e\}$$

is another k-triangulation.

On the other hand, since $\overline{\Delta}_k(n)$ is a sphere, the link of every face of codimension two is a cycle. We will need the following property of these cycles, that we prove in the full version of this paper [6, Corollary 2.13]:

▶ Proposition 2.2 (Short cycles). All links of dimension 1 in $\overline{\Delta}_k(n)$ are cycles of length ≤ 5.

With these two properties, the following is our main result about how to realize $\overline{\Delta}_k(n)$ [6, Corollary 2.13], which is a version of [8, Corollary 4.5.20] adapted to our case:

▶ **Theorem 2.3.** *Let $\mathcal{V} = \{v_{ij}\}_{\{i,j\} \in \binom{[n]}{2}} \subset \mathbb{R}^{k(2n-2k-1)}$ be a vector configuration. $\mathcal{V}$ embeds $\overline{\Delta}_k(n)$ as a complete fan in $\mathbb{R}^{k(n-2k-1)}$ if and only if it satisfies the following properties:*

1. *(Basis collection) For every facet (k-triangulation) T, the vectors $\{v_{ij} : \{i,j\} \in T\}$ are a linear basis.*
2. *(Interior Cocircuit Property, ICoP) For every flip between two k-triangulations T_1 and T_2, the unique linear dependence among the vectors $\{v_{ij} : \{i,j\} \in T_1 \cup T_2\}$ has the same sign for the two elements involved in the flip (the unique elements in $T_1 \setminus T_2$ and $T_2 \setminus T_1$).*
3. *(Elementary cycles of length 5) Given a codimension 2 face ρ whose link Z is a cycle of length five (see Proposition 2.2), there are three consecutive elements $i_1, i_2, i_3 \in Z$ such that the unique linear dependence among the vectors $\{v_i : i \in \rho \cup \{i_1, i_2, i_3\}\}$ has opposite sign for i_2 than the sign it takes for i_1 and i_3.*

Once we have the complete fan, we need it to be polytopal, which is equivalent to the feasibility of a system of linear inequalities. For this we use a version of [19, Theorem 3.7], which in turn is closely related to [8, Proposition 5.2.6(i)].

3 Rigidity

The number $k(2n-2k-1) = 2kn - \binom{2k+1}{2}$ of edges in a k-triangulation happens to coincide with the rank of any *abstract rigidity matroid* of dimension $2k$ on n elements. These matroids capture and generalize the combinatorial rigidity of graphs with n vertices generically embedded in $\mathbb{R}^{2k}$. This numerical coincidence (plus some evidence) led [17] to conjecture that *all k-triangulations of the n-gon are bases in the generic bar-and-joint rigidity matroid of n points in dimension $2k$.* If this is true then a natural way to apply Theorem 2.3 is to use as vector configuration $\mathcal{V}$ the rows of the corresponding rigidity matrix, as follows.

Let $\mathbf{p} = (p_1, \dots, p_n)$ be a set of n points in $\mathbb{R}^d$, labelled by $[n]$. Their *bar-and-joint rigidity matrix* is the following $\binom{n}{2} \times nd$ matrix:

$$R(\mathbf{p}) := \begin{pmatrix} p_1 - p_2 & p_2 - p_1 & 0 & \dots & 0 & 0 \\ p_1 - p_3 & 0 & p_3 - p_1 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & & \vdots & \vdots \\ p_1 - p_n & 0 & 0 & \dots & 0 & p_n - p_1 \\ 0 & p_2 - p_3 & p_3 - p_2 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & 0 & \dots & p_{n-1} - p_n & p_n - p_{n-1} \end{pmatrix}. \qquad (1)$$

The shape of the matrix is as follows: there is a row for each pair $\{i,j\} \in \binom{[n]}{2}$, so rows can be considered labeled by edges in the complete graph K_n. Then, there are n blocks of columns, one for each point p_i and with d columns in each block; in the row of an edge $\{i,j\}$ (or $\{j,i\}$) only the blocks of vertices i and j ae nonzero, and they contain respectively the vectors $p_i - p_j$ and $p_j - p_i$. Put differently, the matrix can be interpreted as a "directed incidence matrix" of the complete graph K_n, except instead of having a single $+1$ and -1 corresponding each incidence between a vertex and an edge we have the d-dimensional vectors $p_i - p_j$ and $p_j - p_i$. For any $E \subset \binom{[n]}{2}$ we denote by $R(\mathbf{p})|_E$ the restriction of $R(\mathbf{p})$ to the rows or elements indexed by E.

▶ **Definition 3.1.** Let $E \subset \binom{[n]}{2}$ be a subset of edges of K_n (equivalently, of rows of $R(\mathbf{p})$). We say that E, or the corresponding subgraph of K_n, is *self-stress-free* or *independent* if the rows of $R(\mathbf{p})|_E$ are linearly independent, and *rigid* or *spanning* if they are linearly spanning (that is, they have the same rank as the whole matrix $R(\mathbf{p})$).

Put differently, self-stress-free and rigid graphs are, respectively, the independent and spanning sets in the linear matroid of rows of $R(\mathbf{p})$. We call this matroid the *bar-and-joint rigidity matroid* of $\mathbf{p}$ and denote it $\mathcal{R}(\mathbf{p})$. The following two matrices "with the same shape" (in particular, with the same size) as $R(\mathbf{p})$ also define abstract rigidity matroids:

- The *hyperconnectivity* matroid of $\mathbf{p} \subset \mathbb{R}^d$, denoted $\mathcal{H}(\mathbf{p})$, is the matroid of rows of

$$H(\mathbf{p}) := \begin{pmatrix} p_2 & -p_1 & 0 & \dots & 0 & 0 \\ p_3 & 0 & -p_1 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & & \vdots & \vdots \\ p_n & 0 & 0 & \dots & 0 & -p_1 \\ 0 & p_3 & -p_2 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & 0 & \dots & p_n & -p_{n-1} \end{pmatrix} \tag{2}$$

- For points $\mathbf{q} = (q_1, \dots, q_n)$ in $\mathbb{R}^2$ and a parameter $d \in \mathbb{N}$, the d-dimensional *cofactor rigidity* matroid of the points $q_1, \dots, q_n$, which we denote $\mathcal{C}_d(\mathbf{q})$, is the matroid of rows of

$$C_d(\mathbf{q}) := \begin{pmatrix} \mathbf{c}_{12} & -\mathbf{c}_{12} & 0 & \dots & 0 & 0 \\ \mathbf{c}_{13} & 0 & -\mathbf{c}_{13} & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & & \vdots & \vdots \\ \mathbf{c}_{1n} & 0 & 0 & \dots & 0 & -\mathbf{c}_{1n} \\ 0 & \mathbf{c}_{23} & -\mathbf{c}_{23} & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & 0 & \dots & \mathbf{c}_{n-1,n} & -\mathbf{c}_{n-1,n} \end{pmatrix}, \tag{3}$$

  where the vector $\mathbf{c}_{ij} \in \mathbb{R}^d$ associated to $q_i = (x_i, y_i)$ and $q_j = (x_j, y_j)$ is

$$\mathbf{c}_{ij} = \left((x_i - x_j)^{d-1}, (y_i - y_j)(x_i - x_j)^{d-2}, \dots, (y_i - y_j)^{d-1}\right).$$

  For $d = 1$ this is independent of the choice of $\mathbf{p}$ and equals the directed incidence matrix of K_n. For $d = 2$ we recover the two-dimensional bar-and-joint rigidity matrix $\mathbf{p}$.

In [4] we prove that the three theories coincide when the points $\mathbf{p}$ or $\mathbf{q}$ are chosen along the moment curve (for bar-and-joint and hyperconnectivity) and the parabola (for cofactor). More precisely:

▶ **Theorem 3.2** ([4]). *Let $t_1 < \dots < t_n \in \mathbb{R}$ be real parameters. Let*

$$p_i = (1, t_i, \dots, t_i^{d-1}) \in \mathbb{R}^d, \quad p'_i = (t_i, t_i^2, \dots, t_i^d) \in \mathbb{R}^d, \quad q_i = (t_i, t_i^2) \in \mathbb{R}^2.$$

Then, the matrices $H(p_1, \dots, p_n)$, $R(p'_1, \dots, p'_n)$ and $C(q_1, \dots, q_n)$ can be obtained from one another multiplying on the right by a regular matrix and then multiplying its rows by some positive scalars. In particular, the rows of the three matrices define the same oriented matroid.

▶ **Definition 3.3.** We call the matrix $H(p_1, \dots, p_n)$ in the statement of Theorem 3.2 the *polynomial d-rigidity matrix with parameters $t_1, \dots, t_n$*. We denote it $P_d(t_1, \dots, t_n)$, and denote $\mathcal{P}_d(t_1, \dots, t_n)$ the corresponding matroid.

Now, for any choice of points $\mathbf{p} \in \mathbb{R}^{2k}$ or $\mathbf{q} \in \mathbb{R}^2$ in general position, the rows of the matrices $R(\mathbf{p})$, $H(\mathbf{p})$ or $C_{2k}(\mathbf{q})$ are a real vector configuration $\mathcal{V} \subset \mathbb{R}^{2kn}$ of rank $k(2n-2k-1)$ (since this is the rank of in any abstract rigidity matroid). Moreover, if $\mathbf{p}$ is chosen along the moment curve or $\mathbf{q}$ along the parabola, then the three theories give linearly equivalent embeddings. The question we address is whether using these vectors as the rays for a realization of the k-associahedron $\Delta_k(n)$ we get that the reduced version $\overline{\Delta}_k(n)$ is a polytopal fan. We pose the conjecture that positions along the moment curve realizing $\overline{\Delta}_k(n)$ as a basis collection exist for every k and n:

▶ Conjecture 3.4. k-triangulations of the n-gon are isostatic (that is, bases) in the bar-and-joint rigidity matroid of generic points along the moment curve in dimension $2k$.

This conjecture implies the one from [17] mentioned above, but it would imply the same for the generic cofactor rigidity matroid and for the generic hyperconnectivity matroid. In fact the latter is already known to hold by a previous result of ours [5, Corollary 2.17].

4 Main results

The proofs of the following results are included in the full version.

First, as evidence for Conjecture 3.4 we prove the case $k = 2$:

▶ **Theorem 4.1** ([6, Theorem 1.4]). *2-triangulations are isostatic in dimension $2k$ for generic positions along the moment curve.*

One may be tempted to change "generic" to "arbitrary" in Conjecture 3.4, but we show that this stronger conjecture fails in the worst possible way: for every $k \geq 3$ and $n \geq 2k+3$, the standard positions along the moment curve make some k-triangulation not a basis.

▶ **Theorem 4.2** ([6, Theorem 1.6]). *The graph $K_9 - \{16, 37, 49\}$ is a 3-triangulation of the n-gon, but it is dependent in the rigidity matroid $\mathcal{C}_6$ for any configuration $\{q_1, q_9\} \subset \mathbb{R}^2$ if the lines through q_1q_6, q_3q_7, and q_4q_9 meet at a point. This occurs, for example, if we take the nine points on the parabola with $t_i = i$.*

In fact, for $n \leq 2k+3$ we can characterize exactly what positions realize $\overline{\Delta}_k(n)$ as a fan, for cofactor rigidity (and, in particular, for the other two forms of rigidity with positions along the moment curve). In the case $n = 2k+3$ this is governed by the geometry of the star-polygon formed by the k-relevant edges. More precisely, we call "big side" of each relevant edge (that is, edge of $k+1$) in a $(2k+3)$-gon the open half-plane containing $k+1$ vartices:

▶ **Theorem 4.3** ([6, Theorem 3.14]). ***1.*** *For $n = 2k+2$, any choice of $q_1, \dots, q_{2k+2} \in \mathbb{R}^2$ in convex position realizes $\overline{\Delta}_k(n)$ as a polytopal fan.*

2. *Let $q_1, q_2, \dots, q_{2k+3} \in \mathbb{R}^2$ be in convex position. $\overline{\Delta}_k(2k+3)$ is realized by $C_{2k}(q_1, \dots, q_{2k+3})$ as a complete fan if and only if the big sides of all relevant edges have a non-empty intersection.*

Interestingly, from part (2) of this result it is quite easy to show that *no positions* of points along the moment curve realize $\overline{\Delta}_3(12)$. More generally:

▶ **Corollary 4.4** ([6, Theorem 1.7]). *If $k \geq 3, n \geq 2k+6$ then no choice of points $\mathbf{q} \subset \mathbb{R}^2$ in convex position realizes $\overline{\Delta}_k(n)$ as a fan via cofactor rigidity.*

Observe that this is not a counter-example to Conjecture 3.4; it implies that via rigidity along the moment curve we cannot achieve the three conditions of Theorem 2.3, but Conjecture 3.4 is about condition 1 alone.

Finally, for every $n \leq 13$ we have experimentally found positions along the moment curve realizing $\overline{\Delta}_k(n)$ as a fan, except in the cases $(n, k) \in \{(3, 12), (3, 13)\}$ which are forbidden by Corollary 4.4. For many of them we have also realized the polytope:

▶ **Theorem 4.5** ([6, Lemma 4.13])**.** *Let* $\mathbf{t} = \{1, 2, \ldots, n\}$ *be standard positions for the parameters. Then:*

1. *Standard positions for* $P_4(\mathbf{t})$ *realize* $\overline{\Delta}_2(n)$ *as the normal fan of a polytope if* $n \leq 9$*.*
2. *The non-standard positions* $\mathbf{t} = (-2, 1, 2, 3, 4, 5, 6, 7, 9, 20)$ *for* $P_4(\mathbf{t})$ *realize* $\overline{\Delta}_2(10)$ *as the normal fan of a polytope.*
3. *Standard positions for* $P_4(\mathbf{t})$ *realize* $\overline{\Delta}_2(n)$ *as a complete fan for all* $n \leq 13$*.*

▶ **Theorem 4.6** ([6, Lemma 4.14])**.** *Equispaced positions along the circle realize* $\overline{\Delta}_k(n)$ *as a fan for* $(n, k) \in \{(3, 10), (3, 11), (4, 12), (4, 13)\}$*. The first one is polytopal.*

References

1 Nantel Bergeron, Cesar Ceballos, Jean-Philippe Labbé, Fan realizations of subword complexes and multi-associahedra via Gale duality, *Discrete Comput. Geom.* 54(1), 195–231 (2015).

2 Jürgen Bokowski, Vincent Pilaud, *On symmetric realizations of the simplicial complex of 3-crossing-free sets of diagonals of the octagon*, Proceedings of the 21st Annual Canadian Conference on Computational Geometry, Vancouver, British Columbia, Canada, August 17–19, 2009.

3 Cesar Ceballos, Jean-Philippe Labbé, and Christian Stump. Subword complexes, cluster complexes, and generalized multi-associahedra. *J. Algebraic Combin.*, 39(1):17–51, 2014.

4 Luis Crespo Ruiz, Francisco Santos, *Bar-and-joint rigidity on the moment curve coincides with cofactor rigidity on a conic*, preprint arXiv:2106.08923, 2021. Accepted in *Combinatorial Theory.*

5 Luis Crespo Ruiz, Francisco Santos, *Multitriangulations and tropical Pfaffians*, preprint arXiv:2203.04633, 2022

6 Luis Crespo Ruiz, Francisco Santos, *Realizations of multiassociahedra via rigidity*, preprint arXiv:2212.14265, 2022.

7 Peter R. Cromwell, *Polyhedra*, Cambridge University Press, 1997.

8 Jesús A. De Loera, Jörg Rambau, Francisco Santos, *Triangulations: Structures for Algorithms and Applications*, Springer-Verlag, 2012.

9 Andreas Dress, Jack H. Koolen and Vincent Moulton. On line arrangements in the hyperbolic plane. *Eur. J. Comb.*, **23**(5) (2002), 549–557.

10 Branko Grünbaum, Geoffrey C. Shephard, *Tilings and Patterns*, W. H. Freeman & Co., 1989.

11 Jakob Jonsson, Generalized triangulations and diagonal-free subsets of stack polyominoes, *J. Comb. Theory Ser. A* **112**(1) (2005), 117–142.

12 Gil Kalai, Hyperconnectivity of graphs, *Graphs and Combinatorics*, **1** (1985), 65–79.

13 Allen Knutson and Ezra Miller. Subword complexes in Coxeter groups. *Adv. Math.*, 184(1) (2004), 161–176.

14 Thibault Manneville, Fan realizations for some 2-associahedra, *Experimental Mathematics*, 27(4), 377–394 (2017).

15 Tomoki Nakamigawa. A generalization of diagonal flips in a convex polygon. *Theor. Comput. Sci.* **235**(2) (2000), 271–282.

16 Vincent Pilaud, Michel Pocchiola, Multitriangulations, Pseudotriangulations and Primitive Sorting Networks. *Discrete Comput. Geom.* **41** (2012), 142–191.

17 Vincent Pilaud, F. Santos, Multitriangulations as Complexes of Star Polygons, *Discrete Comput. Geom* 41 (2009), 284–317.

18 Vincent Pilaud, Francisco Santos, The brick polytope of a sorting network. *European J. Combin.* 33:4 (2012), 632–662.

19 Günter Rote, Francisco Santos, Ileana Streinu, *Expansive Motions and the Polytope of Pointed Pseudo-Triangulations*, In "Discrete and Computational Geometry – The Goodman-Pollack Festschrift" (B. Aronov, S. Basu, J. Pach, M. Sharir, eds), Algorithms and Combinatorics 25, Springer Verlag, Berlin, June 2003, pp. 699–736.

20 Christian Stump. A new perspective on k-triangulations. *J. Comb. Theory A* **118**(6) (2011), 1794–1800.

21 Walter Whiteley, *Some Matroids from Discrete Applied Geometry*, in *Matroid Theory* (Joseph E. Bonin, James G. Oxley and Brigitte Servatius, Editors), Contemporary Mathematics **197**, 1996, pp. 171–311

Bichromatic Perfect Matchings with Crossings*

Oswin Aichholzer[1], **Stefan Felsner**[2], **Rosna Paul**[1], **Manfred Scheucher**[2], **and Birgit Vogtenhuber**[1]

1 **Institute of Software Technology, Graz University of Technology, Austria**
`oaich,ropaul,bvogt@ist.tugraz.at`

2 **Institute for Mathematics, Technical University of Berlin, Germany**
`felsner,scheucher@math.tu-berlin.de`

Abstract

We consider bichromatic point sets with n red and n blue points and study straight-line bichromatic perfect matchings on them. We show that every such point set in convex position admits a matching with at least $\frac{3n^2}{8} - O(n)$ crossings. Moreover, this bound is asymptotically tight since for any $k > \frac{3n^2}{8}$ there exist bichromatic point sets that do not admit any perfect matching with k crossings.

1 Introduction

Let $P = R \cup B$, $|R| = |B| = n$ be a point set in *general position*, that is, no three points of P are collinear. We refer to R and B as the set of red and blue points, respectively. A straight-line matching M of P where every point in R is uniquely matched to a point in B is called a *straight-line bichromatic perfect matching* (all matchings considered in this work are straight-line, so we will mostly omit this term). In this work, we study the existence of bichromatic perfect matchings with a fixed number k of crossings on P, where $0 \leq k \leq \binom{n}{2}$. It is well known and easy to see that there exists a crossing-free (that is, $k = 0$) such matching for every P [7]. Perfect matchings with k crossings on uncolored point sets have been considered in [2]. There it is shown that for every $k \leq \frac{n^2}{16} - O(n\sqrt{n})$, every point set of size $2n$ admits a perfect matching with exactly k crossings and that there exist such point sets where every perfect matching has fewer than $\frac{20n^2}{72}$ crossings. As a direct consequence, there exist bichromatic point sets which do not admit bichromatic perfect matchings with k crossings for $k > \frac{20n^2}{72}$. On the other hand, $2n$ uncolored points in convex position admit perfect matchings with k crossings for all k, where $0 \leq k \leq \binom{n}{2}$ [2]. But when we color the points, the situation changes quite drastically.

Consider a point set P of $2n$ points in convex position (convex point set, for short) with an alternating coloring, that is, every second point along the convex hull is red (and the other points are blue). Moreover, let the number n of red (and blue) points be even. Then the number of crossings in a bichromatic perfect matching M on P is at most $\frac{n(n-2)}{2} = \binom{n}{2} - \frac{n}{2}$. The idea is as follows: Label the points of P as $p_0, p_1, \ldots, p_{2n-1}$ along the boundary of the convex hull. The point p_i cannot be matched to p_{i+n} since both points are of the same color. Hence, for any edge e in such a matching M of P, the number of crossings of e is at most $n-2$. As every crossing involves two edges, the number of crossings in M is at most $\frac{n(n-2)}{2} = \binom{n}{2} - \frac{n}{2}$. This bound is tight, since it is possible to construct a bichromatic perfect matching M on P with exactly $\binom{n}{2} - \frac{n}{2}$ crossings as follows. For $0 \leq i \leq n-1$, match the point p_i to the point p_{i+n+1}, when i is even. Otherwise, match p_i to p_{i+n-1}. Together this leads to the following question.

* O.A. and R.P. supported by the Austrian Science Fund (FWF): W1230. S.F. supported by the DFG Grant FE 340/13-1. M.S. supported by the DFG Grant SCHE 2214/1-1.

▶ **Open Problem 1.** *For which values of k does every bichromatic convex point set $P = R \cup B$, $|R| = |B| = n$, admit a straight-line bichromatic perfect matching with exactly k crossings?*

The above example implies that if $k > \binom{n}{2} - \frac{n}{2}$, there exist bichromatic point sets with n red and n blue points that do not have any bichromatic perfect matching with k crossings. Thus, Open Problem 1 can be true only for $k \leq \binom{n}{2} - \frac{n}{2}$. In this paper, we further improve the bound on k as follows.

▶ **Theorem 1.1.** *For any $k > \frac{3n^2}{8}$, there exists a bichromatic convex point set with n red and n blue points that does not have a straight-line bichromatic perfect matching with k crossings.*

Related work: A survey by Kano and Urrutia [6] gives an overview of various problems on bichromatic point sets, including matching problems. Crossing-free bichromatic perfect matchings have been studied from various perspectives such as their structure [5, 8], linear transformation distance [1], and matchings compatible to each other [3, 4]. Sharir and Welzl [9] proved that the number of crossing-free bichromatic perfect matchings on $2n$ points is at most $O(7.61^n)$. However, not much is known about the number or existence of bichromatic perfect matchings with k crossings, for $k > 0$.

2 Bichromatic Convex Point Sets

Let $\mathcal{C}_{n,n}$ be the collection of all bichromatic convex point sets $P = R \cup B$ with $|R| = |B| = n$. For a point set $P \in \mathcal{C}_{n,n}$, we label the points in P in clockwise direction along the convex hull as $p_0, p_1, \ldots, p_{2n-1}$ and refer to this as the *clockwise ordering*. We will consider all indices modulo $2n$. The number of crossings in any birchromatic perfect matching M_P of P is denoted by $\overline{\text{cr}}(M_P)$. If M_P has the maximum number of crossings among all such matchings of P, then it is called a *max-crossing* matching on P. Among all max-crossing matchings for all $P \in \mathcal{C}_{n,n}$, we are interested in max-crossing matchings which have the minimum number of crossings. We call such a matching a *min-max-crossing matching* of $\mathcal{C}_{n,n}$. Further, from now on, we refer to bichromatic perfect matchings just as *matchings*, unless otherwise stated.

For $P \in \mathcal{C}_{n,n}$ we define the collection of all points of the same color which are consecutive in clockwise order as a *block*. For example, if $R_1 = \{p_a, p_{a+1}, \ldots, p_{a+s}\}$ is a block of red points, then p_{a-1} and p_{a+s+1} are blue, and the next collection will be a block of blue points including p_{a+s+1}. By repeating the process along the boundary of the convex hull of P, we get a collection of blocks $\{R_1, B_1, R_2, B_2, \ldots, R_s, B_s\}$ w.r.t. the clockwise order such that $|R_1 \cup R_2 \cup \ldots \cup R_s| = |B_1 \cup B_2 \cup \ldots \cup B_s| = n$. These blocks are non-empty and alternate in color. A bichromatic convex point set with the collection of blocks $\{R_1, B_1, R_2, B_2, \ldots, R_s, B_s\}$ is called a $2s$*–block* coloring. For example, a $2n$–block coloring is an alternating coloring. If all $2s$ blocks have the same cardinality, then the coloring is called a *balanced* $2s$*–block* coloring. See Figure 2(a) for a bichromatic point set with a balanced 4–block coloring. Strictly speaking, a balanced 4–block coloring can only be achieved if n is even. If n is odd, the cardinalities of some blocks differ by at least ± 1. We call a 4–block coloring with maximum cardinality difference of ± 1 *nearly balanced*. Considering the properties that we discuss in this paper, nearly balanced 4–block colorings and balanced 4–block colorings behave very similarly. So abusing the terminology a bit, we will mostly refer to all of them as balanced 4–block colorings. With these definitions we can now formulate a stronger version of Theorem 1.1.

▶ **Theorem 2.1.** *Let $P \in \mathcal{C}_{n,n}$ have a balanced 4–block coloring and let $\mathrm{M}_P^{\vee}$ be a max-crossing matching of P. Then $\overline{\text{cr}}(\mathrm{M}_P^{\vee}) = \frac{3n^2}{8} - O(n)$. Moreover, $\mathrm{M}_P^{\vee}$ is a min-max-crossing matching of the set $\mathcal{C}_{n,n}$.*

Clearly, Theorem 2.1 implies Theorem 1.1, as any bichromatic convex point set with a balanced 4–block coloring satisfies Theorem 1.1. Theorem 2.1 will follow from Lemma 2.4 and Lemma 2.5 below which are stated and shown in the next sections.

2.1 Min-Max-Crossing Matching for 4–block colorings

In this section, we construct a min-max-crossing matching among all bichromatic convex point sets with a 4–block coloring.

▶ **Lemma 2.2.** *Let $P \in \mathcal{C}_{n,n}$ have a 4–block coloring with blocks R_1, B_1, R_2, B_2 and let $\mathrm{M}_P^{\vee}$ be a max-crossing matching on P. Then for any block $X \in \{R_1, R_2, B_1, B_2\}$, the edges emanating from X form a* crossing family*, that is, every pair of these edges forms a crossing.*

Proof. Let e and f be two edges of $\mathrm{M}_P^{\vee}$ such that $e = r_i b_{i'}$ and $f = r_j b_{j'}$ where $r_i, r_j \in X$ with $i < j$. For the sake of contradiction, assume that e and f do not cross. By replacing e and f by $e' = r_i b_{j'}$ and $f' = r_j b_{i'}$, we get a new matching, say M'_P. As all points are in convex position, the edges crossing e and f will also cross the new edges e' and f'. Also, the edges crossing exactly one of e and f will cross exactly one of e' and f'. In addition, e' and f' cross each other. Hence the number of crossings of M'_P is strictly larger than the number of crossings of $\mathrm{M}_P^{\vee}$, a contradiction. ◀

Recall the clockwise ordering of the points in P. Without loss of generality, assume that $R_1 = \{p_1, p_2, \ldots, p_{|R_1|}\}$. Let $\mathrm{M}_P^{\vee}$ be a max-crossing matching of P. Assume that in $\mathrm{M}_P^{\vee}$, the point $p_i \in R_1$ is matched to a point (say p'_i) in B_2, and $p_j \in R_1$ is matched to a point (say p'_j) in B_1 such that $i < j$. By the clockwise ordering of P, $i < j < j' < i'$ and hence $p_i p'_i$ does not cross $p_j p'_j$. This is a contradiction by Lemma 2.2. Thus, in $\mathrm{M}_P^{\vee}$, if $p_i \in R_1$ is matched to a point in B_2, then all $p_j \in R_1$ with $i < j$ must be matched to a point in B_2. More precisely, there exists an integer a_1 for R_1 such that all $p_i \in R_1$ with $i \le a_1$ are matched to B_1 and all $p_i \in R_1$ with $i > a_1$ are matched to B_2. In a similar way we can show that all $p_i \in B_1$ with $i \le |R_1| + |B_1| - a_1$ have to be matched to R_2 and the remaining unmatched points have to be matched to R_1. In other words, there exists an integer a_1 such that first a_1 points of R_1 are matched to the last a_1 points of B_1 (as a crossing family). The remaining last $|R_1| - a_1$ points of R_1 have to be matched to the first $|R_1| - a_1$ points of B_2 (as a crossing family). This fixes the remaining edges of $\mathrm{M}_P^{\vee}$. That is, the remaining last $n - |B_1| - |R_1| + a_1$ points of B_2 must be matched, as a crossing family, to the first $n - |B_1| - |R_1| + a_1$ points of R_2 (see Figure 1). Finally, match the remaining points as a crossing family. Hence, to get a max-crossing matching on P, it is sufficient to determine the optimal value of a_1.

▶ **Lemma 2.3.** *Let $P \in \mathcal{C}_{n,n}$ have a 4–block coloring with blocks R_1, B_1, R_2, and B_2. Let M_P be a matching on P such that the first x points of the set R_1 are matched to the last x points of B_1, as a crossing family. Then M_P is a max-crossing matching on P iff $x = \frac{1}{2}(|R_1| + |B_1| - \frac{n}{2})$.*

Proof. Consider a matching M_P with the above property. Assume that $|R_1| = r_1 \ge \frac{n}{2}$ and $|B_1| = b_1 \ge \frac{n}{2}$. The number of pairs of non-crossing edges in M_P is obtained by $(r_1 - x)(b_1 - x) + x(n - r_1 - b_1 + x) = r_1 b_1 - 2x b_1 - 2x r_1 + nx + 2x^2$. As we want to find the value of x that gives the maximum number of crossings, we calculate the value of x such that $f(x) = (n - 2r_1 - 2b_1)x + 2x^2 + r_1 b_1$ attains the minimum. This is achieved by setting the first derivative of $f(x)$ to zero as $f''(x) > 0$. We get $f'(x) = 0$ for $x = \frac{1}{2}(r_1 + b_1 - \frac{n}{2})$. ◀

By the proof of Lemma 2.3 the minimum number of pairs of non-crossing edges in $\mathrm{M}_P^{\vee}$ is $r_1 b_1 - \frac{1}{2}(r_1 + b_1 - \frac{n}{2})^2$. This gives the exact structure and the number of crossings in $\mathrm{M}_P^{\vee}$.

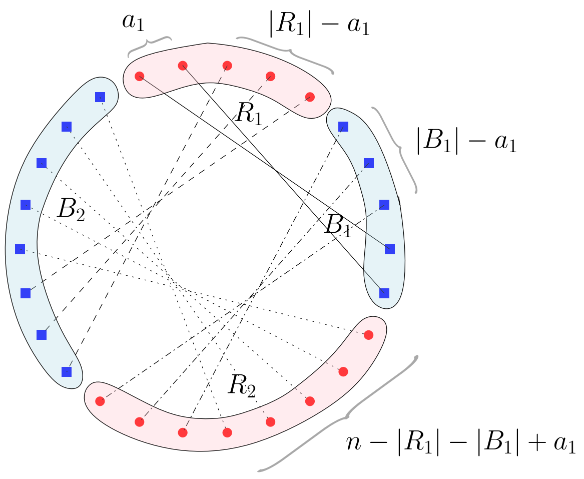


Figure 1 Structure of a max-crossing matching in a 4–block coloring.

The number of crossings $\overline{\mathrm{cr}}(\mathrm{M}_P^{\vee})$ can be obtained by subtracting the number of non-crossing edge pairs from the theoretical maximum number of the crossing edge pairs in $\mathrm{M}_P^{\vee}$. That is, $\overline{\mathrm{cr}}(\mathrm{M}_P^{\vee}) = \binom{n}{2} - (r_1 b_1 - \frac{1}{2}(r_1 + b_1 - \frac{n}{2})^2)$. Next we check which coloring, among all 4–block colorings, produces a min-max-crossing matching for all 4–block colorings. Full proofs for statements marked with ($\star$) can be found in the full version of this paper.

▶ **Lemma 2.4.** *($\star$) Let $P \in \mathcal{C}_{n,n}$ have a balanced 4–block coloring and let $\mathrm{M}_P^{\vee}$ be a max-crossing matching of P. Then $\overline{\mathrm{cr}}(\mathrm{M}_P^{\vee}) = \frac{3n^2}{8} - O(n)$. Moreover, $\mathrm{M}_P^{\vee}$ is a min-max-crossing matching for all the 4-block colored point sets of size $2n$.*

Proof sketch. The minimum number of pairs of non-crossing edges in $\mathrm{M}_P^{\vee}$ is given by $h(r_1, b_1) = r_1 b_1 - \frac{1}{2}(r_1 + b_1 - \frac{n}{2})^2$. Since we want to find the values of r_1 and b_1 that minimizes the number of crossings among max-crossing matchings of the 4–block colorings, we need to maximize $h(r_1, b_1)$ for $r_1, b_1 \geq \frac{n}{2}$. By analyzing the partial derivatives of the function h, it follows that h attains its maximum when $r_1 = b_1 = \frac{n}{2}$. This shows that the number of crossings in a max-crossing matching is minimized on a balanced 4–block coloring. If n is a multiple of 4, then $\overline{\mathrm{cr}}(\mathrm{M}_P^{\vee}) = \frac{3n^2}{8} - \frac{n}{2}$. Otherwise, $\overline{\mathrm{cr}}(\mathrm{M}_P^{\vee})$ is either $\lfloor \frac{3n^2}{8} - \frac{n}{2} \rfloor$ or $\lceil \frac{3n^2}{8} - \frac{n}{2} \rceil$. These values are obtained for $r_1 = b_1 = \lceil \frac{n}{2} \rceil$ and $x = \frac{1}{2}(r_1 + b_1 - \frac{n}{2})$ by analyzing the structure of matchings in each of the remaining cases. ◀

We remark that the balanced 4–block coloring is not the only coloring that gives the min-max-crossing matching. See Figure 2(a) for the matching constructed in the above proof and Figure 2(b) for a different example.

2.2 Min-Max-Crossing Matching for all colorings

In the following, we extend Lemma 2.4 to all bichromatic convex point sets. Let $P \in \mathcal{C}_{n,n}$. For any point $v \in P$, the point $w \in P$ is called the antipodal point of v, if the line through v and w partitions P into two equal sized halves (this is possible as we have an even number of points). If the antipodal points v and w are of the same color, then they are called *monochromatic antipodal* points (in short *m-antipodal* points) and if they have different colors then they are called *bichromatic antipodal* points (in short *b-antipodal* points).

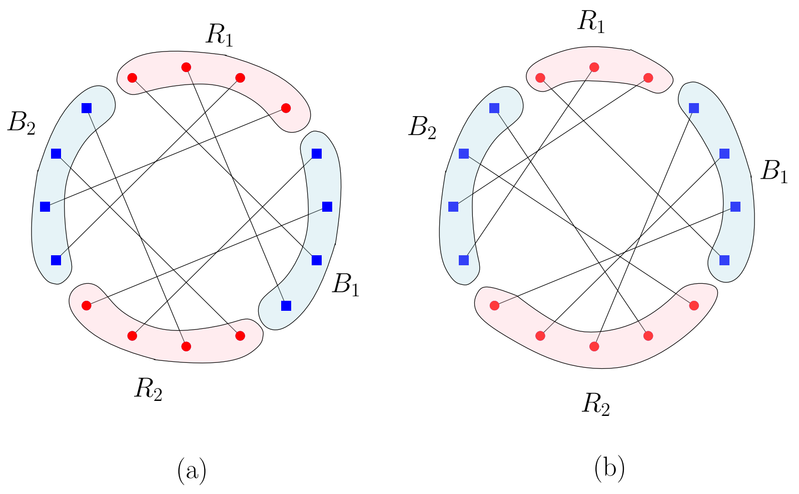


Figure 2 A balanced 4–block coloring (a) and a slightly unbalanced 4–block coloring (b) on 16 points and max-crossing matchings on them, each with 20 crossings.

▶ **Lemma 2.5.** *(⋆) Let $P \in \mathcal{C}_{n,n}$ and let $\mathrm{M}_P^\vee$ be a max-crossing matching of P. Then $\overline{\mathrm{cr}}(\mathrm{M}_P^\vee) \geq \frac{3n^2}{8} - O(n)$. Moreover, if $Q \in \mathcal{C}_{n,n}$ has a balanced 4–block coloring and $\mathrm{M}_Q^\vee$ is a max-crossing matching of Q, then $\mathrm{M}_Q^\vee$ is a min-max-crossing matching of the set $\mathcal{C}_{n,n}$.*

To prove Lemma 2.5, we make use of the following well-known theorem.

▶ **Theorem 2.6** (Ham sandwich theorem [10]). *For any bichromatic point set $P = R \cup B$ there exists a halfplane H such that $|R \cap H| = \lfloor \frac{|R|}{2} \rfloor$ and $|B \cap H| = \lfloor \frac{|B|}{2} \rfloor$.*

Proof sketch of Lemma 2.5. For a point set $P \in \mathcal{C}_{n,n}$, construct a new bichromatic point set with 4–block coloring using the following steps S1-S4.

S1: Remove all b-antipodal points from P, name the obtained bichromatic point set as S.

S2: Partition the set S into 4 groups as follows. First, partition S into two halves, say L and R, each having an equal number of red and blue points. Using the ham sandwich theorem partition one of the two halves such that each partition has an equal number of red and blue points. This partition can be duplicated on the other half as S consists only of m-antipodal points. Thus, we get 4 groups, each having an equal number of red and blue points. They are labeled as R_U, R_L, L_L, L_U w.r.t. clockwise order (see Figure 3).

S3: Add all the removed b-antipodal points back to S to get P and the partition of S induces a partition $R_{PU}, R_{PL}, L_{PL}, L_{PU}$ in P. Here the number of red (blue) points of R_{PU} and the number of blue (red) points of L_{PL} are equal. The same holds for R_{PL} and L_{PU}. Sort the points in R_{PU} and L_{PL} such that all the red points appear before the blue points w.r.t. the clockwise order. Then sort the points in R_{PL} and L_{PU} such that all the blue points appear before the red points. This gives a bichromatic point set K with the partition $R_{KU}, R_{KL}, L_{KL}, L_{KU}$. See Figure 4.

S4: Define the matchings M_P and M_K on P and K, respectively, as follows. For any pair $(X, Y) \in \{(R_{PU}, L_{PL}), (R_{PL}, L_{PU}), (R_{KU}, L_{KL}), (R_{KL}, L_{KU})\}$, the points in X are matched to points in Y such that any two of the matching edges emanating from the same colored points on X cross each other. Hence the matching edges of X give two crossing families, where the size of each family is determined by the number of points in X of each color. By our construction, the size of the crossing families is the same in both P and K. But in P, these crossing families cross each other and in K, these

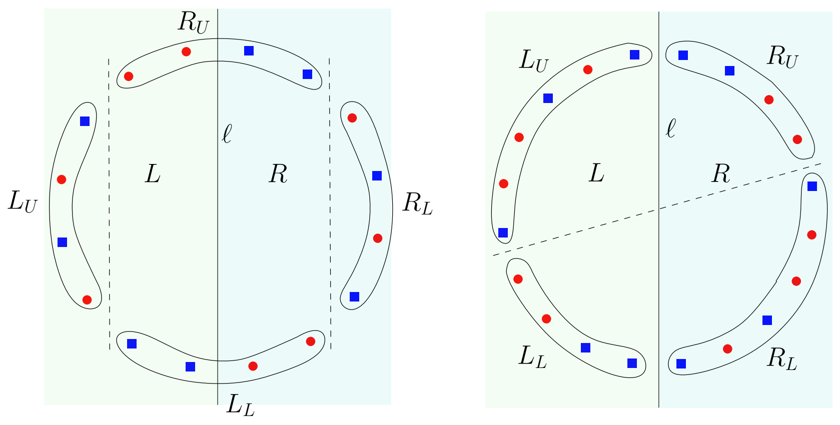


Figure 3 A bichromatic point set S with 16 points (left) and 20 points (right). In both cases, dotted lines represent a partition w.r.t. the ham sandwich theorem on the set R and L.

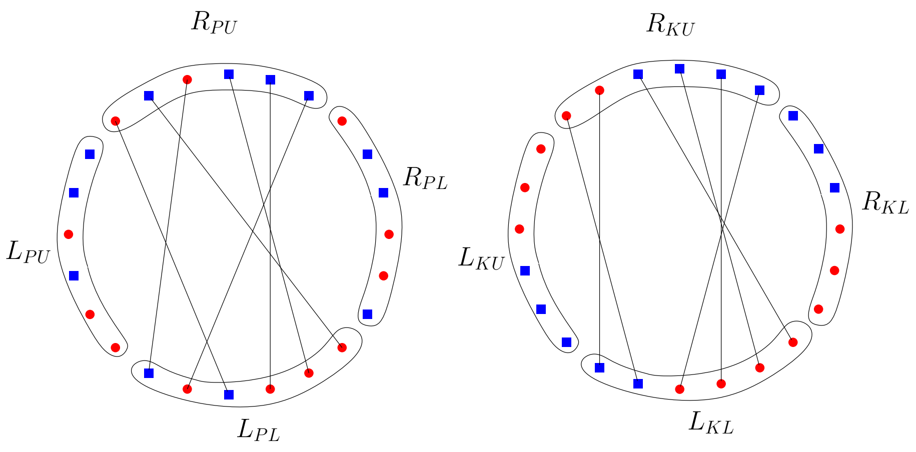


Figure 4 An example of bichromatic point set P (left) and the corresponding K (right) with 24 points. Also, the partial matching M_P and M_K.

crossing families do not cross each other. Hence, $\overline{\text{cr}}(M_P) \geq \overline{\text{cr}}(M_K)$. By construction, $K \in \mathcal{C}_{n,n}$ has a 4–block coloring. But it might not be a balanced 4–block coloring. Thus, by Lemma 2.4, $\overline{\text{cr}}(M_K) \geq \overline{\text{cr}}(\text{M}_Q^\vee)$. Also, the constructed matching M_P might not be a max-crossing matching of P. Hence, if $\text{M}_P^\vee$ is a max-crossing matching for P, then $\overline{\text{cr}}(\text{M}_P^\vee) \geq \overline{\text{cr}}(M_P)$. This implies $\overline{\text{cr}}(\text{M}_P^\vee) \geq \overline{\text{cr}}(\text{M}_Q^\vee)$, which completes the proof. ◀

As mentioned, Theorem 2.1 now follows directly from Lemma 2.4 and Lemma 2.5.

3 Conclusion

We showed that for any $k > \frac{3n^2}{8}$ there exists a bichromatic point set with n red and n blue points that does not admit any bichromatic perfect matching with k crossings. By straight-forward calculations, we can show that, for n even, bichromatic convex point sets containing n red and n blue points with alternating coloring cannot have a bichromatic perfect matching with one or two crossings. In ongoing work, we study the range $k \in [3, \frac{3n^2}{8}]$ for bichromatic convex point sets and also work on extending our results to bichromatic point sets in general (non-convex) position.

References

1 Oswin Aichholzer, Luis Barba, Thomas Hackl, Alexander Pilz, and Birgit Vogtenhuber. Linear transformation distance for bichromatic matchings. *Computational Geometry*, 68:77–88, 2018. doi:10.1016/j.comgeo.2017.05.003.

2 Oswin Aichholzer, Ruy Fabila-Monroy, Philipp Kindermann, Irene Parada, Rosna Paul, Daniel Perz, Patrick Schnider, and Birgit Vogtenhuber. Perfect matchings with crossings. In *Combinatorial Algorithms (IWOCA 2022)*, volume 13270 of *LNCS*, pages 46–59. Springer, 2022. doi:10.1007/978-3-031-06678-8_4.

3 Oswin Aichholzer, Ferran Hurtado, and Birgit Vogtenhuber. Compatible matchings for bichromatic plane straight-line graphs. *Proceedings of EuroCG'12*, pages 257–260, 2012. URL: https://www.eurocg.org/2012/booklet.pdf.

4 Greg Aloupis, Luis Barba, Stefan Langerman, and Diane L. Souvaine. Bichromatic compatible matchings. *Computational Geometry*, 48(8):622–633, 2015. doi:10.1016/j.comgeo.2014.08.009.

5 Andrei Asinowski, Tillmann Miltzow, and Günter Rote. Quasi-parallel segments and characterization of unique bichromatic matchings. *J. Comput. Geom.*, 6(1):185–219, 2015. doi:10.20382/jocg.v6i1a8.

6 Mikio Kano and Jorge Urrutia. Discrete geometry on colored point sets in the plane — a survey. *Graphs and Combinatorics*, 37(1):1–53, Jan 2021. doi:10.1007/s00373-020-02210-8.

7 Loren C Larson. *Problem-solving through problems.* Springer Science & Business Media, 2012.

8 Marko Savić and Miloš Stojaković. Structural properties of bichromatic non-crossing matchings. *Applied Mathematics and Computation*, 415:126695, 2022.

9 Micha Sharir and Emo Welzl. On the number of crossing-free matchings, cycles, and partitions. *SIAM Journal on Computing*, 36(3):695–720, 2006. doi:10.1137/050636036.

10 Csaba D. Tóth, Joseph O'Rourke, and Jacob E. Goodman. *Handbook of Discrete and Computational Geometry.* CRC press, third edition, 2017. doi:10.1201/9781315119601.

Axis-Parallel Right Angle Crossing Graphs

Patrizio Angelini[1], Michael A. Bekos[2], Julia Katheder[3], Michael Kaufmann[3], and Maximilian Pfister[3]

1 **Department of Mathematics, Natural, and Applied Sciences, John Cabot University, Rome, Italy.** pangelini@johncabot.edu
2 **Department of Mathematics, University of Ioannina, Ioannina, Greece.** bekos@uoi.gr
3 **Wilhelm-Schickard-Institut für Informatik, Universität Tübingen, Tübingen, Germany.** julia.katheder@uni-tuebingen.de, mk@informatik.uni-tuebingen.de, maximilian.pfister@uni-tuebingen.de

Abstract

In this work, we introduce RAC drawings of graphs in which each pair of crossing edge-segments are axis parallel. We study relationships with traditional RAC drawings and give edge-density bounds.

1 Introduction

RAC drawings of graphs were introduced a decade ago in [17] as an extension of planar drawings, and since then they have been a fruitful subject of intense research in Graph Drawing [3, 12, 14, 18, 19]. The motivation for their study primarily stems from cognitive experiments indicating that the negative effect of edge crossings in a drawing tends to be eliminated when the angles formed at the edge crossings are large [22]. In that aspect, RAC drawings form the optimal case of this scenario, in which all crossing angles occur at 90°.

The research on RAC drawings has mainly focused on two main directions depending on whether bends are allowed along the edges or not. Formally, a k-bend RAC graph is one admitting a k-bend RAC drawing, i.e., a drawing in which each edge is a polyline with at most k bends and the angle between any two crossing edge-segments is 90°. Concerning the edge density, a 0-bend RAC graph (or simply RAC graph) with n vertices has at most $4n - 10$ edges [17]. The corresponding edge-density bounds for 1- and 2-bend RAC graphs are $5.5n - 10$ [1] and $74.2n$ [6], respectively, while for $k \geq 3$ it is known that every graph is k-bend RAC [17]. The research on RAC graphs, however, is not limited to the study of edge-density bounds. Several algorithmic and combinatorial results [3, 4, 5, 13, 16, 19], as well as relationships with other graph classes [7, 9, 10, 11, 18] are known; for a survey see [15].

Our contribution. In this work, we introduce and study a natural subfamily of k-bend RAC graphs, called k-bend apRAC, which restricts all edge segments involved in crossings to be axis parallel. The motivation for this model is two-fold. First, by restricting the crossings to be axis parallel we expect to improve readability as, e.g., with orthogonal graph drawings [8, 20, 21]. Second, several algorithms from the literature about k-bend RAC graphs in fact yield k-bend apRAC drawings; see, e.g., [2, 3, 17]. So, two natural questions that arise are the following.

▶ Question 1. Do the classes of k-bend RAC and k-bend apRAC graphs coincide?

For $k \geq 3$, the answer to Question 1 is positive, as the construction given in [17], establishing that every graph is 3-bend RAC, can be converted to 3-bend apRAC by a rotation of 45°. For $k = 0$, we give a negative answer to Question 1 by determining 0-bend RAC graphs that are not 0-bend apRAC. For $k \in \{1, 2\}$, giving an answer to Question 1 is more challenging due to the degrees of freedom introduced by bends and we leave it as an open problem.

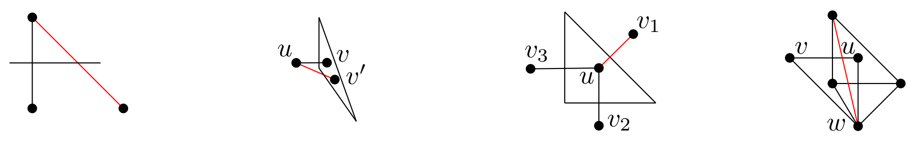


Figure 1 Forbidden configurations by Properties 1 to 4.

▶ Question 2. What is the edge density of k-bend apRAC graphs with n vertices?

For $k \geq 3$, the answer to Question 2 follows from the one of Question 1. For $k = 0$, the trivial upper-bound is $4n - 10$ [17]. However, we are able to provide corresponding lower-bound constructions with at most $4n - \Theta(\sqrt{n})$ edges. For $k = 1$, we prove that a 1-bend apRAC graph with n vertices has at most $5n - 6$ edges and provide a lower-bound construction with $5n - \Theta(\sqrt{n})$ edges. For $k = 2$, we give an upper bound of $10n - 12$ on the edge density of 2-bend apRAC graphs, which is significantly smaller than the trivial one of $74.2n$ derived from 2-bend RAC graphs [6]. Our lower-bound construction is a graph with n vertices and $10n - \Theta(\sqrt{n})$ edges. Notably, this bound extends to general 2-bend RAC graphs and improves the previous bound of $7.83n - \mathcal{O}(\sqrt{n})$ [6], answering an open question in [1].

2 Preliminaries

Properties 1 and 2 hold for 0-bend RAC (and thus for 0-bend apRAC) drawings.

▶ Property 1 ([17]). In a 0-bend RAC drawing no edge is crossed by two adjacent edges.

▶ Property 2 ([17]). A 0-bend RAC drawing does not contain a triangle T formed by edges of the graph and two edges (u, v) and (u, v'), such that u lies outside T and v, v' lie inside T.

The following two properties are limited to 0-bend apRAC drawings.

▶ Property 3. A 0-bend apRAC drawing does not contain a triangle T formed by edges of the graph and three vertices v_1, v_2, v_3 adjacent to a vertex u, such that v_1, v_2, v_3 lie outside T and u lies inside T.

Proof. Assuming the contrary, Property 1 implies that no two edges incident to u cross the same boundary edge of T. Hence, T consists of three axis-parallel edges; a contradiction. ◀

▶ Property 4. Let Γ be a 0-bend apRAC drawing containing a triangle T formed by edges of the graph and two adjacent vertices u and v such that u is contained inside T while v is outside T. Then, Γ cannot contain a vertex adjacent to u, v and all vertices of T.

Proof. For a contradiction, let w be a vertex adjacent to u, v and all vertices of T. If w is inside T in Γ, then (v, u) and (v, w) violate Property 2; a contradiction. Otherwise, since (u, v) and (u, w) cross T, by Property 1, it follows that T is a right-angle triangle, whose legs are axis-parallel. W.l.o.g., let (v_1, v_2) and (v_2, v_3) be the legs of T crossed by (u, v) and (u, w), respectively, such that (v_1, v_2) is horizontal and (v_2, v_3) is vertical. It follows that the edge (v_2, v_3) of T is crossed by (u, w) and (w, v_1) violating Property 1; a contradiction. ◀

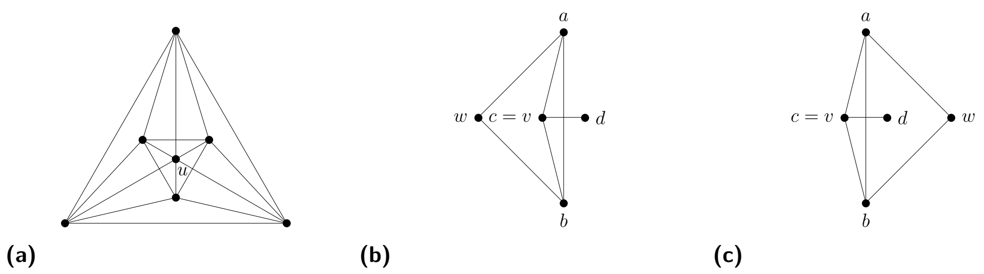


Figure 2 Illustrations for the proof of Theorem 3.2.

3 0-bend apRAC graphs

In this section, we present bounds on the edge density of 0-bend apRAC graphs and we further establish that these graphs form a proper subset of the 0-bend RAC graphs.

▶ **Theorem 3.1.** *A* 0*-bend apRAC graph with* n *vertices has at most* $4n - 10$ *edges. Also, there exist infinitely many* 0*-bend apRAC graphs with* n *vertices and* $4n - \Theta(\sqrt{n})$ *edges.*

Proof. The upper bound directly follows from [17]. The lower bound is obtained from an axis-parallel $\sqrt{n} \times \sqrt{n}$ drawing of an n-vertex grid, where each quadrangle contains a pair of crossing edges that can be made axis-parallel by a rotation of the grid by 45°. ◀

▶ **Theorem 3.2.** *There exist RAC graphs that are not* 0*-bend apRAC.*

Proof. Denote by G the graph shown in Fig. 2a, which is 0-bend RAC as witnessed in the figure. We next establish that G is not 0-bend apRAC, implying that any RAC graph containing it as a subgraph is not 0-bend apRAC. For a contradiction, assume that G admits a 0-bend apRAC drawing Γ. Denote by u the (sole) vertex of degree six in G and let G' be the subgraph of G without u, that is, $G' = G \setminus \{u\}$. Let Γ' be the subdrawing of Γ restricted to G'. If Γ' is plane, then u lies in a (triangular) face of Γ' since G' is a maximal planar graph. However, Property 3 applied to the internal faces and Property 2 applied to the external face of Γ' imply that u cannot lie in any face of Γ'; a contradiction.

Hence, Γ' necessarily contains at least one crossing, say between edges (a, b) and (c, d). Note that a, b, c and d are pairwise different, since Γ' is 0-bend. Any edge of G' belongs to exactly two cycles of length three. In particular, let (a, b) belong to two cycles abv and abw, while (c, d) belongs to cycles cdx and cdy. We first consider the case where v is contained inside the triangle representing abw - the case where w is contained inside the triangle abv is symmetric. By assumption, since (c, d) intersects (a, b) and since no edge can intersect two edges of a three cycle by Property 1, we have that one endpoint, say c, is either contained inside the triangle representing abv or it coincides with v; see Fig. 2b. In any case, c is contained inside the triangle representing abw, which is impossible by Property 4 since u is connected to all vertices of G'. We then consider the case where the triangles representing abv and abw do not contain a vertex of the other; see Fig. 2c. As above, c is either inside abv or coincides with v and d is either inside abw or coincides with w. However, $c = v$ and $d = w$ at the same time implies a K_4 in G', which is not possible. Hence, by Property 4 we derive a contradiction also in this case. ◀

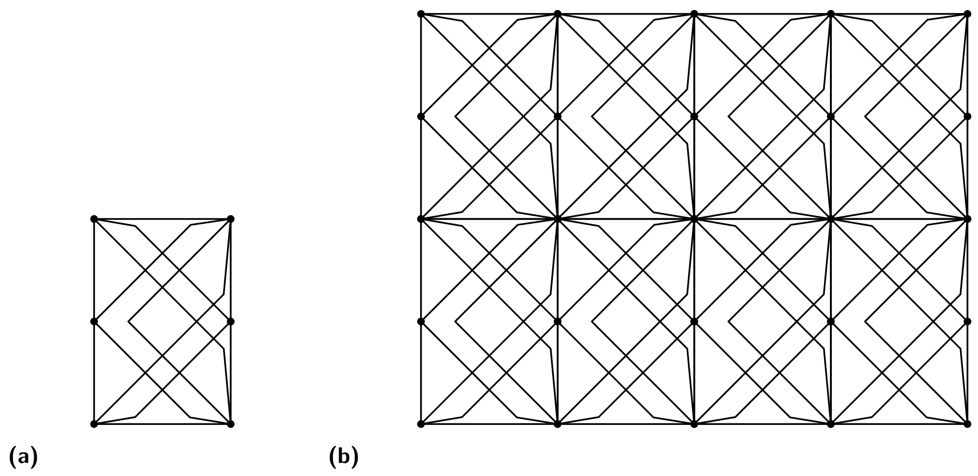


Figure 3 Lower bound construction for the class of 1-bend apRAC graphs. Here and in the following, the drawings have been rotated by 45° for layout reasons.

4 1-bend apRAC graphs

In this section, we will establish an upper bound and an almost matching lower bound for the class of 1-bend apRAC graphs.

▶ **Theorem 4.1.** *A 1-bend apRAC graph with n vertices has at most $5n - 6$ edges. Also, there exist infinitely many 1-bend apRAC graphs with n vertices and $5n - \Theta(\sqrt{n})$ edges.*

Proof. For the upper bound, consider a 1-bend apRAC drawing Γ of an n-vertex graph G. Each edge segment in Γ is either horizontal (h) or vertical (v) or oblique (o). For $x, y \in \{h, v, o\}$, let E_{xy} be the edges of G with two edge segments of type x and y. Then, E_{hv}, E_{ho}, E_{vo} and E_{oo} form a partition of the edge set of G, assuming that edges that consist of only one h-, v- or o-segment are counted towards E_{ho}, E_{vo} and E_{oo}, respectively. By construction, any crossing involves exactly one vertical and one horizontal segment. Hence, the subgraph of G induced by $E_{ho} \cup E_{oo}$ is planar and contains at most $3n - 6$ edges. Further, as every segment is incident to a vertex and since any vertex is incident to at most two vertical segments, we have $|E_{vo} \cup E_{hv}| \leq 2n$. Thus, $|E| = |E_{ho}| + |E_{vo}| + |E_{hv}| + |E_{oo}| \leq 5n - 6$.

For the lower bound, Fig. 3a depicts a so-called *tile* consisting of a 6-cycle with seven internal edges. Fig. 3b illustrates a tiling which consists of 4×2 tiles. Choose $h \geq 1$ and set $w = 2h$. The resulting $w \times h$ tiling contains $n = (2h + 1)^2$ vertices. Since the tiling consists of $2h^2$ many tiles and since any tile contributes at least ten edges in total (seven internal edges and six boundary edges, each shared by at most two tiles), it follows that the number of edges is at least $2h^2 \cdot 10 = 5n - \Theta(\sqrt{n})$ since $h = \frac{\sqrt{n}-1}{2}$. ◀

The current-best lower-bound construction for general 1-bend RAC graphs with n vertices has $5n - 10$ edges [1]. Therefore, by Theorem 4.1 it may be 1-bend apRAC as well. Towards an answer to Question 1 for $k = 1$, we deem important to state that it is far from clear to us whether and how this construction can be converted to be 1-bend apRAC.

5 2-bend apRAC graphs

In Theorems 5.1 and 5.2 we give upper and lower bounds on the edge density of 2-bend apRAC graphs, respectively.

▶ **Theorem 5.1.** *A 2-bend apRAC graph with n vertices G has at most $10n - 12$ edges.*

Proof. Consider a 2-bend apRAC drawing Γ of an n-vertex graph G. Each edge segment in Γ is either horizontal (h) or vertical (v) or oblique (o). Denote by S the set of edges that contain at least one segment in {h,v} incident to a vertex. Since any vertex is incident to at most two vertical and at most two horizontal segments, it follows that $|S| \leq 4n$. Let E_h, E_v and E_o be the set of edges of $E \backslash S$ whose middle part is h, v and o, respectively. Assuming that an edge of $E \backslash S$ consisting of less than three segments belongs to E_o, it follows that E_h, E_v and E_o form a partition of $E \setminus S$. Observe that the edges of E_o cannot be involved in any crossing in Γ, as all of its segments are oblique. Further, no two edges of E_h or E_v can cross. Hence, the subgraphs induced by $E_h \cup E_o$ and $E_v \cup E_o$ are planar and contain at most $3n-6$ edges each. Recall that $|S| \leq 4n$ and thus $|E| \leq |S| + |E_h| + |E_v| + 2|E_o| \leq 4n + 3n - 6 + 3n - 6 = 10n - 12$. ◀

▶ **Theorem 5.2.** *There exist infinitely many 2-bend apRAC graphs with n vertices and $10n - \Theta(\sqrt{n})$ edges.*

Proof. Our proof follows the general idea of the lower-bound construction of Theorem 4.1 using a different tile, namely, the one shown in Fig. 4. We construct an $h \times h$ tiling for $h \geq 3$; for space reason see Fig. 5 for an exemplary 2×2 tiling. Observe that besides the boundary edges of the tiles and the interior edges of the tiles, there exist edges that bridge two tiles. Fig. 6 outlines how the boundary, bridging and internal edges of a tile are connected.

Denote by $\mathcal{I}$ the set of tiles whose boundary edges do not bound the outer face. Consider an arbitrary tile $t \in \mathcal{I}$. Every boundary edge of t bounds at most two tiles in $\mathcal{I}$, hence for each of the 16 boundary edges, we assign t one half of it. Similarly, we assign to t one half of each of the 16 bridging edges that have an endpoint at a vertex of t and that are partially contained inside t. Since t contains 54 internal edges, we have that t contributes at least $8 + 8 + 54 = 70$ edges in the construction.

The tiling contains $n = 3h \cdot (h+1) + (h+1) \cdot (4h+1) = 7h^2 + 8h + 1$ vertices in total. Since $|\mathcal{I}| = (h^2 - (4h-4))$ and $h = \frac{1}{7} \cdot (\sqrt{7n+9} - 4)$ it follows that $m \geq 70 \cdot (h^2 - (4h-4)) \geq 70 \cdot (\frac{n}{7} - \frac{8}{49}\sqrt{7n+9} + \frac{25}{49}) - 70 \cdot (4 \cdot (\frac{1}{7} \cdot (\sqrt{7n+9} - 4)) - 4) = 10n - \Theta(\sqrt{n})$ as desired. ◀

6 Conclusion and Open Problems

We conclude with the following open problems:

- Are there k-bend RAC graphs that are not k-bend apRAC for $k \in \{1, 2\}$? For $k = 1$, we strongly believe that the construction in [1] is not 1-bend apRAC.
- Both the proofs for the upper bound as well as the lower-bound construction for the k-bend apRAC case with $k \in \{1, 2\}$ do not take the simplicity of the drawing into consideration, i.e., it is possible that two edges have more than one point in common (endpoint and crossing point). Do we get different bounds for simple drawings similar to the 1-bend RAC case [1]?
- For $k \in \{0, 1, 2\}$, narrow the gaps between the lower and the upper bounds. In particular, do the tight upper bounds contain subtractive lower-order terms of $\Theta(\sqrt{n})$?
- It is known that the recognition of 0-bend RAC graphs is NP-hard [5] (in fact, even for the fixed embedding setting it was shown to be $\exists\mathbb{R}$-hard [23]). What about the recognition of 0-bend apRAC graphs?

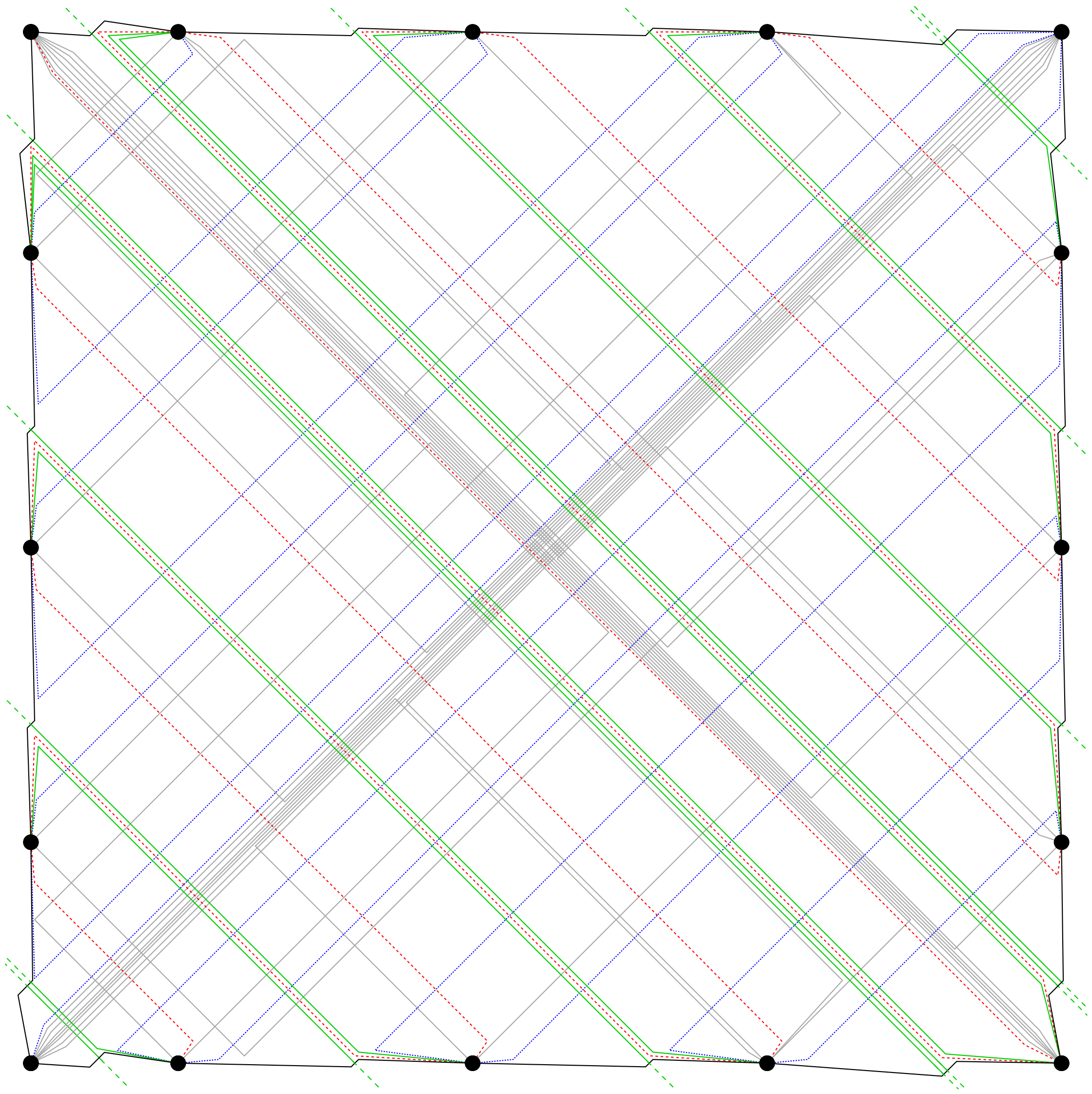


Figure 4 Tile for Theorem 5.2. The edges colored blue, red and gray are internal edges, the black edges are boundary edges and the green (half-) edges are bridging edges. Observe that the subgraph induced by the red and green edges is crossing-free, the same holds for the subgraph induced by the black and blue edges.

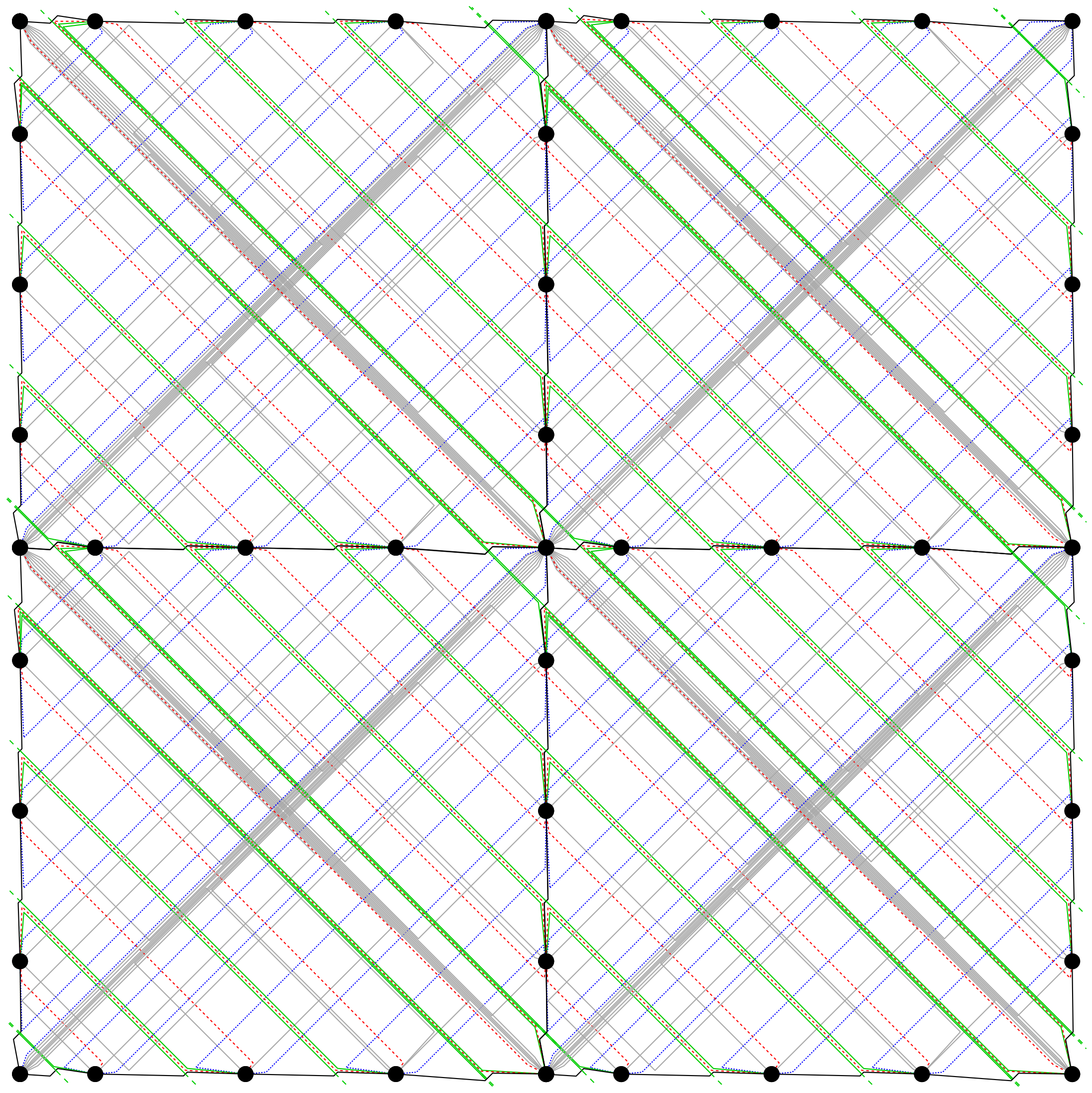


Figure 5 A 2×2 tiling using the tile shown in Fig. 4. Since the printed version might be cluttered, we refer the reader to the electronic version.

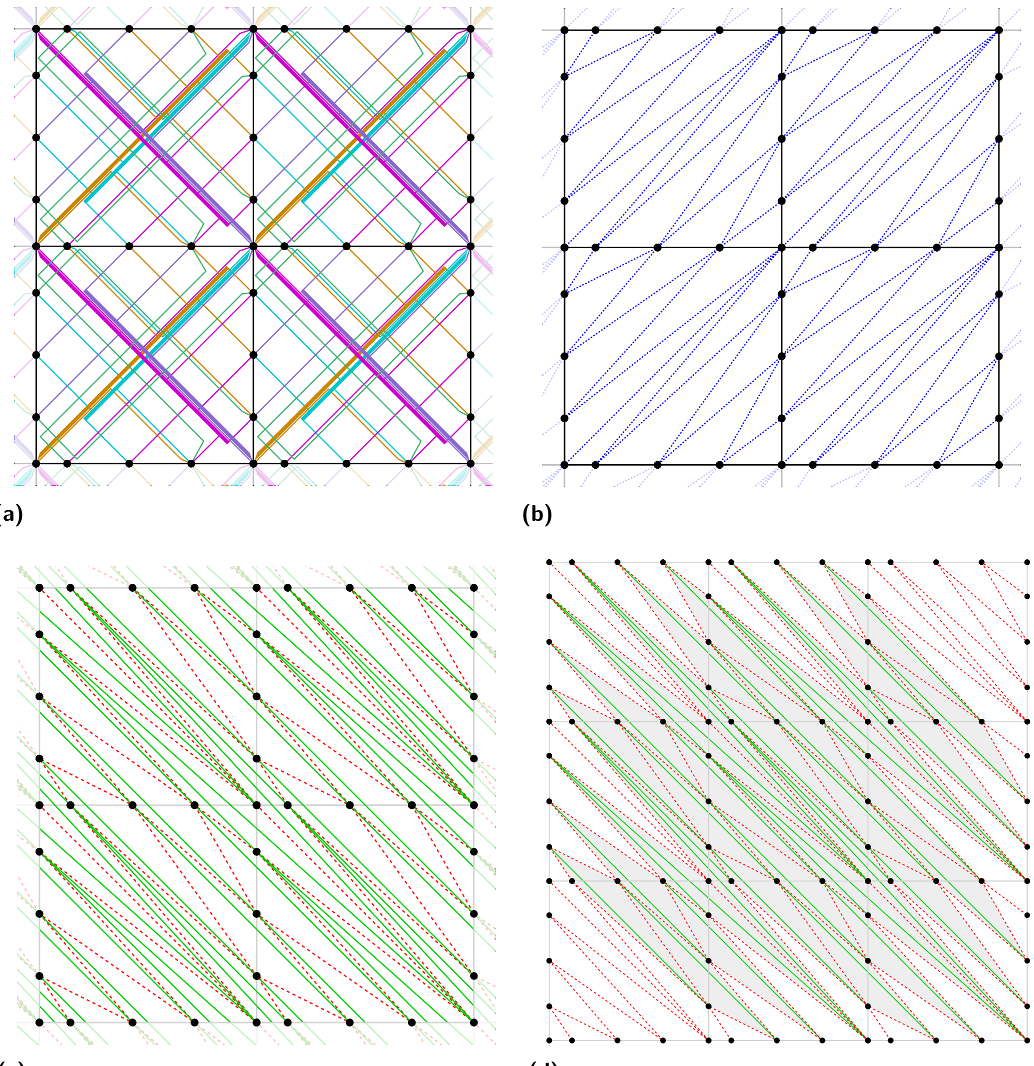


Figure 6 In (a), the edges which are drawn gray in Fig. 5 are drawn with colors and partially bundled to avoid clutter. (b) and (c) schematize the crossing-free subgraphs induced by the blue and black, and by the red and green edges, respectively. (d) depicts the same information as (c) in a larger 3×3 tiling.

References

1 Patrizio Angelini, Michael A. Bekos, Henry Förster, and Michael Kaufmann. On RAC drawings of graphs with one bend per edge. *Theor. Comput. Sci.*, 828-829:42–54, 2020. doi:10.1016/j.tcs.2020.04.018.

2 Patrizio Angelini, Michael A. Bekos, Julia Katheder, Michael Kaufmann, and Maximilian Pfister. RAC drawings of graphs with low degree. In Stefan Szeider, Robert Ganian, and Alexandra Silva, editors, *MFCS*, volume 241 of *LIPIcs*, pages 11:1–11:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPIcs.MFCS.2022.11.

3 Patrizio Angelini, Luca Cittadini, Walter Didimo, Fabrizio Frati, Giuseppe Di Battista, Michael Kaufmann, and Antonios Symvonis. On the perspectives opened by right angle crossing drawings. *J. Graph Algorithms Appl.*, 15(1):53–78, 2011. doi:10.7155/jgaa.00217.

4 Evmorfia N. Argyriou, Michael A. Bekos, Michael Kaufmann, and Antonios Symvonis. Geometric RAC simultaneous drawings of graphs. *J. Graph Algorithms Appl.*, 17(1):11–34, 2013. doi:10.7155/jgaa.00282.

5 Evmorfia N. Argyriou, Michael A. Bekos, and Antonios Symvonis. The straight-line RAC drawing problem is NP-hard. *J. Graph Algorithms Appl.*, 16(2):569–597, 2012. doi:10.7155/jgaa.00274.

6 Karin Arikushi, Radoslav Fulek, Balázs Keszegh, Filip Moric, and Csaba D. Tóth. Graphs that admit right angle crossing drawings. *Comput. Geom.*, 45(4):169–177, 2012. doi:10.1016/j.comgeo.2011.11.008.

7 Michael A. Bekos, Walter Didimo, Giuseppe Liotta, Saeed Mehrabi, and Fabrizio Montecchiani. On RAC drawings of 1-planar graphs. *Theor. Comput. Sci.*, 689:48–57, 2017. doi:10.1016/j.tcs.2017.05.039.

8 Therese C. Biedl and Goos Kant. A better heuristic for orthogonal graph drawings. *Comput. Geom.*, 9(3):159–180, 1998. doi:10.1016/S0925-7721(97)00026-6.

9 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.

10 Steven Chaplick, Fabian Lipp, Alexander Wolff, and Johannes Zink. Compact drawings of 1-planar graphs with right-angle crossings and few bends. *Comput. Geom.*, 84:50–68, 2019. doi:10.1016/j.comgeo.2019.07.006.

11 Hooman Reisi Dehkordi and Peter Eades. Every outer-1-plane graph has a right angle crossing drawing. *Int. J. Comput. Geom. Appl.*, 22(6):543–558, 2012. doi:10.1142/S021819591250015X.

12 Emilio Di Giacomo, Walter Didimo, Peter Eades, and Giuseppe Liotta. 2-layer right angle crossing drawings. *Algorithmica*, 68(4):954–997, 2014. doi:10.1007/s00453-012-9706-7.

13 Emilio Di Giacomo, Walter Didimo, Luca Grilli, Giuseppe Liotta, and Salvatore Agostino Romeo. Heuristics for the maximum 2-layer RAC subgraph problem. *Comput. J.*, 58(5):1085–1098, 2015. doi:10.1093/comjnl/bxu017.

14 Emilio Di Giacomo, Walter Didimo, Giuseppe Liotta, and Henk Meijer. Area, curve complexity, and crossing resolution of non-planar graph drawings. *Theory Comput. Syst.*, 49(3):565–575, 2011. doi:10.1007/s00224-010-9275-6.

15 Walter Didimo. Right angle crossing drawings of graphs. In Seok-Hee Hong and Takeshi Tokuyama, editors, *Beyond Planar Graphs*, pages 149–169. Springer, 2020. doi:10.1007/978-981-15-6533-5_9.

16 Walter Didimo, Peter Eades, and Giuseppe Liotta. A characterization of complete bipartite RAC graphs. *Inf. Process. Lett.*, 110(16):687–691, 2010. doi:10.1016/j.ipl.2010.05.023.

17 Walter Didimo, Peter Eades, and Giuseppe Liotta. Drawing graphs with right angle crossings. *Theor. Comput. Sci.*, 412(39):5156–5166, 2011. doi:10.1016/j.tcs.2011.05.025.

18 Peter Eades and Giuseppe Liotta. Right angle crossing graphs and 1-planarity. *Discret. Appl. Math.*, 161(7-8):961–969, 2013. doi:10.1016/j.dam.2012.11.019.

19 Henry Förster and Michael Kaufmann. On compact RAC drawings. In Fabrizio Grandoni, Grzegorz Herman, and Peter Sanders, editors, *ESA*, volume 173 of *LIPIcs*, pages 53:1–53:21. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPIcs.ESA.2020.53.

20 Ulrich Fößmeier and Michael Kaufmann. Drawing high degree graphs with low bend numbers. In Franz-Josef Brandenburg, editor, *GD*, volume 1027 of *LNCS*, pages 254–266. Springer, 1995. doi:10.1007/BFb0021809.

21 Ashim Garg and Roberto Tamassia. On the computational complexity of upward and rectilinear planarity testing. *SIAM J. Comput.*, 31(2):601–625, 2001. doi:10.1137/S0097539794277123.

22 Weidong Huang, Peter Eades, and Seok-Hee Hong. Larger crossing angles make graphs easier to read. *J. Vis. Lang. Comput.*, 25(4):452–465, 2014. doi:10.1016/j.jvlc.2014.03.001.

23 Marcus Schaefer. RAC-drawability is $\exists\mathbb{R}$-complete. In Helen C. Purchase and Ignaz Rutter, editors, *GD*, volume 12868 of *LNCS*, pages 72–86. Springer, 2021. doi:10.1007/978-3-030-92931-2_5.

k-planar Placement and Packing of Δ-regular Caterpillars[*]

Carla Binucci[1], Emilio Di Giacomo[1], Michael Kaufmann[2], Giuseppe Liotta[1], and Alessandra Tappini[1]

1 **Dipartimento di Ingegneria, Università degli Studi di Perugia, via G. Duranti 93, 06125, Perugia, Italy.**
`{carla.binucci, emilio.digiacomo, giuseppe.liotta, alessandra.tappini}@unipg.it`

2 **Wilhelm-Schickard Institut für Informatik, Universität Tübingen, Sand 13, 72076, Tübingen, Germany.**
`mk@informatik.uni-tuebingen.de`

Abstract

This paper studies a *packing* problem in the case that the host graph is k-planar, i.e., it can be drawn with at most k crossings per edge, and focuses on families of Δ-regular caterpillars, that are caterpillars whose non-leaf vertices have the same degree Δ. We study the dependency of k on the number h of caterpillars that are packed, both when these caterpillars are all isomorphic and when they are not. We give necessary and sufficient conditions for the packing of h isomorphic Δ-regular caterpillars and sufficient conditions for the packing of a set of h caterpillars such that each one is Δ-regular and the value of Δ can be different for different caterpillars.

1 Introduction

Graph *packing* is a classical problem in graph theory. It requires to merge several smaller graphs into a larger graph, called the *host graph*, without creating multiple edges. More precisely, let $G_1, G_2, \dots, G_h$ be h graphs, all having n vertices, an *h-packing* of $G_1, G_2, \dots, G_h$ is an n-vertex graph G that contains $G_1, G_2, \dots, G_h$ as edge-disjoint spanning subgraphs. We say that $G_1, G_2, \dots, G_h$ can be *packed into* G and that G is the *host graph* of $G_1, G_2, \dots, G_h$. If the h input graphs are all isomorphic, an h-packing is called an *h-placement* and we talk about *placement* problem in this case. We also say that $G_1, G_2, \dots, G_h$ can be *placed into* G. Many combinatorial problems can be regarded as packing/placement problems. For example, the Hamiltonian cycle problem for a graph G can be stated as the problem of packing an n-vertex cycle with the complement of G.

When no restriction is imposed on the host graph, we say that the host graph is K_n. Some classical results in this setting are those by Bollobás and Eldridge [3], Teo and Yap [20], Sauer and Spencer [19], while related famous conjectures are by Erdős and Sós from 1963 [6] and by Gyárfás from 1978 [12]. Within this line of research, Wang and Sauer [21], and Mahéo et al. [17] characterized triples of trees that admit a packing into K_n. Haler and Wang [13] extended this result to four copies of a tree. Further results in this area

[*] This work is partially supported by: *(i)* MIUR, grant 20174LF3T8 "AHeAD: efficient Algorithms for HArnessing networked Data"; *(ii)* Dipartimento di Ingegneria - Università degli Studi di Perugia, grants RICBA21LG: "Algoritmi, modelli e sistemi per la rappresentazione visuale di reti" and RICBA22CB "Modelli, algoritmi e sistemi per la visualizzazione e l'analisi di grafi e reti". The research in this paper started during the *Summer Workshop on Graph Drawing 2021, Castiglione del Lago (PG).*

are by Hedetniemi et al. [14], Wozniak and Wojda [22] and Aichholzer et al. [1]. Also refer to [13, 21, 23] for results about the placement problem.

A tighter relation to graph drawing was established when researchers did not consider K_n to be the host graph, but required the host graph to be planar. A central question here is how to pack two trees of size n into a planar graph of size n. After a long series of intermediate steps [7, 8, 9, 10, 18], enlarging the class of trees that could be packed, Geyer et al. [11] showed that any two non-star trees can be embedded into a planar graph.

Relaxing the planarity condition allows for packing of more (than two) trees, and restricting the number of crossings for each edge, i.e., in the so-called beyond planar setting [5, 15, 16], still keeps the host graph sparse. The study of the packing problem in the beyond planarity setting was started by De Luca et al. [4], who consider how to pack caterpillars, paths, and cycles into 1-planar graphs. While two trees can always be packed into a planar graph, it may not be possible to pack three trees into a 1-planar graph.

In this work, we further generalize the problem by allowing the host graph to be k-planar for any $k \geq 1$, and we study the dependency of k on the number of caterpillars to be packed and on their vertex degree. We consider Δ-*regular* caterpillars, which are caterpillars whose non-leaf vertices all have the same degree Δ. Our results can be briefly outlined as follows.

- We characterize those families of h Δ-regular caterpillars which admit a placement into a k-planar graph and show that $k \in O(\Delta h + h^2)$.
- We consider the packing problem of h caterpillars, $C_1, C_2, \dots, C_h$, such that C_i is Δ_i-regular, for $i \in \{1, 2, \dots, h\}$, and $\Delta_i \geq \Delta_{i+1}$, for $i \in \{1, 2, \dots, h-1\}$. By extending the techniques of the bullet above, we give sufficient conditions for the existence of a k-planar packing of these caterpillars and show that $k \in O(\Delta_1 h^2)$.

For reasons of space, some proofs are omitted and can be found in the full version, which also contains additional results [2].

2 Preliminaries

Given a graph G, we denote by $\deg_G(v)$ the degree of a vertex v in G. Let $G_1, G_2, \dots, G_h$ be h graphs, all having n vertices. The following property holds.

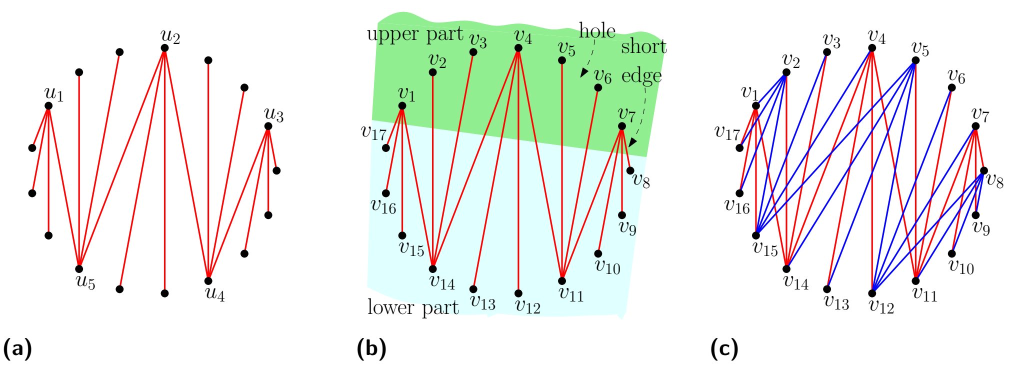
▶ Property 1. A packing of h connected n-vertex graphs exists only if $n \geq 2h$ and $\deg_{G_i}(v) \leq n - h$, for each $i \in \{1, 2, \dots, h\}$ and for each vertex v.

A *k-planar graph* is a graph that admits a drawing in the plane such that each edge is crossed at most k times. If the host graph of an h-packing (h-placement) is k-planar, we will talk about a *k-planar h-packing* (*k-planar h-placement*) or simply about a k-planar packing/placement, when the value of h is not relevant.

A *caterpillar* is a tree T such that removing all leaves we are left with a path, called *spine*; T is Δ-*regular*, for $\Delta \geq 2$, if $\deg_T(v) = \Delta$ for every spine vertex v. The number of vertices of a Δ-regular caterpillar is $n = \sigma(\Delta - 1) + 2$, where σ is the number of vertices of the spine. While Δ-regular caterpillars are defined for any value of $\sigma \geq 1$, when we want to pack a set of $h \geq 2$ caterpillars, Property 1 requires that each caterpillar has $\sigma \geq 2$. Otherwise, the unique spine vertex would have degree $n - 1$ and Property 1 would not hold.

3 h-placement of Δ-regular Caterpillars into k-planar Graphs

We start by showing that Property 1 is, in general, not sufficient to guarantee a placement even for Δ-regular caterpillars.



u_1
u_2
u_3
u_4
u_5
(a)
upper part
hole
short edge
lower part
v_1 v_2 v_3 v_4 v_5 v_6 v_7 v_8 v_9 v_{10} v_{11} v_{12} v_{13} v_{14} v_{15} v_{16} v_{17}
(b)
v_1 v_2 v_3 v_4 v_5 v_6 v_7 v_8 v_9 v_{10} v_{11} v_{12} v_{13} v_{14} v_{15} v_{16} v_{17}
(c)

Figure 1 (a) A zig-zag drawing of a 4-regular caterpillar; (b) the upper and the lower part are highlighted; (c) a 2-packing obtained by the drawing of (b) with a copy of it rotated by one step.

▶ **Theorem 1.** *For every $h \geq 2$, let Δ be a positive integer such that $\frac{h-1}{\Delta-1}$ is not integer. A set of h Δ-regular caterpillars with $n = 2h$ vertices cannot be placed into any graph.*

In order to establish necessary and sufficient conditions for the existence of a k-planar h-placement of h isomorphic Δ-regular caterpillars, we now describe a construction of a type of drawings called *zig-zag drawings* and study their properties.

Let C be a Δ-regular caterpillar with n vertices; we construct a drawing Γ of C as shown in Figure 1. The number of vertices of the spine of C is $\sigma = \frac{n-2}{\Delta-1}$; consider a set of σ points on a circle γ and denote by $u_1, u_2, \dots, u_\sigma$ these points according to the circular clockwise order they appear along γ. Draw the spine of C by connecting, for $i = 1, 2, \dots, \lfloor \frac{\sigma}{2} \rfloor$, the points u_i and u_{i+1} to the point $u_{\sigma-i+1}$; see Figure 1a. If σ is even and $i = \frac{\sigma}{2}$, the points u_{i+1} and $u_{\sigma-i+1}$ coincide and therefore the point $u_{\frac{\sigma}{2}}$ is connected only to $u_{\frac{\sigma}{2}+1}$. Notice that all points u_i have two incident edges, except u_1 and $u_{\lfloor \frac{\sigma}{2} \rfloor+1}$ which have only one. We add the leaves adjacent to each vertex $u_i \notin \{u_1, u_{\lfloor \frac{\sigma}{2} \rfloor+1}\}$ by connecting $u_{\sigma-i+1}$ to $\Delta - 2$ points between u_i and u_{i+1}; we then add the leaves adjacent to u_1 by connecting it to $\Delta - 1$ points between u_σ and u_1; we finally add the leaves adjacent to $u_{\lfloor \frac{\sigma}{2} \rfloor+1}$ by connecting it to $\Delta - 1$ points between $u_{\frac{\sigma}{2}}$ and $u_{\frac{\sigma}{2}+1}$ if σ is even, or to $\Delta - 1$ points between $u_{\lfloor \frac{\sigma}{2} \rfloor+1}$ and $u_{\lfloor \frac{\sigma}{2} \rfloor+2}$ if σ is odd. The resulting drawing is called a *zig-zag drawing* of C.

From now on, we assume that in a zig-zag drawing the points that represent vertices are equally spaced on the circle γ. Let χ be the convex hull of the points representing the vertices of C in Γ. A zig-zag drawing has exactly two sides of χ that coincide with two edges of C; we call these two edges *short edges* of Γ. Denote by $v_1, v_2, \dots, v_n$ the vertices of Γ according to the circular clockwise order they appear along χ with $v_1 \equiv u_1$; see Figure 1b. Notice that (v_1, v_n) is a short edge and v_n is its leaf vertex.

Consider a straight line s that intersects both short edges of Γ; line s intersects all the edges of the zig-zag drawing. Without loss of generality, assume that s is horizontal and denote by U the set of vertices that are above s and by L the set of vertices that are below s. The vertices in U form the *upper part* of Γ and those in L form the *lower part* of Γ. Without loss of generality, assume that v_1 is in the upper part (and therefore v_n is in the lower part). It follows that each edge has the end-vertex with lower index in the upper part, and the end-vertex with higher index in the lower part. Hence the short edge different from (v_1, v_n), which we denote as (v_{r-1}, v_r), is such that v_{r-1} is in the upper part and v_r is in the lower part. The first vertex of the upper part, i.e., vertex v_1, is called *starting point* of Γ, while the first vertex of the lower part, i.e., vertex v_r, is called *ending point* of Γ. We observe that

$r = 1 + \frac{n}{2}$ if the number of vertices of the spine $\sigma = \frac{n-2}{\Delta-1}$ is even, while $r = 1 + \frac{n-(\Delta-1)}{2}$ if σ is odd. This can be written with a single formula as $r = 1 + \frac{n-(\Delta-1)(\sigma \mod 2)}{2}$.

Let ℓ be a positive integer and let Γ' be the drawing obtained by re-mapping vertex v_i to the point[1] representing $v_{i+\ell}$ in Γ. We say that Γ' is the drawing obtained by *rotating* Γ *by ℓ steps.* Note that the starting point of Γ' is v_j with $j = 1 + \ell$ and the ending point is v_r with $r = 1 + \ell + \frac{n-(\Delta-1)(\sigma \mod 2)}{2} = j + \frac{n-(\Delta-1)(\sigma \mod 2)}{2}$. The drawing in Figure 1c is the union of two zig-zag drawings Γ_1 and Γ_2, where Γ_2 is obtained by rotating Γ_1 by one step; Γ_1 has starting point v_1 and ending point v_8; Γ_2 has starting point v_2 and ending point v_9.

▶ **Lemma 1.** *Let Γ_1 be a zig-zag drawing of a Δ-regular caterpillar C with starting point j_1; let Γ_2 be a zig-zag drawing of C with starting point j_2. If $0 < j_2 - j_1 < \frac{n-(\Delta-1)(\sigma \mod 2)}{2}$, where σ is the number of spine vertices of C, then $\Gamma_1 \cup \Gamma_2$ has no multiple edges.*

The next lemma computes the maximum number of crossings per edge in the union of two zig-zag drawings without overlapping edges. We state the lemma assuming that the two Δ-regular caterpillars can have different vertex degrees, as we are going to use the lemma to establish upper bounds on k both for k-planar h-placements and for k-planar h-packings.

▶ **Lemma 2.** *Let C_1 be an n-vertex Δ_1-regular caterpillar and let C_2 be an n-vertex Δ_2-regular caterpillar with $\Delta_i \leq n-2$ (for $i = 1,2$). Let Γ_1 be a zig-zag drawing of C_1 with starting point v_{j_1} and let Γ_2 be a zig-zag drawing of C_2 with starting point v_{j_2} with $0 < j_2 - j_1 < \frac{n}{2}$. If $\Gamma_1 \cup \Gamma_2$ has no multiple edges, then any edge of $\Gamma_1 \cup \Gamma_2$ is crossed at most $2(\Delta_1 + \Delta_2) + 4(j_2 - j_1)$ times.*

We are now ready to characterize the Δ-regular caterpillars that admit an h-placement.

▶ **Theorem 2.** *Let $C_1, C_2, \ldots, C_h$ be h isomorphic Δ-regular caterpillars with n vertices. An h-placement of $C_1, C_2, \ldots, C_h$ exists if and only if: (i) $\Delta \leq n-h$; and (ii) $n \geq 2h+(\Delta-1)\cdot(\sigma \mod 2)$, where σ is the number of spine vertices of each C_i ($i = 1, 2, \ldots, h$). Further, if an h-placement exists, there exists one that is k-planar for $k \in O(\Delta h + h^2)$.*

Proof. We first prove the sufficient condition. Let $C_1, C_2, \ldots, C_h$ be the h caterpillars and assume that $n \geq 2h + (\Delta-1)(\sigma \mod 2)$. We compute an h-placement of $C_1, C_2, \ldots, C_h$ starting from a zig-zag drawing Γ_1 of C_1 and obtaining the drawing Γ_i of C_i by rotating Γ_1 by $i-1$ steps, for $i = 2, 3, \ldots, h$. Since $h \leq \frac{n-(\Delta-1)(\sigma \mod 2)}{2}$ each Γ_i is rotated by less than $\frac{n-(\Delta-1)(\sigma \mod 2)}{2}$ steps and each pair of drawings Γ_i and Γ_j satisfies Lemma 1. Thus, there are no multiple edges and the union of all Γ_i's is a valid h-placement of $C_1, C_2, \ldots, C_h$.

We now prove the necessary condition. If σ is even, then conditions (i) and (ii) are necessary by Property 1. Hence, consider the case when σ is odd. Condition (i) is necessary by Property 1. Assume, by contradiction, that condition (ii) is not necessary, i.e., there exists a placement of h caterpillars $C_1, C_2, \ldots, C_h$ such that $n < 2h + (\Delta-1)$. Since $C_1, C_2, \ldots, C_h$ admit an h-placement, by Property 1 n must be at least $2h$. Thus, it would be $2h \leq n < 2h + (\Delta-1)$; in other words, $n = 2h + \alpha$ with $0 \leq \alpha \leq \Delta-2$.

Let G be the host graph of the h-placement and let v be a vertex of G to which the largest number of spine vertices of $C_1, C_2, \ldots, C_h$ is mapped. Let β be the number of spine vertices that are mapped to v. There are other $h-\beta$ leaf vertices that are mapped to v (because one vertex per caterpillar has to be mapped to each vertex of G). The degree of

[1] In a drawing with the vertices in convex position the indices of the vertices are handled in a circular fashion, i.e., the element with index n is followed by the element with index 1.

v in G is at most $n-1$ and each of the spine vertices mapped to v has degree Δ. Hence, the β spine vertices mapped to v have degree $\beta\Delta$ in total. Vertex v can have at most $n-1-\beta\Delta$ other edges and therefore it must be $n-1-\beta\Delta \geq h-\beta$, i.e., $\beta \leq \frac{n-1-h}{\Delta-1}$. On the other hand, there are σh spine vertices in total and, since G has n vertices, there are at least $\lceil \frac{\sigma h}{n} \rceil$ spine vertices mapped to v, i.e., $\beta \geq \lceil \frac{\sigma h}{n} \rceil$. Putting together the conditions on β, we obtain: $\lceil \frac{\sigma h}{n} \rceil \leq \beta \leq \frac{n-1-h}{\Delta-1}$. Since $h = \frac{n-\alpha}{2}$ and $n = \sigma(\Delta-1)+2$, we obtain: $\left\lceil \frac{\sigma}{2} - \frac{\sigma\alpha}{2(\sigma(\Delta-1)+2)} \right\rceil \leq \beta \leq \frac{\sigma(\Delta-1)+\alpha}{2(\Delta-1)}$, which implies:

$$\left\lceil \frac{\sigma}{2} - \frac{\alpha}{2(\Delta-1)+\frac{4}{\sigma}} \right\rceil \leq \frac{\sigma}{2} + \frac{\alpha}{2(\Delta-1)}. \tag{1}$$

We prove that Eq. (1) does not hold. Since σ is odd, it is $\sigma = 2i+1$ for $i \in \mathbb{N}$, and thus:

$$\left\lceil i + \frac{1}{2} - \zeta \right\rceil \leq i + \frac{1}{2} + \zeta', \tag{2}$$

with $\zeta = \frac{\alpha}{2(\Delta-1)+\frac{4}{\sigma}}$ and $\zeta' = \frac{\alpha}{2(\Delta-1)}$. We have $\zeta < \zeta'$ and $\zeta' = \frac{\alpha}{2(\Delta-1)} \leq \frac{\Delta-2}{2(\Delta-1)} < \frac{\Delta-1}{2(\Delta-1)} = \frac{1}{2}$. The first term of Eq. (2) is $i+1$ because $0 < \frac{1}{2} - \zeta < 1$; the second term is less than $i+1$ because $0 < \frac{1}{2} + \zeta' < 1$. It follows that Eq. (2), and hence Eq. (1), does not hold.

We now prove the bound on the number of crossings along an edge. Consider an edge e of Γ_1; the number of crossings along an edge of the drawing of another caterpillar is bounded by the same number. By Lemma 2, the number of crossings χ_e along e due to the edges of another drawing Γ_l (with $2 \leq l \leq h$) is at most $2(\Delta_1 + \Delta_l) + 4(j_l - j_1)$. Summing over all drawings distinct from Γ_1, we obtain $\chi_e \leq \sum_{l=2}^{h}(2(\Delta_1+\Delta_l)+4(j_l-j_1))$. Considering that $\Delta_l = \Delta$ for every l and that $j_l - j_1 = l-1$, we have $\chi_e \leq \sum_{l=2}^{h}(4\Delta + 4(l-1)) \leq (4\Delta-2)h + 2h^2 - 4\Delta$. ◀

Theorem 2 considers h copies of the same caterpillar that are packed into a host graph. By exploiting the zig-zag drawing technique, we can establish sufficient conditions for packing Δ_1-, Δ_2-, ..., Δ_h-regular caterpillars with $\Delta_i \neq \Delta_j$, $1 \leq i,j \leq h$.

▶ **Theorem 3.** *Let $C_1, C_2, \ldots, C_h$ be h caterpillars such that C_i is Δ_i-regular, for $1 \leq i \leq h$, and $\Delta_h \leq \Delta_{h-1} \leq \cdots \leq \Delta_1 \leq n-h$. If $\sum_{i=1}^{h} \Delta_i \leq n-1$ and $\sum_{i=2}^{h} \lceil \frac{\Delta_i}{2} \rceil < \frac{n-(\Delta_1-1)}{2}$, then there exists a k-planar packing with $k \in O(\Delta_1 h^2)$.*

4 Open Problems

We conclude with some open problems related to our results. (i) Extend the characterization of Theorem 2 to the placement of caterpillars that are not Δ-regular; (ii) Theorem 3 gives sufficient conditions for the k-planar packing of a family of caterpillars. It would be interesting to give a complete characterization of the packability of this family into k-planar graphs.

References

1 Oswin Aichholzer, Thomas Hackl, Matias Korman, Marc van Kreveld, Maarten Löffler, Alexander Pilz, Bettina Speckmann, and Emo Welzl. Packing plane spanning trees and paths in complete geometric graphs. *Information Processing Letters*, 124:35 – 41, 2017.

2 Carla Binucci, Emilio Di Giacomo, Giuseppe Liotta, Michael Kaufmann, and Alessandra Tappini. k-planar placement and packing of Δ-regular caterpillars, 2023. doi:10.48550/arXiv.2301.01226.

3 Béla Bollobás and Stephen E. Eldridge. Packings of graphs and applications to computational complexity. *J. Comb. Theory, Ser. B*, 25(2):105–124, 1978.

4 Felice De Luca, Emilio Di Giacomo, Seok-Hee Hong, Stephen G. Kobourov, William Lenhart, Giuseppe Liotta, Henk Meijer, Alessandra Tappini, and Stephen K. Wismath. Packing trees into 1-planar graphs. *J. Graph Algorithms Appl.*, 25(2):605–624, 2021.

5 Walter Didimo, Giuseppe Liotta, and Fabrizio Montecchiani. A survey on graph drawing beyond planarity. *ACM Comput. Surv.*, 52(1):4:1–4:37, 2019.

6 P. Erdős. Extremal problems in graph theory. In *Theory of Graphs and its Applications, Proc. Sympos. Smolenice*, pages 29–36, 1964.

7 Fabrizio Frati. Planar packing of diameter-four trees. In *Proceedings of the 21st Annual Canadian Conference on Computational Geometry*, pages 95–98, 2009.

8 Fabrizio Frati, Markus Geyer, and Michael Kaufmann. Planar packing of trees and spider trees. *Inf. Process. Lett.*, 109(6):301–307, 2009.

9 Alfredo García Olaverri, M. Carmen Hernando, Ferran Hurtado, Marc Noy, and Javier Tejel. Packing trees into planar graphs. *Journal of Graph Theory*, 40(3):172–181, 2002.

10 Markus Geyer, Michael Hoffmann, Michael Kaufmann, Vincent Kusters, and Csaba D. Tóth. Planar packing of binary trees. In *WADS 2013*, volume 8037 of *LNCS*, pages 353–364. Springer, 2013.

11 Markus Geyer, Michael Hoffmann, Michael Kaufmann, Vincent Kusters, and Csaba D. Tóth. The planar tree packing theorem. *JoCG*, 8(2):109–177, 2017.

12 András Gyárfás and Jenő Lehel. Packing trees of different order into K_n. In *Combinatorics (Proc. Fifth Hungarian Colloq., Keszthely, 1976)*, volume 1, pages 463–469. North-Holland New York, 1978.

13 Sean P. Haler and Hong Wang. Packing four copies of a tree into a complete graph. *Australas. J Comb.*, 59:323–332, 2014. URL: `http://ajc.maths.uq.edu.au/pdf/59/ajc_v59_p323.pdf`.

14 S.M. Hedetniemi, Stephen Hedetniemi, and P.J. Slater. A note on packing two trees into K_n. *Ars Combinatoria*, 11, 01 1981.

15 Seok-Hee Hong and Takeshi Tokuyama, editors. *Beyond Planar Graphs*. Springer, 2020. `doi:10.1007/978-981-15-6533-5`.

16 Stephen G. Kobourov, Giuseppe Liotta, and Fabrizio Montecchiani. An annotated bibliography on 1-planarity. *Computer Science Review*, 25:49 – 67, 2017.

17 Maryvonne Mahéo, Jean-François Saclé, and Mariusz Wozniak. Edge-disjoint placement of three trees. *Eur. J. Comb.*, 17(6):543–563, 1996.

18 Yoshiaki Oda and Katsuhiro Ota. Tight planar packings of two trees. In *22nd European Workshop on Computational Geometry*, 2006.

19 Norbert Sauer and Joel Spencer. Edge disjoint placement of graphs. *J. Comb. Theory, Ser. B*, 25(3):295–302, 1978.

20 S. K. Teo and H. P. Yap. Packing two graphs of order n having total size at most $2n-2$. *Graphs Comb.*, 6(2):197–205, 1990.

21 Hong Wang and Norbert Sauer. Packing three copies of a tree into a complete graph. *European Journal of Combinatorics*, 14(2):137 – 142, 1993.

22 Mariusz Wozniak and A. Pawel Wojda. Triple placement of graphs. *Graphs and Combinatorics*, 9(1):85–91, 1993.

23 Andrzej Zak. A note on k-placeable graphs. *Discret. Math.*, 311(22):2634–2636, 2011. `doi:10.1016/j.disc.2011.08.002`.

Clustering with Obstacles

Mark de Berg[1], Leyla Biabani[2], Morteza Monemizadeh[3], and Leonidas Theocharous[4]

1 Department of Mathematics and Computer Science, TU Eindhoven
M.T.d.Berg@tue.nl
2 Department of Mathematics and Computer Science, TU Eindhoven
l.biabani@tue.nl
3 Department of Mathematics and Computer Science, TU Eindhoven
M.Monemizadeh@tue.nl
4 Department of Mathematics and Computer Science, TU Eindhoven
l.theocharous@tue.nl

Abstract

We study the discrete k-median problem on a set of n points in a polygonal domain P with m vertices. Here the goal is to find a set $S \subset D$ of k centers such that $\sum_{d \in D}\{\min_{s\in S}\{\|\pi(d,s)\|\}\}$ is minimized, where $\|\pi(d,s)\|$ denotes the length of the shortest path between d and s. We develop an exact algorithm for this problem, which runs in time $poly(n,m) + n^{O(\sqrt{k})}$. Subsequently, we show that our approach can also be applied to solve the discrete k-center problem with z outliers in the same running time.

1 Introduction

Problem statement. Let P be a polygonal domain (that is, a polygon with holes) with m vertices, and let D be a set of n *demand points* in P. The *discrete k-median problem* asks to find a set $S \subset D$ of k *center points* such that the quantity $\sum_{d \in D}\{\min_{s\in S}\{\|\pi(d,s)\|\}\}$ is minimized, where $\|\pi(p,q)\|$ denotes the length of the shortest path $\pi(p,q)$ between two points p, q in P. The *discrete k-center problem* asks to find a set $S \subset D$ of k center points such that the maximum distance of points in D to their nearest center in S is minimized. An outlier can significantly increase the maximum distance to the nearest center, so we study the *k-center problem with z outliers*, which asks to minimize the maximum distance of all but z points of D to their nearest center in S.

Our interest lies in developing a subexponential exact algorithm for these problems, which depends exponentially on k and not on the complexity of the polygonal domain. We first show how to do this for k-median and as a nice by-product, we show that our approach can also be applied to the k-center problem with z outliers.

Previous work. In the Euclidean setting, exact algorithms for k-median and k-center have been known for a long time. Most relevant to our paper is the work by Hwang *et al.* [4], who presented algorithms with running time $n^{O(\sqrt{k})}$ for the planar version of the problems. Their approach works with the Voronoi diagram of the (unknown) optimal solution, and "guesses" a cycle separator of its dual graph. The separator splits the problem into two subproblems, which are then solved recursively. This is an idea that we also make use of. The same approach was employed more recently by Marx and Pilipczuk [5] to solve a wide range of covering and packing problems defined on planar graphs. This includes k-center, which they solve in $n^{O(\sqrt{k})}$. To the best of our understanding, their approach cannot be used to tackle k-median and also cannot directly handle outliers. For small values of k and in the

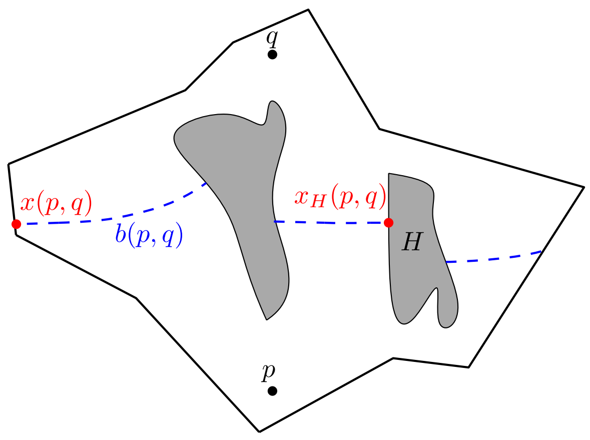


Figure 1 Illustration for the definition of $b(p,q), x(p,q)$ and $x_H(p,q)$.

setting of a simple polygon, only the 1-center and 2-center problems have received attention. More specifically, Ahn *et al.* [1] studied the problem of computing the *geodesic center* of a (weakly) simple polygon P, where the task is to find the point $s \in P$ which minimizes the maximum geodesic distance from *any* other point in P. They developed a linear-time algorithm for this problem. Their algorithm can be used to compute the 1-center of a set D of n points in P: as observed in [2], the 1-center of D coincides with the 1-center of the relative convex hull of D (denoted by $\textsc{rch}(D)$). Since the geodesic convex hull is a weakly simple polygon that can be computed in time $O(n \log n + m)$ (where m is the complexity of P), the 1-center of D can be computed in the same time. For $k = 2$, Oh *et al.* [7] observed that the 2-center of $\textsc{rch}(D)$ does not anymore necessarily coincide with the 2-center of D and went on to present an $O(m(n+m)\log^3(n+m))$-time algorithm for the 2-center of D. When holes are also allowed, we are not aware of any work that specifically addresses the 1-center or the 2-center problem with respect to a given subset of points in the domain.

Notation. We denote the outer polygon of our polygonal domain P by P_0, and we use $\mathcal{H}$ to denote the collection of holes in P. Recall that $\pi(p,q)$ denotes the shortest path between two points p, q in P. For a finite set $D \subset P$, the *geodesic Voronoi diagram* of D in P, denoted $\textsc{gvd}(D)$, is the partition of P into $|D|$ Voronoi cells, where the Voronoi cell $V(q)$ of a point $q \in D$ is defined as $V_D(q) := \{x \in P : \|\pi(x,q)\| \leqslant \|\pi(x,p)\| \text{ for all } p \in D\}$. When the set D is clear from the context, we may simply write $V(q)$. For two points $p, q \in P$, let $b(p,q)$ denote their geodesic bisector and let $b_D(p,q)$ denote the part of $b(p,q)$ which appears in $\textsc{gvd}(D)$. Let $B(p,q) = \{x \in P : \|\pi(p,x)\| \leqslant \|\pi(q,x)\|\}$. We will denote by $x(p,q)$ the first point of $b(p,q)$ that is met during a clockwise transversal of ∂P_0 which starts from a point of $\partial P_0 \cap B(p,q)$. Finally, we will denote by $x_H(p,q)$ the first point of $b(p,q)$ that is met during a clockwise transversal of hole H, starting from a point of $H \cap B(p,q)$; see Fig. 1.

2 The k-median problem in a polygonal domain.

The idea is to extend the approach by Hwang *et al.* [4], which worked for $\mathbb{R}^2$, to a polygonal domain. We therefore start by considering the geodesic Voronoi diagram of our (unknown) optimal solution S. In the Euclidean case, the dual of this diagram is called the Delaunay triangulation, denoted by $\textsc{dt}(S)$. Every inner face of $\textsc{dt}(S)$ is a triangle, and one can add a

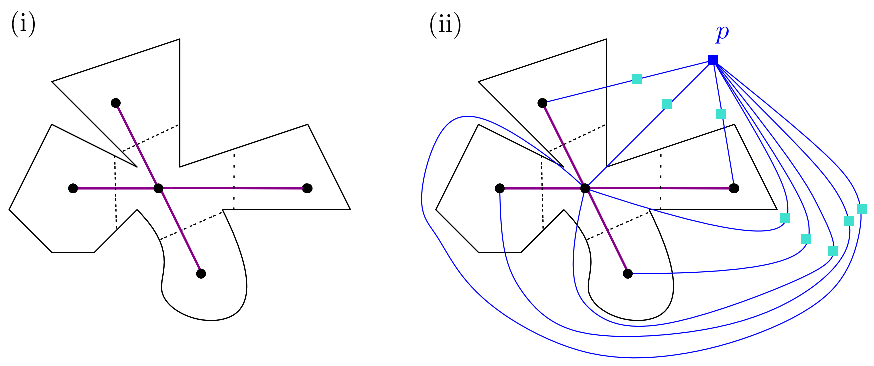


Figure 2 (i) An example where the dual of the geodesic Voronoi diagram corresponds to a tree. (ii) Adding p to the outside face of P and connecting it to cells incident to ∂P_0 via the intervals I_i.

set I of three extra points to S sufficiently far away, such that the outside face of $\text{DT}(S \cup I)$ also becomes a triangle. This results in a maximal planar graph. Hence, by Miller's separator theorem [6] there exists a simple cycle separator C of $\text{DT}(S \cup I)$ of size $O(\sqrt{k})$ which is $(2/3)$-balanced with respect to $S \cup I$. (The latter means that at most $2/3$ of the points in $S \cup I$ lie inside C and at most $2/3$ of the points in $S \cup I$ lie outside C.) In our setting, it is not guaranteed that the dual of $\text{GVD}(S)$ is an (almost) triangulated graph. See Fig.2(i) for an example where it corresponds to a tree. Therefore we will need a few extra steps before we can apply a separator theorem.

2.1 Transforming the dual of $\text{GVD}(S)$

Let $\mathcal{G} = (V, E)$ denote the dual graph of $\text{GVD}(S)$. The goal is to transform $\mathcal{G}$ to a graph $\mathcal{G}^* = (V^*, E^*)$ such that any face of $\mathcal{G}^*$ has size at most three. The Voronoi cells of $\text{GVD}(S)$ which are incident to ∂P_0, induce a decomposition of ∂P_0 into disjoint intervals. Note that it is possible for a Voronoi cell to contribute to more than one interval. The following lemma gives a linear bound on the number of these intervals.

▶ **Lemma 2.1.** *Let $I_1, \ldots, I_r$ denote the intervals along ∂P_0 induced by $\text{GVD}(S)$, enumerated in clockwise order. Then $r = O(k)$.*

Proof. For $1 \leqslant i \leqslant r$, let $s_i \in S$ be the point in our solution S whose Voronoi cell has I_i on its boundary. Note that the s_i need not all be distinct. For $i = 1, \ldots r-1$, we charge I_i to $b(s_i, s_{i+1})$. Any bisector can be charged at most two times. Moreover, a bisector uniquely corresponds to an edge of $\mathcal{G}$ and we know that $\mathcal{G}$ is a planar graph. Therefore $r \leqslant 2|E| \leqslant 6k - 12$. ◀

Now let p denote an arbitrary point in the outside face of P. We connect p to each s_i via any arbitrary interior point of I_i. Let $\{e_1, \ldots, e_r\}$ denote the set of these extra edges. Then we have so far, $V^* = V \cup \{p\}$ and $E^* = E \cup \{e_i\}_{i=1}^{r}$. It's easy to see that we can embed these edges such that: (i) they are pairwise non-crossing and (ii) any face of the resulting graph incident to p is a triangle. See Figure 2 (ii) for an example.

Handling the faces that do not contain p. Now we need to handle the faces of $\mathcal{G}^*$ that are not incident to p. By construction, the outer face of $\mathcal{G}^*$ contains p and thus is a triangle.

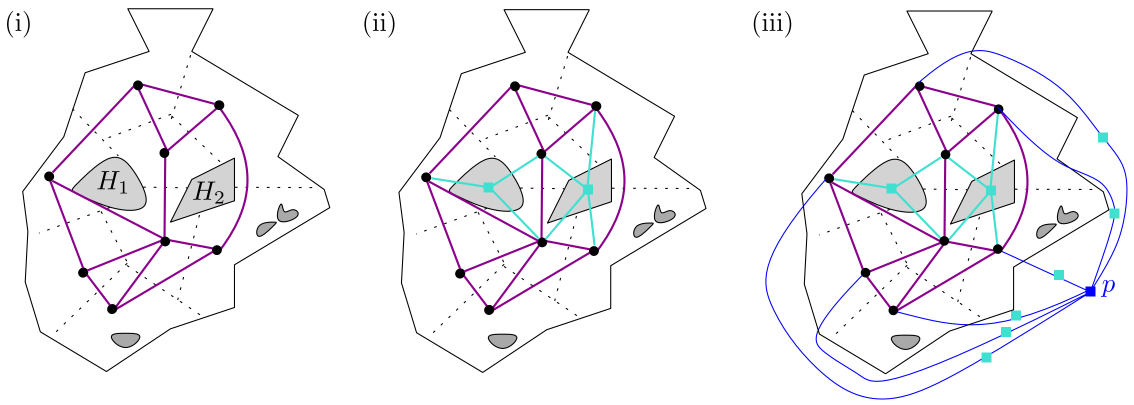


Figure 3 Holes H_1 and H_2 are essential, so we add a vertex for each of them. In (iii), observe that every face has bounded size.

Therefore, the only way $\mathcal{G}^*$ can contain a face of size at least four is if there exists a cycle of size four "around" a hole as in Figure 3 (i).

We define an *essential hole* to be a hole $H \in \mathcal{H}$ which is incident to at least four Voronoi cells of $\textsc{gvd}(S)$. Since every essential hole corresponds to a face of $\mathcal{G}$ (or $\mathcal{G}^*$), the number of essential holes is $O(k)$. Let $\mathcal{H}^*$ denote the set of essential holes and for every $H \in \mathcal{H}^*$ let p_H be an arbitrary point in H. We add the set $\{p_H\}_{H \in \mathcal{H}^*}$ to V^* and we connect p_H to the vertices of the Voronoi cells which intersect H. If $V(q)$ is such a Voronoi cell, then, similarly as before, we can embed the edge (p_H, q) by going through any interior point of $H \cap V(q)$. At the end of this process, $\mathcal{G}^*$ is a graph where every face is a triangle.

2.2 Applying the Separator Theorem to $\mathcal{G}^*$

We now want to apply Miller's Separator Theorem to $\mathcal{G}^*$. One thing that prevents us from doing so, is that $\mathcal{G}^*$ could be a multigraph, because p may be connected to the same Voronoi vertex more than once. (Recall that a Voronoi cell may contribute to more than one interval I_i). To deal with this, we can add a "dummy vertex" to each edge which has p as an endpoint; see Figure 2 (ii). This way we only increase the number of vertices and edges by $O(k)$. Moreover, the faces of the resulting graph still have bounded size, which ensures that a separator theorem can still be applied (see below). Note that we want our separator to be balanced with respect to V. To ensure that, we employ the cost-balanced version of the Planar Separator Theorem, proven by Djidjev and Venkatesan [3].

Planar Separator Theorem. Let $G = (\mathcal{V}, \mathcal{E})$ be a maximal planar graph with n nodes. Let each node $v \in \mathcal{V}$ have a non-negative weight, denoted weight(v), with $\sum_{v \in V}$ weight$(v) = 1$. Then $\mathcal{V}$ can be partitioned in $O(n)$ time into three sets A, B, C such that (i) C is a simple cycle of size $O(\sqrt{n})$, (ii) G has no arcs between a node in A and a node in B, and (iii) $\sum_{v \in A}$ weight$(v) \leqslant 2/3$ and $\sum_{v \in B}$ weight$(v) \leqslant 2/3$.

The theorem is stated for maximal planar graphs, but as pointed out in [3], it can be extended to graphs with faces of bounded size (as is our case). In our application, we give weight zero to the intermediate vertices as well as all vertices in $V^* \setminus V$, and weight $\frac{1}{|V|}$ to each vertex in V. Thus we obtain a simple-cycle separator C, which we can turn into a separator for $\mathcal{G}^*$ by ignoring any of the dummy vertices appearing on it. We obtain the following lemma.

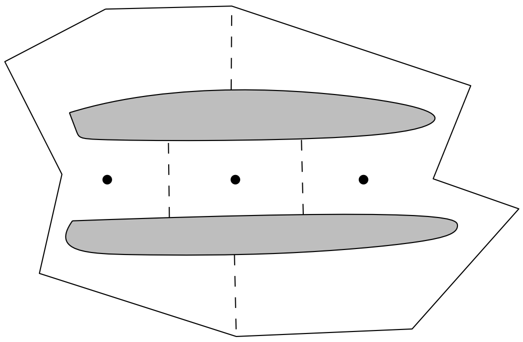


Figure 4 An example of three points and two holes, such that all three pairwise bisectors between the points intersect both holes.

▶ **Lemma 2.2.** *There exists a separator C for $\mathcal{G}^*$ with the following properties:* ***1.*** *C is a simple cycle,* ***2.*** *C has size $O(\sqrt{k})$ and* ***3.*** *C is $(2/3)$-balanced with respect to V.*

2.3 Guessing and Embedding the Separator

What we would like to do now, is guess the separator C of $\mathcal{G}^*$. Regarding the essential holes, note that we know that $|\mathcal{H}^*| = O(k)$, but we don't have any bound on $\mathcal{H}$ in terms of n. This is problematic for the running time because we will need to "guess" what the essential holes are. However, we will argue that only $O(n^3)$ holes are good candidates for being essential. We start with the following lemma.

▶ **Lemma 2.3.** *Let P be a polygon and let $\mathcal{H} = \{H_1, H_2, ..., H_m\}$ be the set of holes in P. Let $T = \{p, q, r\}$ be a set of three points in P. Then there are at most two holes in $\mathcal{H}$ that are incident to all three Voronoi cells $V_T(p), V_T(q), V_T(r)$.*

Proof. Assume for contradiction that there exist three holes H_i, H_j, H_k incident to all three cells $V_T(p), V_T(q), V_T(r)$. Since Voronoi cells are connected, for each $x \in \{p, q, r\}$ and for each $H \in \{H_i, H_{,j}, H_k\}$, there exists a path connecting x to H which stays inside $V(x)$. In this way, we get a planar embedding of $K_{3,3}$, which is a contradiction. (Note that it *is* possible to have two holes bordering $V_T(p), V_T(q), V_T(r)$, see Figure 4.) ◀

Now we define the set of *candidate essential holes* $\mathcal{R}$ as follows: for every triplet of points in D we identify at most two holes which are incident to all three pairwise bisectors between the points. We then place these holes in $\mathcal{R}$. Clearly, $|\mathcal{R}| = O(n^3)$. Therefore we can afford to guess $O(\sqrt{k})$ essential holes on our separator C. Now we give a more detailed description of how our algorithm works. Recall that each node in $\mathcal{G}^*$ (and, hence, each node on the separator we are looking for) corresponds to either a point in D, or to the extra point p we added, or to an essential hole. Thus, to find the separator, we guess all ordered subsets of size $O(\sqrt{k})$ from the set $R \cup D \cup p$. This results in $n^{O(\sqrt{k})}$ candidate separators. We then would like to use each separator to split our problem into two independent subproblems, one for the inside and one for the outside of the separator. To do that, we have to make sure that for every demand point $d \in D$ its closest center point in the optimal solution, is located at the same side of the separator as d. For this, it suffices to embed the edges of the separator such that no edge crosses a Voronoi cell of a point which is not one of its endpoints. We have three categories of edges:

- **Edges that connect p to a Voronoi site q.** Clearly if $(p,q) \in E^*$, then there exists some $r \in S$ such that $(q,r) \in E$ and then we can embed such an edge by going through the point $x(q,r)$. Note that we don't know r, but we can afford to guess it from D and therefore there are n options.
- **Edges that connect a Voronoi site q to a p_H for some $H \in \mathcal{R}$.** If $(q,p_H) \in E^*$, then again there exists some $r \in S$ such that $q,r \in E$ and then we can embed such an edge by going through $x_H(q,r)$. Again we can guess r from D and there are n options.
- **Edges that connect two Voronoi sites q and r.** Note that then one of the following holds if $(q,r) \in E$:
  1. there exists a $t \in S$, such that $V_S(q), V_S(r), V_S(t)$ meet at a point c in P or at a hole $H \in H^*$
  2. $b_S(q,r)$ intersects ∂P_0 at two points.

  Therefore we can check for all $t \in D$ whether 1. holds and if yes we embed (q,r) via c or $x_H(q,r)$. Otherwise, we can embed (q,r) via $x(q,r)$. Again we have at most n guesses.

Assuming our guessed separator is correct, the points on the separator have to be part of the optimal solution that we seek. Therefore in the two subproblems, these points have to be passed on as part of the input. If we assume that our separator has size i then we also need to guess how many of the remaining $k-i$ optimal centers lie in the inside and how many lie in the outside subproblem. In terms of running time, this is clearly not a problem since it can only give an extra factor of $O(k) = O(n)$. The base case of our algorithm is when $k = 1$, where we simply try all possible options. The recursion then leads to an algorithm with total running time $poly(n,m) + n^{O(\sqrt{k})}$.

3 The k-Center problem with outliers

To solve the k-center problem with z outliers, we first show that the same approach as our k-median algorithm works to solve the so-called (k,r)-coverage problem, and then we show how to reduce the k-center problem with z outliers to the (k,r)-coverage problem. Let P be a polygonal domain, D denote a set of n demand points in P, and k, r be two parameters. We define a *(k,r)-coverage of D* as a set of k balls of radius at most r, such that the number of outliers (that is, points in D not covered by the balls) is minimized.

The divide-and-conquer algorithm we presented earlier for k-median clustering has a base case of $k = 1$. Our algorithm relies on the n candidates for the optimal centers in k-median, and has a running time of $n^{O(\sqrt{k})}$. However, we only use the properties of k-median to solve for the base case and determine the candidate centers when guessing the separator in an optimal solution. Therefore, the same approach can be applied to solve the (k,r)-coverage problem, where for the base case, we can consider all the possible $O(n)$ balls of radius r and find the one that covers the maximum number of points in D. This means that our algorithm can compute an optimal (k,r)-coverage in $n^{O(\sqrt{k})}$ time.

It remains to reduce the k-center problem with z outliers to the (k,r)-coverage problem. Observe that if r is at least the optimal radius for k-center clustering with z outliers, then the number of outliers for (k,r)-coverage is at most z. Additionally, any minimal ball in an optimal solution for the k-center of D with z outliers contains either three points from D on its boundary or two points from D that form a diametrically opposite pair. Therefore, there are at most $O(n^3)$ candidates for the optimal radius. By performing a binary search over these $O(n^3)$ possible radii, we can find the minimum radius r^* such that the (k,r^*)-coverage covers all but at most z outliers. This (k,r^*)-coverage is an optimal solution for the k-center problem with z outliers, and the running time to find it is $O(n^{O(\sqrt{k})} \cdot \log(n^3)) = n^{O(\sqrt{k})}$.

References

1 Hee-Kap Ahn, Luis Barba, Prosenjit Bose, Jean-Lou De Carufel, Matias Korman, and Eunjin Oh. A linear-time algorithm for the geodesic center of a simple polygon. *Discret. Comput. Geom.*, 56(4):836–859, 2016. doi:10.1007/s00454-016-9796-0.

2 B. Aronov, S. Fortune, and G. Wilfong. The furthest-site geodesic voronoi diagram. In *Proceedings of the Fourth Annual Symposium on Computational Geometry*, SCG '88, page 229–240, New York, NY, USA, 1988. Association for Computing Machinery. doi:10.1145/73393.73417.

3 Hristo Djidjev and Shankar M. Venkatesan. Reduced constants for simple cycle graph separation. *Acta Informatica*, 34(3):231–243, 1997. doi:10.1007/s002360050082.

4 R. Z. Hwang, R. C. Chang, and Richard C. T. Lee. The searching over separators strategy to solve some NP-hard problems in subexponential time. *Algorithmica*, 9(4):398–423, 1993. doi:10.1007/BF01228511.

5 Dániel Marx and Michał Pilipczuk. Optimal parameterized algorithms for planar facility location problems using voronoi diagrams. *ACM Trans. Algorithms*, 18(2), mar 2022. doi:10.1145/3483425.

6 Gary L. Miller. Finding small simple cycle separators for 2-connected planar graphs. *Journal of Computer and System Sciences*, 32(3):265–279, 1986. URL: https://www.sciencedirect.com/science/article/pii/0022000086900309, doi:https://doi.org/10.1016/0022-0000(86)90030-9.

7 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. URL: https://www.sciencedirect.com/science/article/pii/S0925772119300756, doi:https://doi.org/10.1016/j.comgeo.2019.05.001.

2-point link distance queries in polygonal domains

Mart Hagedoorn[1] **and Valentin Polishchuk**[2]

1 **TU Dortmund, Germany**
`mart.hagedoorn@tu-dortmund.de`

2 **Linköping University, Sweden**
`valentin.polishchuk@alumni.stonybrook.edu`

Abstract

We show how to preprocess a polygonal domain with holes so that the link distance (the number of links in a minimum-link path) between two query points in the domain can be reported efficiently. Answering 2-point link distance queries in polygonal domains have been open questions since the same problem was studied for simple polygons in the 90s.

1 Introduction and Preliminaries

A minimum-link (minlink) path between points s, t in a polygonal domain P is an s-t path with a minimum number of edges (links); the number of links in a minlink path is the *link distance* between s and t. Unlike the geodesic distance, link distance does not obey the triangle inequality: if s and t see each other, then the link distance from any point on the segment st to both s and t is 1 – as well as the link distance between s and t themselves.

1.1 Terminology

The *visibility polygon* (VP) of a point $p \in P$ is the set of points seen by p. The *weak visibility polygon* of a subset $S \subseteq P$ is the union of the VPs of the points of S; equivalently, the weak visibility polygon are the points seen by at least one point of S. The *visibility graph* (VG) of P is the graph on vertices of P, whose edges connect pairs of mutually visible vertices. We assume that edges of P are also edges of the VG (i.e., that neighboring vertices of P see each other along the boundary edge).

For a line segment ℓ in P, let $\overline{\ell}$ be the chord of P containing ℓ (i.e., $\overline{\ell}$ is ℓ extended maximally within P). We define the Extended Visibility Graph EVG $= \{\overline{uv} : uv$ is an edge of VG$\}$ as the set of chords of P obtained by maximally extending every edge of VG.

Furthermore, the *link distance map*, denoted LDM(s), from the *source* point s is the decomposition of P into cells such that the link distance from s to any point within one cell is the same. Note that the cells of an LDM are not required to be maximal (i.e., that the link distance necessarily changes as you step from a cell to a neighboring cell): the only requirement is that the distance never changes within a cell. Thus, for a polygon P we are allowed to overlay the EVG with an LDM(s). Hence, for the remainder of the paper, when referring to LDM(s) we assume all chords of the EVG are included.

Algorithms for computing link distance [5, 7, 10, 12] employ the "staged illumination" paradigm (see, e.g., the handbooks [11, Chapter 12] and [13, Chapter 31.3]): At the first stage, place the light source at s and illuminate VP of s. At the beginning of any subsequent stage, the boundary between the lit and the dark portions of P is defined by a set of *windows* which are used to calculate the next illuminated VP.

39th European Workshop on Computational Geometry, Barcelona, Spain, March 29–31, 2023.

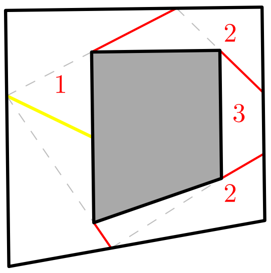


Figure 1 LDM(c) is obtained by starting the illumination from c (yellow); the edges of the map (windows) are solid red, and the cells are marked with the link distance to c

1.2 Prior work on 2-point queries and Our results

While LDMs give complete answers to the *1-point* link. (for a *fixed* source, report the distance from a query point to the source), for the *2-point* distance query problem (report the distance between two query points s and t), the solutions achieving optimal $O(\log n)$ query time are much more involved [1, 3, 6, 14].

No data structure for 2-point link distance queries in polygonal domains with holes has been known previously. The contribution of this paper is working out such data structure (analogous to the 2d decomposition for geodesic queries [3]) by extending the solution of [1] (for 2-point link distance queries in simple polygons) to polygons with holes.

2 Two-point link distance queries

In this section we show how to compute the 2d LDM equivalence decomposition – the data structures for 2-point link distance queries, analogous to the structures of [3] for geodesic queries (see Section 1.2). Our data structures extend the data structure of [1] (for 2-point link distance queries in simple polygons) to polygonal domains with holes. The extension, allowing us to track combinatorial changes in LDM(s) for $s \in P$, is two-fold:

Same windows Where for simple polygons only the boundary is decomposed in "atomic segments" [1], we decompose the whole polygon into cells by overlaying LDMs from extensions of VG edges – for all s in one cell σ of this decomposition, LDM(s) has the same set $W(\sigma)$ of windows.

Same arrangement of windows We track possible intersections among windows in LDM(s), which may change because the rotating windows sweep through the domain as s moves

2.1 Static and Rotating windows

Every window w necessarily goes through a vertex of P (Fig. 2). If the chord $\overline{w}$ goes through another vertex ($\overline{w} \in$ EVG), we call w a *pinned window* – such windows do not change as s moves locally; in particular, chords of EVG are pinned windows (the term "pinned" is borrowed from [1]). Otherwise we call w a *bash window*, and the endpoints of $\overline{w}$ – *bash points*. A bash point lies in the interior of an edge of P; we call such an edge a *bash wall* (the term "bash" is borrowed from [9]). As s moves slightly, pinned windows of LDM(s) do not change, whilst a bash window w may remain still or may rotate (refer to Fig. 3 – as s moves left, all windows rotate). Any window w coming after a pinned window or bash window is called a static bash window or bash-bash window, respectively.

Overall, we obtain the classification of LDM windows into static and rotating. The former may be EVG chords or static bash windows; the latter are bash-bash windows.

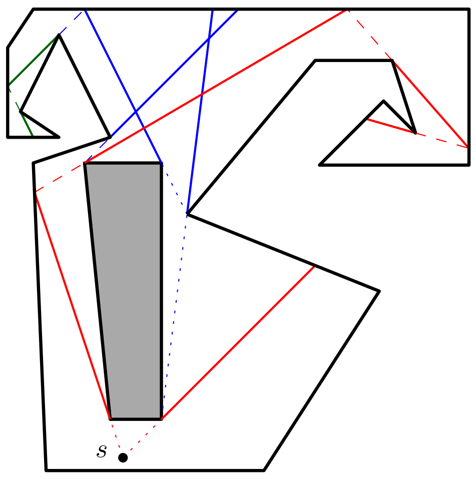


Figure 2 Pinned windows (those not aligned with edges of P) are blue (the aligned VG edges are dotted blue), and static bash windows are green, rotating windows are red.

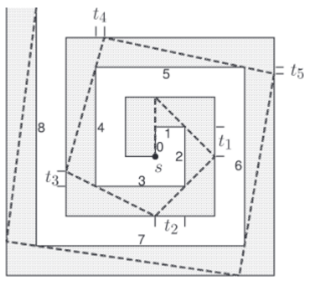


Figure 3 Figure 1 from [8]: the coordinates of the vertex t_1 of the path are obtained as a solution to a system of 2 linear equations with 2 unknowns and integer coefficients

2.2 LDMs overlay

We build LDMs from all chords of the EVG; let W_{EVG} denote the set of all windows in these LDMs. The first step in constructing our data structures is computing the overlay $\mathcal{O}$ of the windows in W_{EVG}. Similar to [1], we build the full overlay of the LDMs, while [1] considered only the interaction of W_{EVG} with the boundary of P.

Now, consider LDM(s) for a point s in P. As s moves, the bash-bash windows of LDM(s) rotate and the map may change combinatorially when either:

Simple case A window hits a vertex of P when s crosses an edge of $\mathcal{O}$, or

New case The arrangement of the windows changes because 3 windows pass through a common point (out of the 3 windows, at least one must be a rotating window).

We called the first event "simple" because this is the only thing that may happen in a simple polygon (where windows are pairwise-disjoint). As we will show in Section 2.5, to account for new events one needs to build LDMs from intersection points of windows in W_{EVG}, as well as to do some additional, less straightforward computations described later.

An important connection between the overlay $\mathcal{O}$ and static windows in LDMs is that all possible static windows in an LDM from any point of P are known in advance. Note that we do not claim that any window from W_{EVG} is necessarily a window in any LDM.

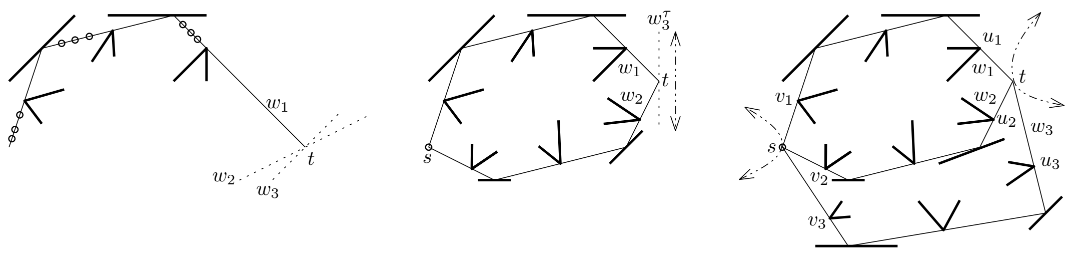


Figure 4 P is bold, static windows from W_{EVG} are dotted, bash paths are solid. Left: LDM(s) may change combinatorially when s crosses a bash-bash window of LDM(t) where $t = w_1 \cap w_2$ for some static windows $w_1, w_2 \in W_{\text{EVG}}$ of LDM(s); some possible locations for such s are shown with hollow circles. Middle: LDM(s) may change when there are two bash paths between s and $t \in w_3$. Right: Each of s, t moves on a curve so as to remain connected by 3 bash paths.

2.3 Parametric maps (for simple cases)

If the new cases are ignored, then LDMs (with the corresponding projection functions) built for all cells of $\mathcal{O}$ define the 2d LDM equivalence decomposition. The data structure can be used to answer 2-point link distance queries in the same way as the 2d SPM equivalence decomposition [3] answers geodesic distance queries: given query points s and t, first s is located in a cell σ of $\mathcal{O}$ and then t is located in the parametric LDM(s).

In fact, since LDM edges (the windows) are straight line segments (not hyperbolic arcs as edges of SPM), one possibility for locating t in the parametric LDM(s) is to use the monotone subdivision method of [4].

2.4 Triple points (for new cases)

What remains is to account for the new cases (window arrangement changes due to 3 windows passing through a common point). In Section 2.5 we give an algorithm to compute the locations $\mathcal{S} \subset P$ for s such that 3 windows in LDM(s) may intersect in a common point t. The set $\mathcal{S}$ is overlaid with $\mathcal{O}$. By construction, for all points s in one cell of the obtained 2d overlay $\mathcal{O}^*$, LDM(s) is the same combinatorially: it has the same windows and they form the same arrangement (before the arrangement changes, 3 windows must pass through a common point, at which moment s is in $\mathcal{S}$). Our final data structure, the (full) 2d LDM equivalence decomposition ("full" in the sense that it accounts for both simple and new cases) is built from $\mathcal{O}^*$ in the same way as the data structure for handling simple cases was built from $\mathcal{O}$ (Section 2.3).

2.5 Intersecting 3 windows

To find the (super)set $\mathcal{S}$ of sources potentially having triple points in their LDMs, we first build another decomposition for tracking bash-bash windows (Section **??**). We are now ready to compute $\mathcal{S}$. We emphasize that we do not claim that LDM(s) has a triple point for every s in $\mathcal{S}$; we only claim that $\mathcal{S}$ is sufficiently rich to "catch" all possible (sources of) maps with triple points.

Let w_1, w_2, w_3 be windows of LDM(s) that intersect at t, of which at least 1 window rotates (so that LDM(s) changes combinatorially when the 3 windows intersect at t). Recall that the (super)set W_{EVG} of possible static windows in LDM(s) does not depend on s. We make 3 different guesses on how many of the 3 windows w_1, w_2, w_3 are rotating (1, 2, or all 3), and do different things for each of the guesses (Fig. 4):

- Assuming only w_1 is rotating, $\mathrm{LDM}(s)$ changes when w_1 passes through $t = w_2 \cap w_3$ (the intersection point of two static windows w_2, w_3). At this point there is a bash path between s and t, implying that s is on a bash-bash window of $\mathrm{LDM}(t)$. We thus build $\mathrm{LDM}(t)$ from every intersection point t of two (potential) static windows $w_2, w_3 \in W_{\mathrm{EVG}}$ and add to $\mathcal{S}$ all bash-bash windows of LDMs.
- Assuming 2 windows (say, w_1, w_2) are rotating, $\mathrm{LDM}(s)$ changes when their intersection point t passes through w_3. At this point there are two bash paths between s and t (in one path $t \in w_1$, in the other $t \in w_2$). That is, s is the intersection point of two rotating windows in $\mathrm{LDM}(t)$ (one window belonging to the bash path from t that starts from following w_1; the other – following w_2). We thus take every window $w_3 \in W_{\mathrm{EVG}}$ as the potential static window in $\mathrm{LDM}(s)$ and intersect it with every cell τ of $\mathcal{O}$; let w_3^τ denote part of w_3 inside τ. For all points $t \in w_3^\tau$, $\mathrm{LDM}(t)$ has the same set of windows, and in particular, the same set of bash-bash windows; each window is a known function of t (via the projection functions). For every pair of the bash-bash windows (with the same link distance from t) in $\mathrm{LDM}(t)$ we add to $\mathcal{S}$ the curve traced by their intersection as t varies along w_3^τ.
- Assuming all 3 windows are rotating, we go through all pairs of cells σ, τ in $\mathcal{D}$, guessing that $s \in \sigma, t \in \tau$. In addition, we go through all triples v_1, v_2, v_3 of vertices visible from s and all triples u_1, u_2, u_3 of vertices visible to t, guessing that the vertices support the first and the last windows resp. of the 3 bash paths (ending with the windows w_1, w_2, w_3) between s and t. Since vertices of P are endpoints of atomic segments, the decomposition $\mathcal{D}$ includes chords of EVG, implying that the set of vertices visible to any point in a cell of $\mathcal{D}$ is the same: it is thus legitimate to speak about a triple of vertices visible to $s \in \sigma$ or to $t \in \tau$ without specifying the exact locations of s and t in the cells.

  Using the projection functions, we know whether for some number k of links there indeed exist 3 length-k bash paths from s, with the first links of the paths supported by v_1, v_2, v_3 and last links supported by u_1, u_2, u_3. If yes, let $w_1(s), w_2(s), w_3(s)$ be these last links (windows of $\mathrm{LDM}(s)$). For t to be a triple point, the system

$$\begin{cases} t \in w_1(s) \\ t \in w_2(s) \\ t \in w_3(s) \end{cases} \tag{1}$$

  of 3 equations (each involving the projection function) with 4 unknowns (the coordinates of s and t) must be satisfied (for $s \in \sigma, t \in \tau$). We solve the system and obtain the decomposition for the 2d LDM equivalence decomposition (the set $\mathcal{S} \subseteq P$ such that $\mathrm{LDM}(s)$ may change combinatorially only when s is crossing $\mathcal{S}$). That is, for the 2d set $\mathcal{S}$ we take only the first two coordinates of the 4d pairs $(s,\ t)$ (i.e., the coordinates for s; the knowledge of corresponding locations for t is ignored). Fig. 5 is a snapshot of a GeoGebra example for $\mathcal{S}$ (dashed pink curve).

▶ **Theorem 2.1.** *2-point link distance queries in P can be answered in $O(\log n)$ time after polynomial-time preprocessing.*

We focused on reporting the link distance between query points s, t. To report the minlink s-t path, we can enhance the parametric LDMs with backpointers (similarly to [10]).

3 Conclusions

We presented data structures for 2-point link distance queries in polygonal domains. We were only after polynomiality, and one obvious question is improving efficiency of our algorithms (fast solutions with small additive errors are known [1]). As far as 2-point link distance queries go, in polygonal domains with holes, even the visibility queries (Do s and t see each other, i.e., is the link distance between them equal to 1?) are quite challenging [2].

References

1 Esther M Arkin, Joseph SB Mitchell, and Subhash Suri. Logarithmic-time link path queries in a simple polygon. *International Journal of Computational Geometry & Applications*, 5(04):369–395, 1995.

2 Danny Z Chen and Haitao Wang. Visibility and ray shooting queries in polygonal domains. *Computational Geometry*, 48(2):31–41, 2015.

3 Yi-Jen Chiang and Joseph SB Mitchell. Two-point euclidean shortest path queries in the plane. In *Proc. 10th ACM-SIAM Symposium on Discrete Algorithms*. Citeseer, 1999.

4 Herbert Edelsbrunner, Leonidas J Guibas, and Jorge Stolfi. Optimal point location in a monotone subdivision. *SIAM Journal on Computing*, 15(2):317–340, 1986.

5 Subir Kumar Ghosh. Computing the visibility polygon from a convex set and related problems. *Journal of Algorithms*, 12(1):75–95, 1991.

6 Leonidas J Guibas and John Hershberger. Optimal shortest path queries in a simple polygon. In *Proceedings of the third annual symposium on Computational geometry*, pages 50–63, 1987.

7 John Hershberger and Jack Snoeyink. Computing minimum length paths of a given homotopy class. *Computational geometry*, 4(2):63–97, 1994.

8 Simon Kahan and Jack Snoeyink. On the bit complexity of minimum link paths: Superquadratic algorithms for problems solvable in linear time. In *Proceedings of the twelfth annual symposium on Computational geometry*, pages 151–158, 1996.

9 Joseph SB Mitchell, C Piatko, and Esther M Arkin. Computing a shortest k-link path in a polygon. In *Proceedings., 33rd Annual Symposium on Foundations of Computer Science*, pages 573–582. IEEE, 1992.

10 J S. B. Mitchell, G Rote, and GJ Woeginger. Minimum-link paths among obstacles in the plane. *Algorithmica*, 8(1):431–459, 1992.

11 Jörg-Rüdiger Sack and Jorge Urrutia. *Handbook of computational geometry*. Elsevier, 1999.

12 Subhash Suri. A linear time algorithm for minimum link paths inside a simple polygon. *Computer Vision, Graphics, and Image Processing*, 35(1):99–110, 1986.

13 Csaba D Toth, Joseph O'Rourke, and Jacob E Goodman. *Handbook of discrete and computational geometry*. CRC press, 2017.

14 Haitao Wang. A divide-and-conquer algorithm for two-point L1 shortest path queries in polygonal domains. *J. Comput. Geom.*, 11(1):235–282, 2020. doi:10.20382/jocg.v11i1a10.

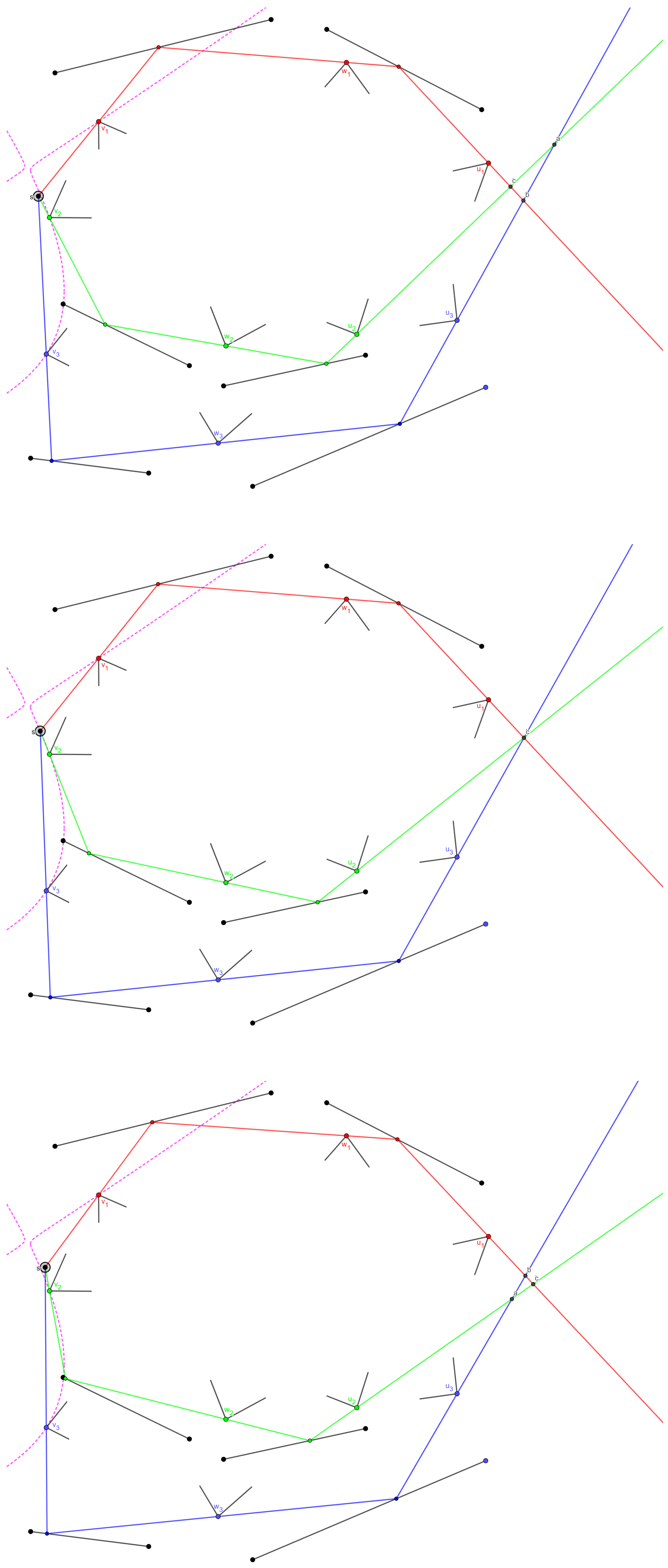


Figure 5 `https://www.geogebra.org/classic/sumsrytt`: P's edges and vertices are black; move s slightly and see LDM change (triple point appearing) as s crosses $\mathcal{S}$ (dashed pink)

Towards Crossing-Free Hamiltonian Cycles in Simple Drawings of Complete Graphs*

Oswin Aichholzer[1], **Joachim Orthaber**[1], **and Birgit Vogtenhuber**[1]

1 **Institute of Software Technology, Graz University of Technology, Austria**
`{oaich,orthaber,bvogt}@ist.tugraz.at`

Abstract

It is a longstanding conjecture that every simple drawing of a complete graph on $n \geq 3$ vertices contains a crossing-free Hamiltonian cycle. We confirm this conjecture for cylindrical drawings, strongly c-monotone drawings, as well as x-bounded drawings. Moreover, we introduce the stronger question of whether a crossing-free Hamiltonian path between each pair of vertices always exists.

1 Introduction

A *simple drawing* is a drawing of a graph where each pair of edges meets in at most one point (a crossing or a common endpoint) and no edge crosses itself. A fundamental line of research is concerned with finding *crossing-free* sub-drawings (that is, sub-drawings with pairwise non-crossing edges; also called *plane* sub-drawings) in simple drawings of the complete graph K_n on n vertices. In 1988, Nabil Rafla stated the following conjecture in his PhD thesis [18].

▶ **Conjecture 1** (Rafla [18]). *Every simple drawing of the complete graph K_n on $n \geq 3$ vertices contains at least one crossing-free Hamiltonian cycle.*

Two simple drawings $\mathcal{D}$ and $\mathcal{D}'$ of the same graph are called *weakly isomorphic* if two edges in $\mathcal{D}$ cross if and only if the corresponding edges in $\mathcal{D}'$ have a crossing. They are called *strongly isomorphic* if there exists a homeomorphism (on the sphere) mapping $\mathcal{D}$ to $\mathcal{D}'$. Weak isomorphism classes can be uniquely represented by rotation systems (see [1, 15] for details).

Related Work. Under the assumption that Conjecture 1 is true, Rafla enumerated all different simple drawings of K_n for $n \leq 7$ up to weak isomorphism. Since then, Conjecture 1 and relaxations of it have attracted considerable attention. Especially, note that a crossing-free Hamiltonian cycle in a simple drawing $\mathcal{D}$ implies that $\mathcal{D}$ also contains a crossing-free Hamiltonian path (just remove an arbitrary edge of the cycle). Furthermore, for even n, a crossing-free Hamiltonian path in turn implies that $\mathcal{D}$ contains a plane perfect matching (take every second edge in the path). However, even the question of the existence of a plane perfect matching in every simple drawing of K_{2n} is still open.

In 2003 Pach, Solymosi, and Tóth [17] showed that every simple drawing of K_n contains plane sub-drawings isomorphic to any tree of size $\mathcal{O}(\log(n)^{1/6})$. This immediately implies a lower bound of $\Omega(\log(n)^{1/6})$ for the largest crossing-free path and largest plane matching in every simple drawing of K_n. Subsequently, a lot of progress has been made with regard to plane matchings (see [5, 19] and references therein). Until recently, a lower bound of

* O.A. and J.O. partially supported by the Austrian Science Fund (FWF) grant W1230. B.V. partially supported by Austrian Science Fund within the collaborative DACH project *Arrangements and Drawings* as FWF project I 3340-N35.

$\Omega(n^{1/2-\varepsilon})$, shown by Ruiz-Vargas [19], was best known. This bound has lately been improved to $\Omega(\sqrt{n})$ in [5], via the introduction and use of generalized twisted drawings.

In the same paper a lower bound of $\Omega(\log(n)/\log(\log(n)))$ for the longest crossing-free path was shown; this is the first improvement in that direction over the result from [17]. Furthermore, the authors of [5] obtained the same bound for the longest crossing-free cycle.

In another direction, in [17] it was also shown that every simple drawing of K_n contains a sub-drawing of size $\Omega(\log(n)^{1/8})$ which is weakly isomorphic to a convex straight-line drawing or a so-called twisted drawing (which has been introduced by Harborth and Mengersen in the context of maximally crossing drawings [14] and empty triangles [13]). This implies the existence of various plane sub-drawings of the respective size. Recently, Suk and Zeng [20] improved the above bound from [17] to $\Omega(\log(n)^{1/4-\varepsilon})$ and also (independently of [5]) proved the existence of a crossing-free path of length $\Omega(\log(n)^{1-\varepsilon})$ in every simple drawing of K_n.

Furthermore, Ruiz-Vargas [19] showed that every c-monotone drawing of K_n contains a plane matching of size $\Omega(n^{1-\epsilon})$; so "almost" a perfect matching. Also, in [5] it is shown that every c-monotone drawing contains a sub-drawing of size $\Omega(\sqrt{n})$ that is weakly isomorphic to either an x-monotone drawing or a generalized twisted drawing, implying that c-monotone drawings of K_n contain a crossing-free path as well as a crossing-free cycle of size $\Omega(\sqrt{n})$.

Concerning crossing-free Hamiltonian cycles, Conjecture 1 has been confirmed for all simple drawings on $n \leq 9$ vertices using the rotation system database [1], and Ebenführer tested the conjecture on randomly generated realizable rotation systems for up to 30 vertices in his Master's thesis [9]. Furthermore, in [3, 9] it was shown that simplicity of the drawings is crucial, by providing a star-simple drawing (non-incident edges are allowed to cross more than once) of K_6 that does not contain any "crossing-free" Hamiltonian cycle (where edges are only considered to be "crossing" when they cross an odd number of times).

Finally, Arroyo, Richter, and Sunohara [7] showed the existence of a crossing-free Hamiltonian cycle in so-called pseudospherical (or h-convex) drawings of K_n. In a current paper, Bergold et al. [8] extend this to (generalized) convex drawings. And in [5] Conjecture 1 is shown to be true for generalized twisted drawings on an odd number of vertices.

Our Contribution. We extend this line of research, showing Conjecture 1 to be true for cylindrical drawings as well as strongly c-monotone drawings. Moreover, we show the inclusion of (strongly) cylindrical drawings in (strongly) c-monotone drawings and the equivalence of x-monotone and x-bounded drawings of K_n, from which it follows that Conjecture 1 is also true for x-bounded drawings. Finally, we consider the question whether there exists a crossing-free Hamiltonian path between each pair of vertices, which we show to be a generalization of Conjecture 1.

All missing proofs can be found in the full version of this paper.

2 Crossing-Free Hamiltonian Cycles

We start by defining some sub-classes of simple drawings and analyzing relations between them.

If every vertical line in the plane crosses each edge of a simple drawing $\mathcal{D}$ at most once, we call $\mathcal{D}$ an *x-monotone drawing.* When the relative interior of each edge is contained between the vertical lines through its left and right end-vertices we call $\mathcal{D}$ an *x-bounded drawing.* Obviously x-bounded drawings are a generalization of x-monotone drawings. Interestingly, for drawings of K_n these classes are basically the same (Fulek et al. [10] show a similar result on not necessarily simple drawings of not necessarily complete graphs).

▶ **Theorem 2.1.** *For every x-bounded drawing $\mathcal{D}$ of K_n there exists a weakly isomorphic x-monotone drawing $\mathcal{D}'$.*

As a generalization of Hill's drawing of K_n (confer [11, 12]), we call a simple drawing *cylindrical* if all vertices lie on two concentric circles and no edge crosses any of these two circles (this is the version of cylindrical drawings introduced in [2]). If, in addition, all edges connecting vertices on the inner (outer, respectively) circle lie inside (outside, respectively) that circle, then we call the drawing *strongly cylindrical*.

We say that an edge e in a simple drawing is *c-monotone with respect to a point p of the plane* if every ray starting at p crosses e at most once. We call a simple drawing $\mathcal{D}$ in the plane a *c-monotone drawing* if all edges in $\mathcal{D}$ are c-monotone with respect to a common point p_c (as defined in [5]). If, in addition, for each star $\mathcal{S}$ in $\mathcal{D}$, there exists a ray starting at p_c that does not cross any edge of $\mathcal{S}$, we say that $\mathcal{D}$ is *strongly c-monotone*.

We state three more results before coming to crossing-free Hamiltonian cycles. In addition, Figure 1 gives an overview on more classes and their relations.

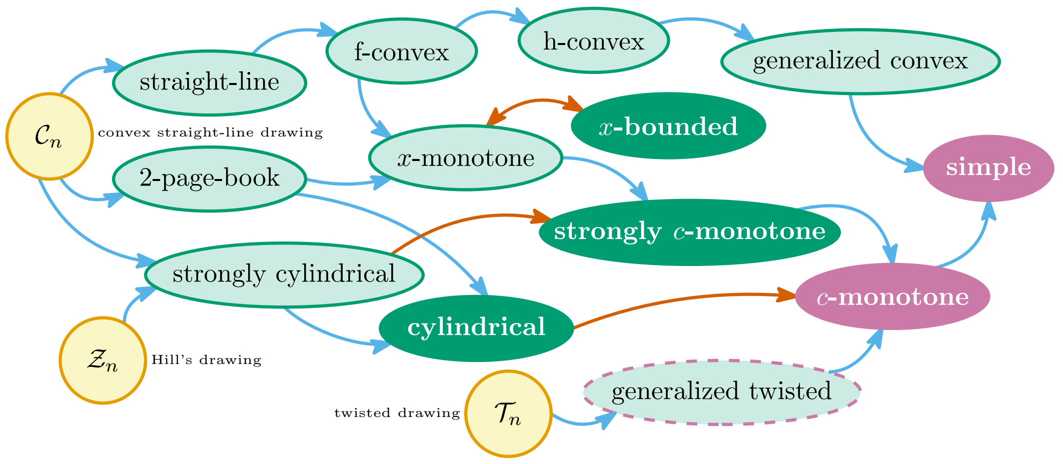


Figure 1 Relations between special drawings (yellow) and classes of simple drawings of K_n (seagreen/violet). Arrows indicate that the "source class" is contained in the "target class" (concerning weak isomorphism); darkorange arrows are shown in the full version of this work. Conjecture 1 is (now) known to be true for the yellow/seagreen classes; for the darker seagreen ones this is shown below.

▶ **Lemma 2.2.** *Let e be an edge of a strongly c-monotone (with respect to p_c) drawing of K_n. Then the sub-drawing induced by all vertices in the wedge bounded by the rays from p_c through the end-vertices of e and containing e is strongly isomorphic to an x-monotone drawing.*

▶ **Lemma 2.3.** *In every cylindrical drawing, per circle, there exists at most one edge between neighboring vertices that is crossed by other edges.*

▶ **Theorem 2.4.** *For every cylindrical drawing $\mathcal{D}$ there exists a weakly isomorphic drawing $\mathcal{D}'$ that is c-monotone. Moreover, for every strongly cylindrical drawing $\mathcal{D}$ there exists a weakly isomorphic drawing $\mathcal{D}'$ that is strongly c-monotone.*

For straight-line drawings of K_n it is easy to see that a crossing-free Hamiltonian cycle always exists (for example, pick an arbitrary vertex v, visit all other vertices in circular order around v, and add v at some position to close the cycle). Further, it was known that every 2-page-book, x-monotone, and strongly cylindrical drawing of K_n contains a crossing-free Hamiltonian cycle (see, for example, [4]); however, as we are not aware of a reference containing proofs for these statements, we present such proofs in this work, starting with x-monotone and x-bounded drawings (which include 2-page-book drawings as a sub-class). We remark that 2-page-book drawings of K_n with $n \geq 3$ actually contain a Hamiltonian cycle of completely uncrossed edges.

▶ **Theorem 2.5.** *Every x-monotone and every x-bounded drawing of the complete graph K_n on $n \geq 3$ vertices contains at least one crossing-free Hamiltonian cycle.*

Proof. First, let $\mathcal{D}$ be an x-monotone drawing of K_n, let the vertices $v_1, \ldots, v_n$ be in that order from left to right in horizontal direction, and see Figure 2 for an example illustration of the construction. Consider the edge $e = \{v_1, v_n\}$ and the Hamiltonian path $\mathcal{P} = v_1 v_2 \ldots v_n$ (which is crossing-free by the definition of x-monotone drawings). If e does not cross $\mathcal{P}$ then $e + \mathcal{P}$ is a crossing-free Hamiltonian cycle. Otherwise, e has $k > 0$ crossings with $\mathcal{P}$ and partitions the vertices of $\mathcal{D} \setminus \{v_1, v_n\}$ into a set above e and a set below e.

Our goal is to find crossing-free paths $\mathcal{P}_1$ and $\mathcal{P}_2$ from v_1 to v_n, which visit all vertices above and below e, respectively. Let x_i be the i-th crossing between e and $\mathcal{P}$ from left to right (in horizontal direction, which is the same as along $\mathcal{P}$ or e). Further, let v_{a_i} and v_{b_i} be the vertices directly before and after x_i, respectively. Then the edge f_0 from v_1 to v_{b_1} and the edge f_k from v_{a_k} to v_n cannot cross e (because e is incident to f_0 and f_k). Similarly, for every crossing x_i ($1 \leq i \leq k-1$) the edge f_i from v_{a_i} to $v_{b_{i+1}}$ cannot cross e because otherwise, f_i and e would have to cross at least twice. In other words, for $1 \leq i \leq k$, the edges f_i alternate between lying completely above and completely below e.

Therefore, the edges f_i lying above e combined with all edges of $\mathcal{P}$ that also lie above e (basically, sub-paths of $\mathcal{P}$ from v_1 or $v_{b_{i-1}}$ to v_{a_i} or v_n) form a crossing-free path $\mathcal{P}_1$ from v_1 to v_n (because the edges lie in separate vertical strips and the start-/end-vertices coincide) visiting all vertices above e. In the same manner, there is a crossing-free path $\mathcal{P}_2$ visiting all vertices below e. Then joining $\mathcal{P}_1$ and $\mathcal{P}_2$ results in a crossing-free Hamiltonian cycle because e separates $\mathcal{P}_1$ and $\mathcal{P}_2$, which completes the proof for x-monotone drawings.

For x-bounded drawings, the statement follows from the above proof and Theorem 2.1. ◀

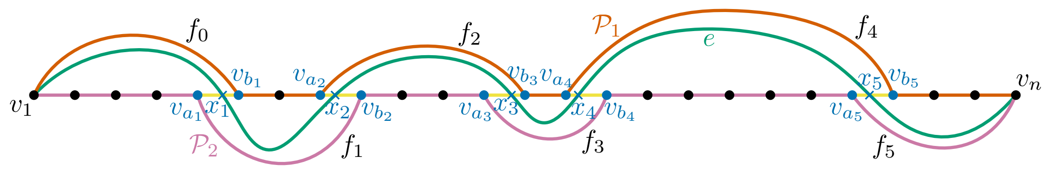


Figure 2 Constructing a crossing-free Hamiltonian cycle in an x-monotone drawing of K_n from crossing-free paths $\mathcal{P}_1$ (darkorange) above and $\mathcal{P}_2$ (violet) below the edge $e = \{v_1, v_n\}$ (seagreen).

The result on strongly c-monotone drawings follows now almost immediately.

▶ **Theorem 2.6.** *Every strongly c-monotone drawing of the complete graph K_n on $n \geq 3$ vertices contains at least one crossing-free Hamiltonian cycle.*

Proof. Let the vertices v_1 to v_n be in that order counter-clockwise around p_c and consider the n edges $e_i = \{v_i, v_{i+1}\}$ between neighboring vertices (see Figure 3(a) for visual assistance). If all of them are in the "short" direction (counter-clockwise from v_i to v_{i+1}) around p_c, then they form a crossing-free Hamiltonian cycle (by the definition of c-monotone drawings) and we are done. Otherwise there is some edge e_j (for $1 \leq j \leq n$) going the "long" direction (clockwise from v_j to v_{j+1}) around p_c. But then the whole drawing is strongly isomorphic to an x-monotone drawing by Lemma 2.2 (for which being *strongly* c-monotone is crucial). Therefore we know by Theorem 2.5 that a crossing-free Hamiltonian cycle exists. ◀

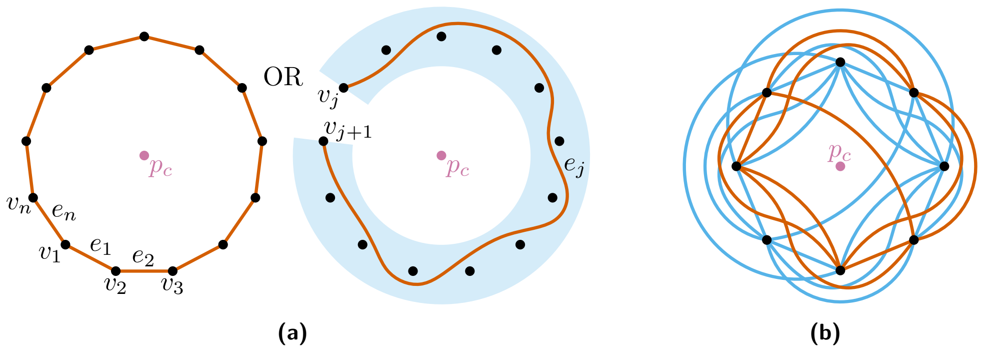


Figure 3 **(a)** A crossing-free Hamiltonian cycle in a strongly c-monotone drawing of K_n: Either visiting the vertices in circular order around p_c is sufficient or the drawing is strongly isomorphic to an x-monotone drawing. **(b)** A strongly c-monotone drawing that is neither x-monotone nor cylindrical nor generalized convex (the darkorange K_5 cannot be drawn straight-line).

In Figure 3(b) we give an example of a strongly c-monotone drawing that is neither x-monotone nor cylindrical (it does not have any uncrossed edge) and also not generalized convex (it contains a non-straight-line drawing of K_5; confer Arroyo et al. [6]).

We conclude by verifying Conjecture 1 for cylindrical drawings, using the same idea as in a previously known proof for *strongly* cylindrical drawings. Note that for strongly cylindrical drawings, Conjecture 1 is also true by Theorem 2.4 together with Theorem 2.6.

▶ **Theorem 2.7.** *Every cylindrical drawing of the complete graph K_n on $n \geq 3$ vertices contains at least one crossing-free Hamiltonian cycle.*

Proof. Assume first that there are at least two vertices on each circle and confer Figure 4. Then by Lemma 2.3, every cylindrical drawing contains two completely uncrossed paths $\mathcal{P}_1$ and $\mathcal{P}_2$ (one per circle) that together contain all vertices. Consider the end-vertices v_a and v_b of $\mathcal{P}_1$, and v_c and v_d of $\mathcal{P}_2$. Then both, the pair of edges $\{v_a, v_c\}$ and $\{v_b, v_d\}$, and the pair $\{v_a, v_d\}$ and $\{v_b, v_c\}$, connect the completely uncrossed paths $\mathcal{P}_1$ and $\mathcal{P}_2$ to a Hamiltonian cycle. Since there can be at most one crossing in the sub-drawing induced by the four-tuple of vertices $\{v_a, v_b, v_c, v_d\}$, at least one of those two Hamiltonian cycles is crossing-free.

Finally, if there is only a single vertex v on one of the circles, then the two edges connecting v to $\mathcal{P}_2$, the completely uncrossed path on the other circle, are incident; therefore, they do not cross anyway. And if all vertices lie on the same circle, then the drawing is strongly isomorphic to a 2-page-book drawing and the result follows from Theorem 2.5. ◀

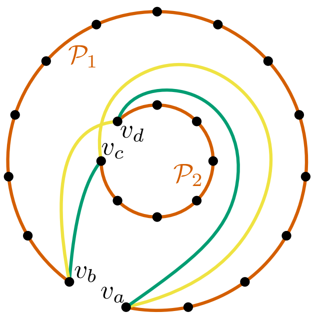


Figure 4 In a cylindrical drawing of K_n: Connecting the two completely uncrossed paths of rim edges (darkorange) with one of two pairs of lateral edges (seagreen/yellow) to a crossing-free Hamiltonian cycle.

3 Conclusion

We showed the existence of a crossing-free Hamiltonian cycle in every strongly c-monotone drawing and in every cylindrical drawing of K_n. By Theorem 2.1, we also extended the result to x-bounded drawings. Furthermore, this work contains the first published proofs of Conjecture 1 for 2-page-book, x-monotone, and strongly cylindrical drawings.

During our research, in addition, we came up with the following conjecture.

▶ **Conjecture 2.** *Every simple drawing $\mathcal{D}$ of K_n for $n \geq 1$ contains, for each pair of vertices v_a and v_b in $\mathcal{D}$, a crossing-free Hamiltonian path starting in v_a and ending in v_b.*

In the full version we show that this conjecture is in fact at least as strong as Conjecture 1.

▶ **Theorem 3.1.** *A positive answer to Conjecture 2 implies a positive answer to Conjecture 1. In particular, if Conjecture 2 is true for all simple drawings of K_{n+1} for some $n \geq 3$ then Conjecture 1 is true for all simple drawings of K_n.*

We can confirm Conjecture 2 for all simple drawings on $n \leq 9$ vertices using the rotation system database. In the full version, we show it to be true for cylindrical and strongly c-monotone drawings as well. A next goal is to extend those results to more classes of simple drawings, especially, generalized twisted drawings on an even number of vertices. Further, the classes of c-monotone drawings and crossing maximal drawings are of interest, too.

Another intriguing question is to figure out the essential reason why Conjecture 1 should be true in general for simple drawings, while it is not true anymore for star-simple drawings.

Moreover, it would be interesting to know whether Theorem 3.1 can be strengthened to an equivalence of Conjectures 1 and 2. We remark, however, that even if Conjecture 2 is strictly stronger than Conjecture 1, it could potentially be easier to prove.

Acknowledgments. All results presented in this work are also contained in the Master's thesis [16] of Joachim Orthaber. We thank Rosna Paul, Daniel Perz, and Alexandra Weinberger for fruitful discussions. We also thank the three anonymous referees for their helpful comments, including the suggestion to use better distinguishable colors in the figures. The colors we now use were recommended in [21].

References

1 Bernardo M. Ábrego, Oswin Aichholzer, Silvia Fernández-Merchant, Thomas Hackl, Jürgen Pammer, Alexander Pilz, Pedro Ramos, Gelasio Salazar, and Birgit Vogtenhuber. All good drawings of small complete graphs. In *Proceedings of the 31st European Workshop on Computational Geometry (EuroCG 2015)*, pages 57–60, 2015. URL: `http://eurocg15.fri.uni-lj.si/pub/eurocg15-book-of-abstracts.pdf`.

2 Bernardo M. Ábrego, Oswin Aichholzer, Silvia Fernández-Merchant, Pedro Ramos, and Gelasio Salazar. Shellable drawings and the cylindrical crossing number of K_n. *Discrete & Computational Geometry*, 52(4):743–753, 2014. `doi:10.1007/s00454-014-9635-0`.

3 Oswin Aichholzer, Florian Ebenführer, Irene Parada, Alexander Pilz, and Birgit Vogtenhuber. On semi-simple drawings of the complete graph. In *Proceedings of the XVII Spanish Meeting on Computational Geometry (EGC 2017)*, pages 25–28, 2017. URL: `https://dmat.ua.es/en/egc17/documentos/book-of-abstracts.pdf`.

4 Oswin Aichholzer, Alfredo García, Irene Parada, Birgit Vogtenhuber, and Alexandra Weinberger. Simple drawings of $K_{m,n}$ contain shooting stars. In *Proceedings of the 36th European Workshop on Computational Geometry (EuroCG 2020)*, pages 36:1–36:7, 2020. URL: `https://www1.pub.informatik.uni-wuerzburg.de/eurocg2020/data/uploads/papers/eurocg20_paper_36.pdf`.

5 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.

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):27, 2022. doi:10.26493/1855-3974.2134.ac9.

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, 2021. doi:10.1137/20M1313234.

8 Helena Bergold, Stefan Felsner, Meghana M. Reddy, and Manfred Scheucher. Using SAT to study plane substructures in simple drawings. In *Proceedings of the 39th European Workshop on Computational Geometry (EuroCG 2023)*, pages 2:1–2:7, 2023.

9 Florian Ebenführer. Realizability of rotation systems. Master's thesis, Graz University of Technology, Austria, 2017. URL: https://diglib.tugraz.at/realizability-of-rotation-systems-2017.

10 Radoslav Fulek, Michael J. Pelsmajer, Marcus Schaefer, and Daniel Štefankovič. Hanani–Tutte, monotone drawings, and level-planarity. In *Thirty essays on geometric graph theory*, pages 263–287. Springer, 2013. doi:10.1007/978-1-4614-0110-0_14.

11 Richard K. Guy, Tom Jenkyns, and Jonathan Schaer. The toroidal crossing number of the complete graph. *Journal of Combinatorial Theory*, 4(4):376–390, 1968. doi:10.1016/S0021-9800(68)80063-8.

12 Frank Harary and Anthony Hill. On the number of crossings in a complete graph. *Proceedings of the Edinburgh Mathematical Society*, 13(4):333–338, 1963. doi:10.1017/S0013091500025645.

13 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.

14 Heiko Harborth and Ingrid Mengersen. Drawings of the complete graph with maximum number of crossings. In *Proceedings of the 23rd Southeastern International Conference on Combinatorics, Graph Theory, and Computing*, pages 225–228, 1992.

15 Jan Kynčl. Enumeration of simple complete topological graphs. *European Journal of Combinatorics*, 30(7):1676–1685, 2009. doi:10.1016/j.ejc.2009.03.005.

16 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.

17 János Pach, József Solymosi, and Géza Tóth. Unavoidable configurations in complete topological graphs. *Discrete & Computational Geometry*, 30(2):311–320, 2003. doi:10.1007/s00454-003-0012-9.

18 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.

19 Andres J. Ruiz-Vargas. Many disjoint edges in topological graphs. *Computational Geometry*, 62:1–13, 2017. doi:10.1016/j.comgeo.2016.11.003.

20 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.

21 Bang Wong. Color blindness. *Nature Methods*, 8(6):441, 2011. URL: https://www.nature.com/articles/nmeth.1618.pdf.

Hamiltonian Cycles and Matchings in 1-planar Graphs[*]

Michael Hoffmann, Meghana M. Reddy, and Emanuel Seemann

Department of Computer Science, ETH Zürich, Switzerland,
`{hoffmann, meghana.mreddy}@inf.ethz.ch, emanuel.seemann@me.com`

Abstract

A graph is 1-*planar* if it can be drawn in the plane such that every edge has at most one crossing. A 1-planar graph is triangulated if it has a 1-plane drawing where every face is a triangle, i.e., every face contains either exactly three distinct vertices or exactly two distinct vertices and one crossing. We investigate the Hamiltonicity and matching properties of 1-planar graphs. We show that there always exists a Hamiltonian cycle with one prescribed edge in a 4-connected triangulated 1-planar graph. The result also holds for 4-connected maximal 1-planar graphs.

A connected graph with an even number of vertices is k-extendable if any matching of size k can be extended to a perfect matching. We prove that all 4-connected triangulated 1-planar graphs are 1-extendable and all 5-connected triangulated 1-planar graphs are 2-extendable. We also construct an infinite family of 7-connected triangulated 1-planar graphs that have a unique 1-plane drawing.

1 Introduction

Planar graphs are graphs that can be drawn in the plane without any crossings. The class of 1-planar graphs was discovered by Ringel [8] in 1961 while trying to prove the 4-color theorem. A graph is 1-planar if it has a local crossing number of one, i.e., it can be drawn in the plane such that every edge has at most one crossing. Over time, these graphs found a new home as generalizations of planar graphs. Consequently, a sizeable part of research on 1-planar graphs is concerned with generalizing properties of planar graphs. For instance, while planar graphs on n vertices can have at most $3n-6$ edges, 1-planar graphs on n vertices can have at most $4n-8$ edges. Not all results generalize this well, probably since 1-planar graphs are NP-hard to recognize [6] while their planar counterparts can be recognized in polynomial time [2].

In this article, we show generalizations of results concerning Hamiltonian cycles and matching extensions in planar graphs to 1-planar graphs. For hamiltonicity, we generalize a result by Thomassen [9], which states that 4-connected planar graphs contain a Hamiltonian cycle through any prescribed edge e. A connected graph with an even number of vertices is k-*extendable* if any matching of size k can be extended to a perfect matching. For matching extensions, we generalize results by Plummer [7] which showed that 4-connected planar graphs are 1-extendable and identified conditions under which they are 2-extendable.

A graph is called *triangulated* if there exists a drawing of the graph that is triangulated, i.e., every face is a triangle. A graph is called *internally triangulated* if there exists a drawing of the graph where every internal face is a triangle (which implies that the outer face is not necessarily a triangle). A planar (resp. 1-planar) graph is called *optimal* if it contains the maximum number of edges possible. A planar (resp. 1-planar) graph is called *maximal* if no edge can be added to the graph such that it remains planar (resp. 1-planar). A 1-plane

[*] M. Hoffmann and M. M. Reddy were supported by the Swiss National Science Foundation within the collaborative DACH project *Arrangements and Drawings* as SNSF Project 200021E-171681.

drawing D is *locally-maximal* if for each crossing of two edges, the end vertices of these edges induce a K_4 in D. A graph is called locally-maximal 1-planar if it admits such a drawing. A 1-plane drawing D is *near-optimal* if the subdrawing H of D induced by uncrossed edges contains faces of sizes three or four only, such that the vertices of faces of size four induce a K_4 in D, and no two triangular faces of H share an edge. A graph is called near-optimal 1-planar if it admits such a drawing.

2 Hamiltonian cycles

Fabrici et al. [4] proved that every 4-connected locally-maximal 1-planar graph is Hamiltonian. Independently, Biedl [1] studied Hamiltonian cycles in 4-connected 1-planar graphs and showed that there exist 4-connected 1-planar graphs without a Hamiltonian cycle. However, she proved that 4-connected 1-planar graphs that are triangulated always contain a Hamiltonian cycle. Both the theorems are a generalization of a theorem by Tutte [10], which states that 4-connected planar graphs are Hamiltonian.

▶ **Theorem 1** ([1]). *Any 4-connected triangulated 1-planar graph G has a Hamiltonian cycle.*

The main idea behind the proof by Biedl [1] is to pick a 4-connected triangulated 1-plane drawing of G, and for each crossing, one of the two edges involved in the crossing is removed. The edge that is removed is chosen such that the resulting graph remains 4-connected. Once we obtain a 4-connected plane graph, Tutte's theorem [10] can be applied to construct a Hamiltonian cycle.

We study the generalization of Theorem 1 where a prescribed edge e must be part of the Hamiltonian cycle. For 4-connected planar graphs, this generalization of Tutte's theorem is known as Thomassen's theorem [9]. If the prescribed edge is one of the edges that is removed to obtain a 4-connected plane graph, then the proof technique of Biedl [1] cannot be used to prove Hamiltonicity. Nevertheless, we can still prove that there exists a Hamiltonian cycle that contains a prescribed edge e of a 4-connected triangulated 1-planar graph.

▶ **Theorem 2.** *Let G be a 4-connected triangulated 1-planar graph. Let e be an arbitrary edge of G. Then there exists a Hamiltonian cycle in G that contains e.*

We fix a triangulated 1-plane drawing of G. The main idea is to re-use Biedl's edge-removing strategy whenever possible. If the prescribed edge e is not removed by Biedl's edge-removing strategy, the proof is trivial and follows immediately. In the case where the prescribed edge e would be removed, we need to be more careful. In this case the edge e must be involved in a crossing, say with the edge f. Instead of removing the edge e we remove the edge f, and remove all the other edges as directed by Biedl's edge-removing strategy to obtain a graph G'. At this point we can no longer guarantee that G' is 4-connected. However we can now guarantee that G' is a maximal planar graph, so every 3-cut in G' must be a separating triangle. Furthermore, we know that every separating triangle of G' contains the edge e. This motivates the following definition.

▶ **Definition 3** (Bishop's hat). Let G be a plane graph. A bishop's hat H over the edge $e = (v_1, v_2)$ is a set of vertices $\{c_1, ..., c_k\}$ such that the triangle (v_1, v_2, c_i) exists in G and is separating for all $i \in [k]$.

We call the edge e the brim of the bishop's hat and refer to each c_i as a tip. Refer to Figure 1 for an illustration. For any plane drawing of a bishop's hat H we say that H is one-sided if there is a triangle $\Delta = (v_1, v_2, c_i)$ that contains all other separating triangles

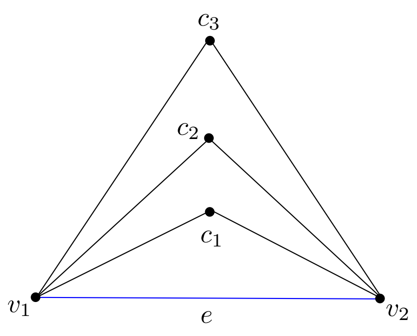


Figure 1 A bishop's hat on 3 vertices with brim e.

of H in its interior. Otherwise we say that H is two-sided. For one-sided bishop's hats we label the tips c_i in the order they appear under set inclusion, i.e. c_i is the tip such that $\Delta = (v_1, v_2, c_i)$ contains all tips c_j in its interior, where $1 \leq j < i \leq k$. For a two-sided bishop's hat, we can label the tips analogously on the two sides independently. We refer to the number of tips k of the bishop's hat as the height of the bishop's hat. We also refer to c_1 as the lowest tip and c_k as the highest tip for brevity.

Using this definition, we know that all separating triangles of G' are contained in a bishop's hat H. We prove Theorem 2 using the following theorem for maximal planar graphs.

▶ **Theorem 4.** *Let G be a maximal planar graph that is free of separating triangles apart from a bishop's hat of arbitrary height. Then G contains a Hamiltonian cycle.In particular, one can construct a Hamiltonian cycle that goes through the brim e of the bishop's hat.*

The proof of this theorem relies on tools developed by Whitney [11] and later by Chen [3].

▶ **Definition 5** ([3]). Let G be an internally triangulated plane graph and let A and B be two vertices on the outer face of an internally triangulated drawing of G. (G, A, B) is said to satisfy Whitney's condition if it satisfies both

- (W1) G has no separating triangles, and
- (W2) for the two paths $(A = a_0, a_1, ..., a_m = B)$ and $(B = b_0, b_1, ..., b_n = A)$ along the outer face there are no chords of the type (a_i, a_j) or (b_i, b_j)

▶ **Lemma 6** ([11]). *Let G be an internally triangulated plane graph and let A and B be two vertices on the outer face of an internally triangulated drawing of G. If (G, A, B) satisfies Whitney's condition, then there is a Hamiltonian path from A to B.*

▶ **Theorem 7** ([3], Theorem 6). *Let G be a 4-connected maximal planar graph. Let e and f be two edges that lie on the same face of G. Then there is a Hamiltonian cycle that goes through both e and f.*

Chen [3] proved the above theorem using Whitney's result detailed in Lemma 6. Using these tools we can prove the following lemma.

▶ **Lemma 8.** *Let G be a maximal plane graph with a one-sided bishop's hat H. Let c_i, c_{i+1} be two consecutive tips of H. Then there is path from c_i to c_{i+1} whose interior consists of all the vertices contained inside of the cycle $\{c_i, v_2, c_{i+1}, v_1\}$ in the plane drawing of G.*

We refer to the paths from Lemma 8 as local Hamiltonian paths. We prove a similar lemma for the lowest tip c_1 of the bishop's hat.

▶ **Lemma 9.** *Let G be a maximal plane graph with a one-sided bishop's hat H. Let c_1 be the lowest tip of H. Then there is path from v_2 to c_1 whose interior consists of all the vertices contained inside of the cycle $\{v_1, v_2, c_1\}$ in the plane drawing of G.*

The proofs of Lemmata 8 and 9 are deferred to the full version.

Proof of Theorem 4. As we can freely choose which face of G is considered as the outer face, it suffices to prove the theorem for one-sided bishop's hats.

Assume G has a one-sided bishop's hat of height k. Let c_k be the highest tip of this hat and $e = (v_1, v_2)$ the brim. We consider the subgraph G' of G, which is obtained from G by removing all vertices and edges contained strictly inside the bishop's hat. In G' the triangle $\Delta = (v_1, v_2, c_k)$ is a facial triangle. G' is also 4-connected, as the triangle Δ is no longer separating. Therefore we can apply Theorem 7 on the edges $e = (v_2, c_k)$ and $f = (v_1, v_2)$ and obtain a Hamiltonian cycle C that contains both the edges e and f, and all the vertices of G'. Inside the bishop's hat, we concatenate the local Hamiltonian paths obtained by applying Lemma 8 to every pair of consecutive tips, along with the path from v_2 to c_1 obtained by Lemma 9. We then create a Hamiltonian cycle in G from C by replacing the edge (v_2, c_k) with the path constructed using the local Hamiltonian paths. This cycle contains the edge $e = (v_1, v_2)$. This construction can be seen in Figure 2. ◀

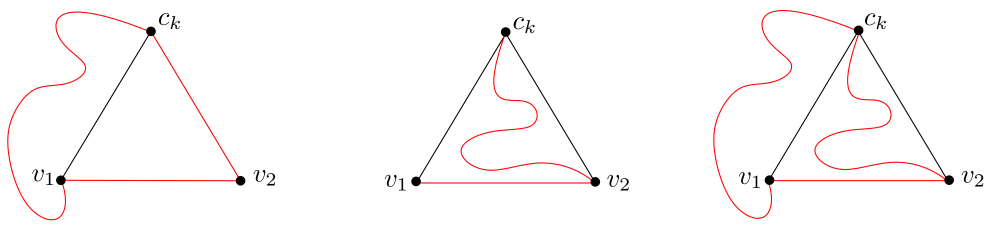


Figure 2 Building the Hamiltonian cycle through $e = (v_1, v_2)$ in G with a one-sided bishop's hat of height k. (left) the cycle C given by Chen's Theorem containing all the exterior vertices and edges (v_1, v_2) and (v_2, c_k). (middle) the concatenated local Hamiltonian paths. (right) the cycle and path combined to obtain the desired Hamiltonian cycle.

Proof of Theorem 2. We start with a 4-connected triangulated 1-plane graph G. We apply Biedl's algorithm to every crossing that does not involve e to get a 4-connected subgraph G' of G. If e is not crossed, G' is plane. In this case we can directly apply Thomassen's Theorem to G' to get a Hamiltonian cycle that contains e. If e is crossed, we instead remove the edge that crosses e. This can potentially cause G' to become 3-connected with a bishop's hat of arbitrary height with brim e. Note that all separating triangles of G' are part of this bishop's hat by construction. We apply Theorem 4 to get a Hamiltonian cycle passing through e. ◀

We can further make the following observation about maximal 1-planar graphs, which then immediately implies Corollary 11. The proof is deferred to the full version.

▶ **Lemma 10.** *Let G be a 4-connected maximal 1-planar graph. Then every 1-plane drawing of G is triangulated.*

▶ **Corollary 11.** *Let G be a 4-connected maximal 1-planar graph. Let e be an arbitrary edge of G. Then there exists a Hamiltonian cycle in G that contains e.*

3 Matching extendability

Theorem 2 immediately implies that for any given edge e of a 4-connected triangulated 1-planar graph, there exists a perfect matching that contains e.

▶ **Corollary 12.** *Every 4-connected triangulated 1-planar graph of even order is 1-extendable.*

Further using Lemma 10, we get:

▶ **Corollary 13.** *Every 4-connected maximal 1-planar graph of even order is 1-extendable.*

Plummer [7] and Fujisawa et al. [5] studied 2-extendability of planar and 1-planar graphs. Plummer [7] proved the 2-extendability of 4-connected maximal planar graphs with forbidden substructures called *generalized butterflies*. Let G be a graph, and let $e = (u, v)$, $f = (w, x)$ be two edges of G. If the graph $G \setminus \{u, v, w, x\}$ contains a component C with an odd number of vertices, the induced subgraph $G[V(C) \cup \{u, v, w, x\}]$ is called a generalized butterfly. Fujisawa et al. [5] proved that any optimal 1-planar graph of even order with no generalized butterfly is 2-extendable. Since trivially 5-connected graphs do not contain generalized butterflies, both the papers were able to extend their results to 5-connected planar graphs and 5-connected optimal 1-planar graphs, respectively. We prove that 5-connected triangulated planar graphs are 2-extendable. It is not difficult to see that optimal 1-planar graphs are triangulated, thus, our result is a strengthening of Fujisawa et al.'s result. The proof of the theorem is deferred to the full version.

▶ **Theorem 14.** *Every 5-connected triangulated 1-planar graph of even order is 2-extendable.*

4 1-planar graphs with high connectivity

Biedl [1] constructed a class of 5-connected 1-plane graphs that are non-Hamiltonian. Interestingly, these graphs contained a matching of size $\lfloor \frac{n}{2} \rfloor - 1$, which are one edge away from being a perfect matching. This motivated Biedl [1] to question whether 1-planar graphs with higher connectivity are Hamiltonian. In particular, if we can prove that 6- or 7-connected 1-planar graphs are always triangulated, then we can use techniques similar to Theorem 2 to prove hamiltonicity for these graphs. Unfortunately, we discovered that there exist 7-connected 1-planar graphs that are neither maximal nor triangulated. The counterexample suggests that new tools that do not depend on a triangulated drawing are required to address the problem of Hamiltonicity of 1-planar graphs with high connectivity.

▶ **Theorem 15.** *For each $k \in \mathbb{N}$, there exists a 7-connected triangulated near-optimal 1-planar graph with $n = 24 + 8k$ vertices. If $k \geq 3$, this graph is maximal 1-planar.*

▶ **Theorem 16.** *There exist 7-connected 1-planar graphs that are neither maximal nor triangulated.*

To this end, we first prove that there exist many 7-connected triangulated near-optimal 1-planar graphs. We then show that these graphs have many edges that can be removed such that the resulting graph remains 7-connected. We briefly describe the construction of the graph. We start with the *double stop-sign* graph given by Fabrici and Madaras, illustrated in Figure 3. Using computer-based testing, we verified that the double stop-sign graph is 7-connected. We then increase the number of 8-cycles around the central K_4. The resulting graph is called a *k-layered double stop-sign* graph, where k refers to the number of newly added 8-cyles. A 1-layered double stop-sign graph is illustrated in Figure 3. We can prove the

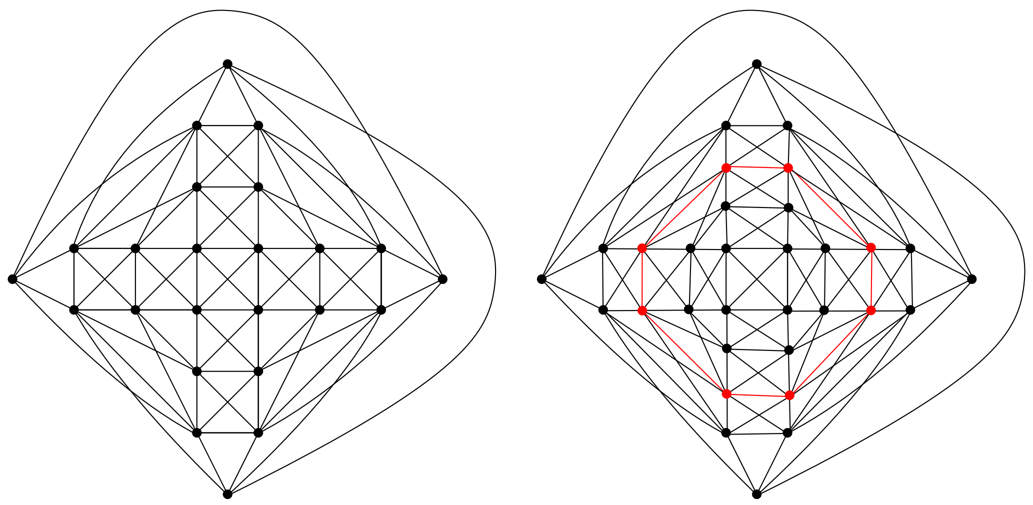


Figure 3 (left) The double stop-sign graph. (right) The 1-layered double stop-sign graph, where the new layer is depicted in red.

7-connectedness of the new graph by using the fact that the double stop-sign is 7-connected and thus there must exist seven internally vertex-disjoint paths between any two vertices, which can be extended to obtain seven new paths in the new graph. Finally, we can observe that one of the diagonal edges between two consecutive layers can be removed, and the graph remains 7-connected, thereby proving Theorem 16.

5 Conclusion

There are quite a few properties of 1-planar graphs that still need to be investigated. Based on our work, we list the most interesting open questions below.

▶ **Open Question 17.** *Let G be a 4-connected triangulated 1-planar graph that contains no generalized butterflies. Is G then 2-extendable?*

▶ **Open Question 18.** *For a 4-connected triangulated 1-planar graph, is there a Hamiltonian cycle through any two of its edges?*

▶ **Open Question 19.** *Are 6- or 7- connected 1-planar graphs Hamiltonian?*

References

1 Therese Biedl. Are highly connected 1-planar graphs hamiltonian? *CoRR*, abs/1911.02153, 2019. URL: http://arxiv.org/abs/1911.02153, arXiv:1911.02153.

2 Ulrik Brandes. The left-right planarity test. *Manuscript submitted for publication*, 2009, accessed at https://www.uni-konstanz.de/algo/publications/b-lrpt-sub.pdf on 04/08/2022.

3 Chiuyuan Chen. Any maximal planar graph with only one separating triangle is hamiltonian. *J. Comb. Optim.*, 7(1):79–86, 2003. doi:10.1023/A:1021998507140.

4 Igor Fabrici, Jochen Harant, Tomás Madaras, Samuel Mohr, Roman Soták, and Carol T. Zamfirescu. Long cycles and spanning subgraphs of locally maximal 1-planar graphs. *J. Graph Theory*, 95(1):125–137, 2020. doi:10.1002/jgt.22542.

5 Jun Fujisawa, Keita Segawa, and Yusuke Suzuki. The matching extendability of optimal 1-planar graphs. *Graphs Comb.*, 34(5):1089–1099, 2018. doi:10.1007/s00373-018-1932-6.

6 Vladimir P. Korzhik and Bojan Mohar. Minimal obstructions for 1-immersions and hardness of 1-planarity testing. *J. Graph Theory*, 72(1):30–71, 2013. doi:10.1002/jgt.21630.

7 Michael D. Plummer. Extending matchings in planar graphs IV. *Discret. Math.*, 109(1-3):207–219, 1992. doi:10.1016/0012-365X(92)90292-N.

8 Gerhard Ringel. Ein Sechsfarbenproblem auf der Kugel. In *Abhandlungen aus dem Mathematischen Seminar der Universität Hamburg*, volume 29, pages 107–117. Springer, 1965.

9 Carsten Thomassen. A theorem on paths in planar graphs. *J. Graph Theory*, 7(2):169–176, 1983. doi:10.1002/jgt.3190070205.

10 W. T. Tutte. A theorem on planar graphs. *Transactions of the American Mathematical Society*, 82(1):99–116, 1956. URL: http://www.jstor.org/stable/1992980.

11 Hassler Whitney. A theorem on graphs. *Annals of Mathematics*, 32(2):378–390, 1931. URL: http://www.jstor.org/stable/1968197.

Recognizing Unit Disk Graphs in Hyperbolic Geometry is $\exists\mathbb{R}$-Complete

Nicholas Bieker[1], Thomas Bläsius[2], Emil Dohse[3], and Paul Jungeblut[4]

1 **Karlsruhe Institute of Technology**
`bieker.nicholas@gmail.com`
2 **Karlsruhe Institute of Technology**
`thomas.blaesius@kit.edu`
3 **Karlsruhe Institute of Technology**
`emildohse@gmail.com`
4 **Karlsruhe Institute of Technology**
`paul.jungeblut@kit.edu`

Abstract

A graph G is a (Euclidean) unit disk graph if it is the intersection graph of unit disks in the Euclidean plane $\mathbb{R}^2$. Recognizing them is known to be $\exists\mathbb{R}$-complete, i.e., as hard as solving a system of polynomial inequalities. In this note we describe a simple framework to translate $\exists\mathbb{R}$-hardness reductions from the Euclidean plane $\mathbb{R}^2$ to the hyperbolic plane $\mathbb{H}^2$. We apply our framework to prove that the recognition of unit disk graphs in the hyperbolic plane is also $\exists\mathbb{R}$-complete.

Related Version *Full Version*: `https://arxiv.org/abs/2301.05550` [4]

1 Introduction

A graph is a *unit disk graph* if its vertices can be represented by unit disk such that two vertices are adjacent if and only if their corresponding disks intersect. The class of unit disk graphs (UDG) is a well studied graph class due to its mathematical beauty and its practical relevance, e.g., in the context of sensor networks.

Naturally, unit disk graphs are usually considered in the Euclidean plane $\mathbb{R}^2$. However, in the past decade, research on intersection graphs of equally sized disks in the hyperbolic plane $\mathbb{H}^2$ has gained traction (stay tuned why we have to talk about *equally sized* disks instead of *unit* disks here). Hyperbolic geometry is well suited to represent a wider range of graph structures, including complex scale-free networks with heterogeneous degree distributions [8, 9, 12, 17, 23]; see Figure 1. Most research on such graphs is driven by the network science community studying probabilistic network models, i.e., hyperbolic random graphs. However, when omitting the probability distribution and looking at hyperbolic unit disk graphs as a graph class, little is known so far.

A graph G is a *hyperbolic unit disk graph* if it is the intersection graph of equally sized disks in the hyperbolic plane $\mathbb{H}^2$. In particular, the radius of the disks may depend on G[1]. We denote by HUDG the class of hyperbolic unit disk graphs [7][2].

[1] For example, stars with up to five leaves are Euclidean unit disk graphs. On the other hand, for every star S there exists a radius r_S such that S has an intersection representation of disks with radius r_S in $\mathbb{H}^2$. However, there is no universal radius r_* such that all stars have an intersection representation of disks with radius r_* in $\mathbb{H}^2$.

[2] We note that there are earlier results on a related family of graph classes parameterized by the disk size by Kisfaludi-Bak [15]. In a sense, the class HUDG is the union of all these classes. This subtle difference is important when considering asymptotic behavior as it can be desirable to grow the disk

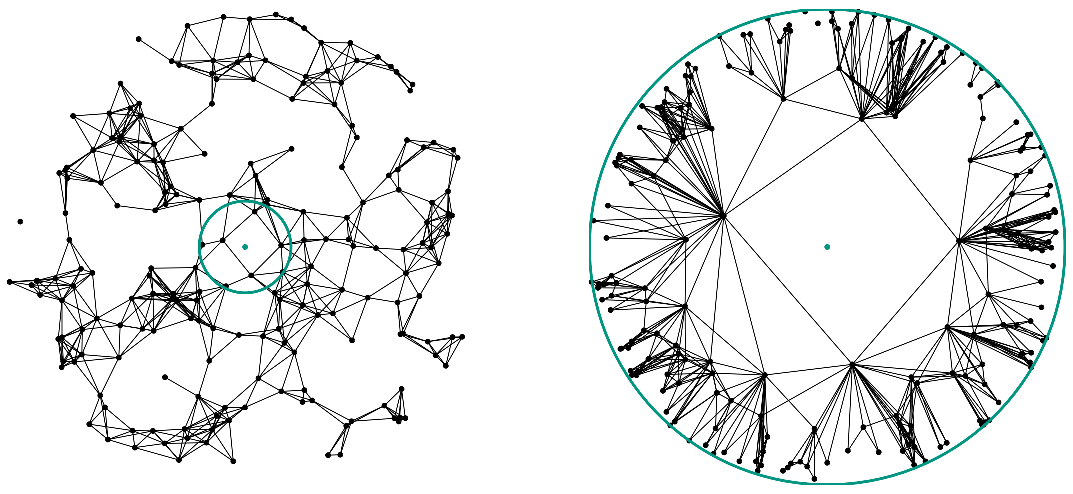


Figure 1 Two hyperbolic unit disk graphs. Thee green circle indicates the threshold distance below which vertices are connected. A small threshold (left) yields structures similar to Euclidean unit disk graphs. A large threshold (right) facilitates heterogeneous vertex degrees.

When choosing disks of small radius, the difference between Euclidean and hyperbolic geometry becomes negligible; also see our interactive visualization[3] and Figure 1.

Arguably the most fundamental algorithmic question when it comes to studying graph classes is the computational complexity of the *recognition problem*, i.e., Recog(HUDG) is the problem of testing whether a given graph is part of HUDG. In this paper we prove that Recog(HUDG) is ∃ℝ-complete. Containment in ∃ℝ is less obvious than in the Euclidean plane as distances are not (square roots of) a polynomial in hyperbolic geometry. Nonetheless, containment is easy to show when using the hyperboloid model of the hyperbolic plane. For ∃ℝ-hardness, our proof consists of five steps switching back and forth between Euclidean and hyperbolic variants of problems in a particular way. Our proof has framework-character in the sense that the first three steps are independent of the specific problem and the remaining steps can probably be translated to other problems. Thus we believe that this can be a template for proving ∃ℝ-hardness for other hyperbolic problems that have an ∃ℝ-hard Euclidean counterpart. For our framework, we in particular use the Beltrami-Klein model of the hyperbolic plane to observe that SimpleStretchability is equivalent in Euclidean and hyperbolic geometry in the sense that a pseudoline arrangement is stretchable in the Euclidean plane if and only if it is stretchable in the hyperbolic plane.

1.1 Existential Theory of the Reals

The *existential theory of the reals* is the set of all true sentences of the form $\exists X \in \mathbb{R}^n : \varphi(X)$, where $\varphi(X)$ is a quantifier-free formula consisting of polynomial equations and inequalities, e.g. $\exists X, Y \in \mathbb{R} : XY = 6 \wedge X + 2Y = 5$. We denote the decision problem whether such a sentence is true by ETR (which also stands for "existential theory of the reals") and define the complexity class ∃ℝ to contain all decision problems that polynomial-time reduce to ETR. It holds NP ⊆ ∃ℝ ⊆ PSPACE [10]. The class ∃ℝ has gained increasing attention in the computational geometry community over the last years as it exactly captures the

size with the graph size; see [7] for a detailed discussion.

[3] `https://thobl.github.io/hyperbolic-unit-disk-graph`

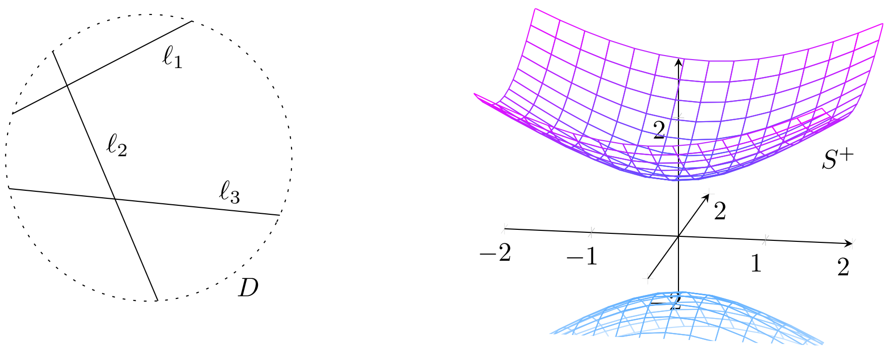


Figure 2 Left: The Beltrami-Klein disk with three hyperbolic lines. Right: The upper sheet S^+ used for the hyperboloid model (in the full version [4]).

complexity of many geometry problems like the art gallery problem [2], geometric packing [3] or the recognition of many classes of geometric intersection graphs [16, 19, 25].

1.2 Hyperbolic Geometry

The are several ways to embed the hyperbolic plane into Euclidean space. In this paper we use the *Beltrami-Klein model* and the *hyperboloid model* (the latter only in the full version [4]). In the Beltrami-Klein model the hyperbolic plane $\mathbb{H}^2$ is represented by the interior of a unit disk D in $\mathbb{R}^2$ (the boundary of D is not part of the model). The set of hyperbolic lines is exactly the set of chords of D. See Figure 2 (left).

2 Simple Stretchability in the Euclidean and the Hyperbolic Plane

An *pseudoline arrangement* $\mathcal{A}$ is a collection of *pseudolines* (x-monotone[4] curves in $\mathbb{R}^2$) such that each pair of curves intersects at most once. We assume that each pseudoline $\ell \in \mathcal{A}$ is oriented and thus divides the plane $\mathbb{R}^2$ into two open half-planes ℓ^- and ℓ^+. Further, $\mathcal{A}$ partitions the plane into *cells*, i.e., maximal connected components of $\mathbb{R}^2 \setminus \mathcal{A}$ not on any pseudoline. We say that $\mathcal{A}$ is *simple* if any two lines intersect exactly once and no three lines intersect in the same point. Given a pseudoline arrangement $\mathcal{A} = \{\ell_1, \dots, \ell_n\}$ we assign to each $p \in \mathbb{R}^2$ a *sign vector* $\sigma(p) = (\sigma_i(p))_{i=1}^n \in \{-, 0, +\}^n$, where

$$\sigma_i(p) := \begin{cases} - & \text{if } p \in \ell_i^- \\ 0 & \text{if } p \in \ell_i \\ + & \text{if } p \in \ell_i^+ \end{cases} .$$

The *combinatorial description* $\mathcal{D}$ of $\mathcal{A}$ is then given by $\{\sigma(p) \mid p \in \mathbb{R}^2\}$[5]. We say that $\mathcal{A}$ realizes $\mathcal{D}$. A pseudoline arrangement is *stretchable* if there is a line arrangement with the same combinatorial description. Not every pseudoline arrangement is stretchable and, given a combinatorial description $\mathcal{D}$, deciding whether $\mathcal{D}$ is stretchable is known as the STRETCHABILITY problem (or SIMPLESTRETCHABILITY if $\mathcal{D}$ is simple). STRETCHABILITY and SIM-

[4] Requiring the curves to be x-monotone is a well-established way to make sure a pseudoline is homeomorphic to a straight line under some homeomorphism of the plane.
[5] Pseudoline arrangements are closely related to oriented matroids of rank 3, see [6].

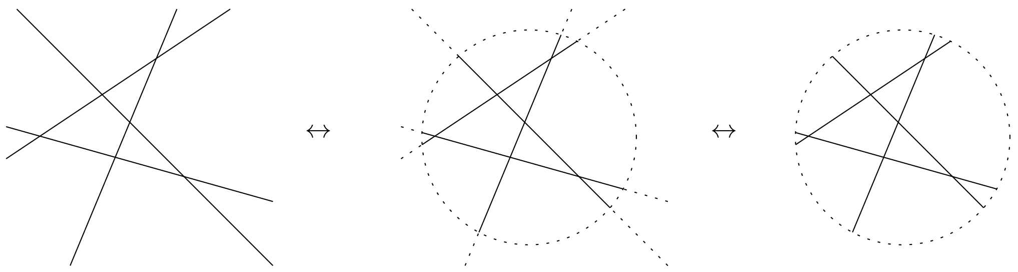


Figure 3 Transforming line arrangements between Euclidean and hyperbolic geometry.

pleStretchability are famously known to be ∃ℝ-complete [22, 24, 28]. SimpleStretchability is the starting problem for many ∃ℝ-hardness reductions, e.g. [5, 13, 16, 25, 26].

Apart from line arrangements in the Euclidean plane $\mathbb{R}^2$ one might also consider line arrangements in the hyperbolic plane $\mathbb{H}^2$. The main result of this section is that SimpleStretchability is equivalent in Euclidean and hyperbolic geometry.

▶ **Proposition 1.** *Let $\mathcal{D}$ be a combinatorial description of a simple pseudoline arrangement. Then there is a line arrangement realizing $\mathcal{D}$ in $\mathbb{R}^2$ if and only if there is one in $\mathbb{H}^2$.*

Proof. The proof is an easy application of the Beltrami-Klein model of the hyperbolic plane. Given a Euclidean line arrangement, we can obtain a hyperbolic line arrangement with the same combinatorial description and vice versa, see Figure 3.

Let $\mathcal{A}_\mathbb{R}$ be a simple line arrangement in $\mathbb{R}^2$ and D be a disk strictly enclosing all intersections of $\mathcal{A}_\mathbb{R}$. For each line in $\mathcal{A}_\mathbb{R}$, keep only its part inside D. We think of D as a unit disk and obtain a representation of a hyperbolic line arrangement in the Beltrami-Klein model.

For the other direction let $\mathcal{A}_\mathbb{H}$ be a simple hyperbolic line arrangement and take a representation inside the Beltrami-Klein disk D, so all hyperbolic lines are chords of D. Remove D and extend all chords to lines. The resulting Euclidean line arrangement has the same combinatorial description $\mathcal{D}$ because $\mathcal{A}_\mathbb{H}$ was simple: All possible intersections between two lines were already inside the Beltrami-Klein disk D. ◀

▶ Remark. Proposition 1 is only about simple (pseudo)line arrangements. There is no corresponding result for the general (non-simple) Stretchability problem: For example, given three lines $\ell_1, \ell_2, \ell_3 \subseteq \mathbb{H}^2$, lines ℓ_2 and ℓ_3 may cross each other while both being parallel to ℓ_1. However, Proposition 1 may be extended to line arrangements where each pair of lines is still required to cross but multiple lines are allowed to cross at the same point.

3 The Framework

Let $\Pi_\mathbb{R}$ be a geometric decision problem for which ∃ℝ-hardness is shown in Euclidean geometry by a polynomial-time reduction f from (Euclidean) SimpleStretchability. We denote by $\Pi_\mathbb{H}$ the corresponding decision problem obtained by considering the hyperbolic plane $\mathbb{H}^2$ instead of the Euclidean plane $\mathbb{R}^2$. Our framework below consists of several (hopefully) simple steps that allow us to prove ∃ℝ-hardness of $\Pi_\mathbb{H}$ by using the reduction for $\Pi_\mathbb{R}$:

1. Let $\mathcal{D}$ be an instance of SimpleStretchability in $\mathbb{H}^2$, i.e., a combinatorial description of a simple pseudoline arrangement.
2. Use Proposition 1 to consider $\mathcal{D}$ to be an instance of SimpleStretchability in $\mathbb{R}^2$.
3. Use the reduction f to obtain an instance $I = f(\mathcal{D})$ of Π_R equivalent to $\mathcal{D}$.

4. Prove that every yes-instance of $\Pi_{\mathbb{R}}$ is also a yes-instance of $\Pi_{\mathbb{H}}$.
5. Prove that a hyperbolic line arrangement realizing $\mathcal{D}$ can be extracted from a realization of I in $\mathbb{H}^2$.

Steps 1, 2 and 3 require no work when applying the framework.

Step 4 ensures that a stretchable instance $\mathcal{D}$ yields a yes-instance of $\Pi_{\mathbb{H}}$. This step requires to come up with a new argument but we expect it to be relatively simple because locally $\mathbb{R}^2$ and $\mathbb{H}^2$ are very similar. A promising approach is to scale a Euclidean realization of I to a tiny area and then interpret the Euclidean polar coordinates as hyperbolic ones.

Step 5 ensures correctness. By showing that a line arrangement realizing $\mathcal{D}$ can be extracted from a realization of I in $\mathbb{H}^2$ we show that a no-instance $\mathcal{D}$ maps to a no-instance of $\Pi_{\mathbb{H}}$. Reduction f might help us again here (though not as a black box as in Step 3): If we are lucky, the argument why a realization of I in $\mathbb{R}^2$ induces a Euclidean line arrangement realizing $\mathcal{D}$ only uses the axioms of *absolute geometry* (the common "subset" of Euclidean and hyperbolic geometry) and works without any adaptations for realizations in $\mathbb{H}^2$, too.

4 Recognition of Hyperbolic Unit Disk Graphs

We apply our framework to prove that RECOG(HUDG), the recognition problem of hyperbolic unit disk graphs, is $\exists\mathbb{R}$-hard. For Euclidean geometry this is shown in [14, 20, 21]. Let us recall that UDG and HUDG are not the same class: For example, star graphs with at least six leaves are hyperbolic unit disk graphs but not Euclidean ones.

For Step 1 of our framework let $\mathcal{D}$ be an instance of SIMPLESTRETCHABILITY in $\mathbb{H}^2$. We consider it to be an equivalent instance in $\mathbb{R}^2$ for Step 2. In Step 3 we use the reduction f from the literature proving that RECOG(UDG) in $\mathbb{R}^2$ is $\exists\mathbb{R}$-hard [14, 20, 21]. We obtain a graph $G_{\mathcal{D}}$ that is a Euclidean unit disk graph if and only if $\mathcal{D}$ is stretchable.

Though not required for the framework, let us shortly summarize the reduction f to construct $G_{\mathcal{D}}$ from $\mathcal{D}$ as given in [20]. Let n be the number of pseudolines $\ell_1, \dots, \ell_n$ and $m = 1 + \binom{n+1}{2}$ be the number of cells $C_1, \dots, C_m$. The arrangement described by $\mathcal{D}$ has exactly this number of cells because it is simple. We define $G_{\mathcal{D}}$ to be the graph with vertex set $V = A \cup B \cup C$ for $A = \{a_1, \dots, a_n\}$, $B = \{b_1, \dots, b_n\}$ and $C = \{c_1, \dots, c_m\}$. Here we assume that vertex c_i corresponds to cell C_i. For the edges, each of the sets A, B, C forms a clique. Further, each $a_i \in A$ (for $i \in \{1, \dots, n\}$) is connected to c_j (for $j \in \{1, \dots, m\}$) if and only if $C_j \in \ell_i^-$. Similarly, each $b_i \in B$ is connected to c_j if and only if $C_j \in \ell_i^+$.

For Step 4 we have to show that every Euclidean unit disk graph is also a hyperbolic unit disk graph. This has recently been proven by Bläsius, Friedrich, Katzmann and Stephan:

▶ **Lemma 2** ([7]). *Every Euclidean unit disk graph is also a hyperbolic one, so* UDG $\subseteq$ HUDG.

As foreshadowed above, the proof scales a Euclidean unit disk intersection representation to a tiny area until the Euclidean and hyperbolic plane are "similar enough". Then the polar coordinates in $\mathbb{R}^2$ can be used as polar coordinates in $\mathbb{H}^2$ without changing any adjacencies.

For Step 5 it remains to prove how a line arrangement realizing $\mathcal{D}$ in $\mathbb{H}^2$ can be extracted from a realization of $G_{\mathcal{D}}$ in $\mathbb{H}^2$.

▶ **Lemma 3** (adapted from [20, Lemma 1]). *Given a realization of $G_{\mathcal{D}}$ as the intersection graph of equally sized disks in $\mathbb{H}^2$. Then the line arrangement $L = \{\ell_1, \dots, \ell_n\}$ defined by*

$$\ell_i := \{p \in \mathbb{H}^2 \mid \mathrm{d}(p, a_i) = \mathrm{d}(p, b_i)\}$$

has combinatorial description $\mathcal{D}$. Here $\mathrm{d}(\cdot, \cdot)$ denotes the hyperbolic distance.

Proof. The proof is exactly the same as the proof of Lemma 1 in [20] where McDiarmid and Müller prove that taking the perpendicular bisectors of the segments between any pair of points a_i and b_i yields a Euclidean line arrangement realizing $\mathcal{D}$. Their argument works in $\mathbb{H}^2$ by just replacing Euclidean distances with hyperbolic distances.

Let us note that the ℓ_i are indeed hyperbolic lines, as easily seen in the Beltrami-Klein model of the hyperbolic plane: Given two points a and b inside the unit disk D, their Euclidean perpendicular bisector is a line ℓ. The chord of D on ℓ is the hyperbolic perpendicular bisector (and in particular a line as lines in $\mathbb{H}^2$ and chords of D are in bijection). ◀

At this point we proved ∃ℝ-hardness of Recog(HUDG). To get ∃ℝ-completeness we prove ∃ℝ-membership in the full version of this paper [4]. The idea is to use the hyperboloid model of the hyperbolic plane, where distances can be computed by taking the arcosh(·) (a monotone function) of a polynomial. We conclude with the following theorem:

▶ **Theorem 4.** *Recognizing hyperbolic unit disk graphs is ∃ℝ-complete.*

5 Conclusion and Outlook

We presented a simple framework that allows us to translate ∃ℝ-hardness reductions for geometric decision problems in $\mathbb{R}^2$ into reductions for their counterparts $\mathbb{H}^2$. As an application we proved that Recog(HUDG) is ∃ℝ-complete. Promising candidates for further applications of our framework are the recognition of unit ball graphs (i.e., a generalization of our result to higher dimensions) as already done in $\mathbb{R}^d$ in [14] or Recog(CONV), the recognition problem for intersection graphs of convex sets (Euclidean reduction is in [25]).

Technically, the framework also works for the recognition problems Recog(HSEG) and Recog(HDISK), where (H)SEG and (H)DISK denote the classes of intersection graphs of (hyperbolic) segments and disks, respectively (Euclidean reductions are in [14, 16, 19, 21, 25]). However, these are not really interesting as SEG = HSEG (easy to see in the Beltrami-Klein model) and DISK = HDISK (easy to see in the Poincaré model, not considered here). Therefore ∃ℝ-completeness for Recog(HSEG) and Recog(HDISK) follows directly from the Euclidean cases. Other interesting problems to consider in $\mathbb{H}^2$ are linkage realizability [1, 26], simultaneous graph embeddings [11, 18] or RAC-drawings [27].

Acknowledgements We thank Torsten Ueckerdt for discussion on proving the membership of Recog(HUDG) in ∃ℝ.

References

1 Zachary Abel, Erik D. Demaine, Martin L. Demaine, Sarah Eisenstat, Jayson Lynch, and Tao B. Schardl. Who Needs Crossings? Hardness of Plane Graph Rigidity. In Sándor Fekete and Anna Lubiw, editors, *32nd International Symposium on Computational Geometry (SoCG 2016)*, volume 51 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 3:1–3:15, 2016. doi:10.4230/LIPIcs.SoCG.2016.3.

2 Mikkel Abrahamsen, Anna Adamaszek, and Tillmann Miltzow. The Art Gallery Problem is ∃ℝ-complete. *Journal of the ACM*, 69(1):1–70, 2022. doi:10.1145/3486220.

3 Mikkel Abrahamsen, Tillmann Miltzow, and Nadja Seiferth. Framework for ER-Completeness of Two-Dimensional Packing Problems. In *61st Annual Symposium on Foundations of Computer Science (FOCS 2020)*, pages 1014–1021, 2020. doi:10.1109/FOCS46700.2020.00098.

4 Nicholas Bieker, Thomas Bläsius, Emil Dohse, and Paul Jungeblut. Recognizing Unit Disk Graphs in Hyperbolic Geometry is ∃ℝ-Complete, 2023. arXiv:2301.05550.

5 Daniel Bienstock. Some Provably Hard Crossing Number Problems. *Discrete & Computational Geometry*, 6(3):443–459, 1991. doi:10.1007/BF02574701.

6 Anders Björner, Michel Las Vergnas, Bernd Sturmfels, Neil White, and M. Ziegler, Günter. *Oriented Matroids*. Cambridge University Press, 2nd edition, 1999. doi:10.1017/CBO9780511586507.

7 Thomas Bläsius, Tobias Friedrich, Maximilian Katzmann, and Daniel Stephan. Strongly hyperbolic unit disk graphs, 2022. To appear at STACS 2023. arXiv:2107.05518.

8 Thomas Bläsius, Tobias Friedrich, and Anton Krohmer. Cliques in hyperbolic random graphs. *Algorithmica*, 80:2324–2344, 2018. doi:10.1007/s00453-017-0323-3.

9 Michel Bode, Nikolaos Fountoulakis, and Tobias Müller. On the largest component of a hyperbolic model of complex networks. *Electronic Journal of Combinatorics*, 22(1–52):P3.24, 2015. URL: https://www.combinatorics.org/ojs/index.php/eljc/article/view/v22i3p24.

10 John Canny. Some Algebraic and Geometric Computations in PSPACE. In *Proceedings of the Twentieth Annual ACM Symposium on Theory of Computing*, STOC '88, pages 460–467, 1988. doi:10.1145/62212.62257.

11 Jean Cardianl and Vincent Kusters. The Complexity of Simultaneous Geometric Graph Embedding. *Journal of Graph Algorithms and Applications*, 19(1):259–272, 2015. doi:10.7155/jgaa.00356.

12 Luca Gugelmann, Konstantinos Panagiotou, and Ueli Peter. Random hyperbolic graphs: Degree sequence and clustering. In *International Colloquium on Automata, Languages, and Programming (ICALP*, pages 573–585, 2012. doi:10.1007/978-3-642-31585-5_51.

13 Udo Hoffmann. On the Complexity of the Planar Slope Number Problem. *Journal of Graph Algorithms and Applications*, 21(2):183–193, 2017. doi:10.7155/jgaa.00411.

14 Ross J. Kang and Tobias Müller. Sphere and Dot Product Representations of Graphs. *Discrete & Computational Geometry*, 47(3):548–568, 2012. doi:10.1007/s00454-012-9394-8.

15 Sándor Kisfaludi-Bak. Hyperbolic Intersection Graphs and (Quasi)-Polynomial Time. In *Proceedings of the 2020 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1621–1638, 2020. doi:10.1137/1.9781611975994.100.

16 Jan Kratochvíl and Jiří Matoušek. Intersection Graphs of Segments. *Journal of Combinatorial Theory, Series B*, 62(2):289–315, 1994. doi:10.1006/jctb.1994.1071.

17 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.

18 Jan Kynčl. Simple Realizability of Complete Abstract Topological Graphs in P. *Discrete & Computational Geometry*, 45(3):383–399, 2011. doi:10.1007/s00454-010-9320-x.

19 Jiří Matoušek. Intersection graphs of segments and ∃ℝ, 2014. arXiv:1406.2636.

20 Colin McDiarmid and Tobias Müller. The Number of Bits Needed to Represent a Unit Disk Graph. In Dimitrios M. Thilikos, editor, *Graph Theoretic Concepts in Computer Science (WG 2010)*, volume 6410 of *Lecture Notes in Computer Science*, pages 315–323, 2010. doi:10.1007/978-3-642-16926-7_29.

21 Colin McDiarmid and Tobias Müller. Integer realizations of disk and segment graphs. *Journal of Combinatorial Theory, Series B*, 103(1):114–143, 2013. doi:10.1016/j.jctb.2012.09.004.

22 Nikolai E. Mnëv. The Universality Theorems on the Classification Problem of Configuration Varieties and Convex Polytopes Varieties. In Oleg Y. Viro and Anatoly M Vershik, editors,

Topology and Geometry — Rohlin Seminar, volume 1346 of *Lecture Notes in Mathematics*, pages 527–543. Springer, Berlin, Heidelberg, 1988. doi:10.1007/BFb0082792.

23 Tobias Müller and Merlijn Staps. The diameter of KPKVB random graphs. *Advances in Applied Probability*, 51(2):358–377, 2019. doi:10.1017/apr.2019.23.

24 Jürgen Richter-Gebert. Mnëv's Universality Theorem revisited, 2002. URL: https://geo.ma.tum.de/_Resources/Persistent/3/e/a/2/3ea2ad59228a1a24a67d1e994fa77266a599e73a/15_MnevsUniversalityhTheorem.pdf.

25 Marcus Schaefer. Complexity of Some Geometric and Topological Problems. In David Eppstein and Emden R. Gansner, editors, *Graph Drawing (GD 2009)*, volume 5849 of *Lecture Notes in Computer Science*, pages 334–344, 2010. doi:10.1007/978-3-642-11805-0_32.

26 Marcus Schaefer. Realizability of Graphs and Linkages. In János Pach, editor, *Thirty Essays on Geometric Graph Theory*, pages 461–482. Springer, 2013. doi:10.1007/978-1-4614-0110-0_24.

27 Marcus Schaefer. RAC-Drawability is ∃ℝ-Complete. In Helen C. Purchase and Ignaz Rutter, editors, *Graph Drawing and Network Visualization (GD 2021)*, volume 12868 of *Lecture Notes in Computer Science*, pages 72–86, 2021. doi:10.1007/978-3-030-92931-2_5.

28 Peter W. Shor. Stretchability of Pseudolines is NP-Hard. In Peter Gritzmann and Bernd Sturmfels, editors, *Applied Geometry And Discrete Mathematics, Proceedings of a DIMACS Workshop, Providence, Rhode Island, USA, September 18, 1990*, volume 4 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, pages 531–554, 1991. doi:10.1090/dimacs/004/41.

Crossing Minimization in Time Interval Storylines[*]

Alexander Dobler, Martin Nöllenburg, Daniel Stojanovic, Anaïs Villedieu, and Jules Wulms

Algorithms and Complexity Group, TU Wien
`{adobler|noellenburg|avilledieu|jwulms}@ac.tuwien.ac.at, e11907566@student.tuwien.ac.at`

Abstract

Storyline visualizations are a popular way of visualizing characters and their interactions over time: Characters are drawn as x-monotone curves and interactions are visualized through close proximity of the corresponding character curves in a vertical strip. Existing methods to generate storylines assume a total ordering of the interactions, although real-world data often do not contain such a total order. Instead, multiple interactions are often grouped into coarser time intervals such as years. We exploit this grouping property by introducing a new model called storylines with time intervals and present two methods to minimize the number of crossings and horizontal space usage. We then evaluate these algorithms on a small benchmark set to show their effectiveness.

Related Version arXiv:2302.14213

1 Introduction

Storyline visualizations are a popular way of visualizing characters and their interactions through time. They were popularized by Munroe's xkcd comic [13] (see Fig. 1 for a storyline describing a movie as a series of scenes through time, in which the characters participate). A character is drawn using an x-monotone curve, and the vertical ordering of the character curves varies from left to right. A scene is represented by closely gathering the curves of characters involved in said scene at the relevant spot on the x-axis, which represents time. Storylines attracted significant interest in visualization research, especially the question of designing automated methods to create storylines adhering to certain quality criteria [12, 14, 15].

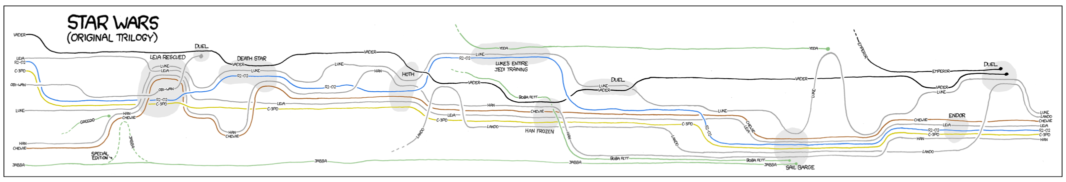


Figure 1 The xkcd comic showing a storyline of the Star Wars movie.

While different design optimization goals can be specified, most theoretical research has been focused on crossing minimization [8, 11] and variants like block crossing minimization [16, 17]. This problem is NP-hard [11, 16] and is commonly solved using ILP and SAT formulations [8, 17]; it has many similarities with the metro line crossing minimization problem [1–3, 5]. Recently a new model for storylines was proposed by Di Giacomo et al. [7] that allows for one character to be part of multiple interactions at the same point in time, by modeling each character as a tree rather than a curve. Using this model, it is possible to

[*] Alexander Dobler was supported by the Vienna Science and Technology Fund (WWTF) [10.47379/ICT19035]. Anaïs Villedieu was supported by the Austrian Science Fund (FWF) under grant P31119. We thank Martin Gronemann for the initial discussion.

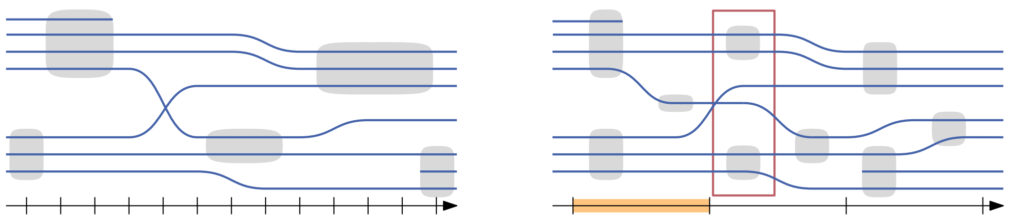


Figure 2 (a) A classic storyline with blue character lines. Interactions are shown in gray, they happen on specific timestamps and have a duration. **(b)** A time interval storyline. The horizontal orange segment shows a slice, every interaction on this segment has the same timestamp. A layer is highlighted in red, containing two interactions with the same timestamp but not sharing a character.

represent data sets which have a more loosely defined ordering of interactions. Furthermore, authorship networks have been a popular application for storylines visualizations [7, 10]. In this paper we introduce *time interval* storylines, an alternative approach to visualize data sets with less precise temporal attributes. In the time interval model, a set of discrete, totally ordered timestamps is given, which serve to label disjoint time intervals (e.g., the timestamp 2021 represents all interactions occurring between January and December of the year 2021). Each interval is represented in a storyline as a horizontal section in which all interactions with the same timestamp occur. The horizontal ordering within this section, however, does not correspond to a temporal ordering anymore (see Fig. 2). For example, an authorship network often sorts publications by year. In a traditional storyline model, the complete temporal ordering of the interactions must be provided. Previous models like the one by van Dijk et al. [17] can place multiple disjoint interactions in the same vertical layer, but the assignment of interactions to the totally ordered set of layers must be given as input. Unlike the traditional model, we have no pre-specified assignment of interactions to layers, but interactions with the same timestamp can be assigned to any layer within the time interval of this timestamp.

Problem setting. We are given a triple $\mathcal{S} = (\mathcal{C}, \mathcal{I}, T)$, of characters $\mathcal{C} = \{c_1, \ldots, c_n\}$, interactions $\mathcal{I} = \{I_1, \ldots, I_m\}$, and totally ordered timestamps $T = \{t_1, \ldots, t_p\}$ as input. Each interaction $(C_j, t) = I_j \in \mathcal{I}$ consists of a set $C_j \subseteq \mathcal{C}$ of characters involved in the interaction and a timestamp $t \in T$ at which the interaction I_j occurred, respectively denoted by $\texttt{char}(I_j) = C_j$ and $\texttt{time}(I_j) = t$. A subset of interactions can form a *layer* ℓ, when for every pair of interactions I, I' in ℓ, $\texttt{time}(I) = \texttt{time}(I')$. A time interval storyline is composed of a sequence of layers to which interactions are assigned. Intuitively, a layer represents a column in the storyline visualization, in which interactions are represented as vertical stacks. Thus, to each layer we associate a vertical ordering of $\mathcal{C}$. Consider the set S containing all interactions with timestamp t, we call the union of layers containing S a *slice*.

Characters are represented with curves passing through each layer at most once. To represent an interaction $I = (C, t)$ in a layer ℓ, the ordering of the characters in ℓ must be such that the characters of C appear consecutively in that ordering. For a pair I, I' of interactions in the same layer, it must hold that $\texttt{char}(I) \cap \texttt{char}(I') = \emptyset$.

For a layer ℓ, we denote the set of interactions by $\texttt{inter}(\ell)$ and the timestamp of a layer by $\texttt{time}(\ell)$ (with slight abuse of notation). We focus on combinatorial storylines, as opposed to geometric storylines, meaning that our algorithm should output a (horizontal) ordering $o_L(\mathcal{S})$ of layers, and for each layer ℓ, a (vertical) ordering $o_c(\ell)$ of the characters, and all interactions must occur in some layer. For two interactions I, I' such that $\texttt{time}(I) < \texttt{time}(I')$, let ℓ and ℓ' be the layers of I and I', respectively. Then ℓ must be before ℓ' in $o_L(\mathcal{S})$. A character is *active* in a layer if it appears in the character ordering for that layer. A character must be active in a contiguous range of layers including the first and last interaction it is involved in. A character is active in a layer if it appears in the character ordering for that layer.

Contributions. In this paper we introduce the time interval storylines model, as well as two methods to compute layer and character orderings. In Section 2.1 we introduce an algorithmic pipeline based on ILP formulations and heuristics that computes time interval storylines. We further present an ILP formulation that outputs a crossing-minimal time interval storyline in Section 2.2. Lastly in Section 3, we experimentally evaluate our pipeline and ILP formulation. Due to space constraints, some details are omitted and can be found in the full version of the paper [4].

2 Computing combinatorial storylines

2.1 A pipeline heuristic

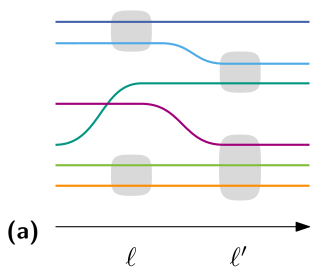
As the traditional storyline crossing minimization problem is a restricted version of the time interval formulation, our problem is immediately NP-hard [11]. Thus, we first aim to design an efficient heuristic to generate time interval storylines, which consists of the following stages.

(i) Initially, we assign each interaction to a layer,
(ii) then, we compute a horizontal ordering $o_L(\mathcal{S})$ of the layers obtained in step (i), and
(iii) finally, we compute a vertical ordering $o_c(\ell)$ of the characters for each layer $\ell \in o_L(\mathcal{S})$.

For step (i), the assignment is obtained using graph coloring. For each $t \in T$, we create a conflict graph $G_t = (\mathcal{I}_t, E)$ where $\mathcal{I}_t \subseteq \mathcal{I}$ and $I \in \mathcal{I}_t$ if and only if $\texttt{time}(I) = t$. Two interactions are connected by an edge if they share at least one character. Each color class then corresponds to a set of interactions which share no characters and can appear together in a layer. We solve this problem using a straightforward ILP formulation based on variables $x_{v,c} = 1$ if color c is assigned to vertex v and 0 otherwise. We can choose to limit the size of each color class by adding an upper bound on the number of interactions assigned to each color, which forces fewer interactions per layer. While this allows us to limit the height of each slice, it likely results in more layers.

To compute a horizontal ordering of the layers in step (ii), we use a traveling salesperson (TSP) model. Concretely, for the slice corresponding to the timestamp t, we create a complete weighted graph $G = (\mathcal{L}, E)$, where $\mathcal{L}$ corresponds to all the layers ℓ such that $\texttt{time}(\ell) = t$. For each edge e between a pair of layers ℓ and ℓ' in $\mathcal{L}$, we associate a weight w_e, estimating the number of crossings that may occur if the two layers are consecutive as follows.

Minimizing the crossings of the curves representing the characters is NP-complete [6, 11], thus we propose two heuristics to estimate the number of crossings. First, we propose to use set similarity measures to describe how similar the interactions in two layers ℓ and ℓ' are: If ℓ and ℓ' both have an interaction that contains the same set of characters, then no crossing should be induced by the curves corresponding to those characters, when these two layers are consecutive (see Fig. 3a). Second, we consider pattern matching methods that guess how many crossings could be induced by a certain ordering of the characters. There are certain patterns of interactions between two layers for which a crossing is unavoidable (see Fig. 3b). We count how many of these patterns occur between each pair of layers in G and set the weight of the corresponding edge to that crossing count. More details on these heuristics can be found in the full version [4].



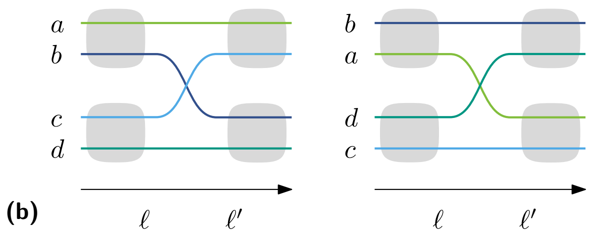


Figure 3 (a) The orange and light green characters are together in two interactions, which increases similarity between ℓ and ℓ', but the two blue characters are once together and once apart, which decreases similarity. **(b)** An example of an unavoidable crossing pattern.

To finish step (ii), we solve the path formulation of the TSP problem on G and find a horizontal ordering of the layers for each time slice. We have now obtained a traditional storyline, in which each interaction belongs to a specific layer, and all layers are totally ordered. Thus, we can solve step (iii) using the state-of-the-art crossing minimization ILP by Gronemann et al. [8].

We call the pipeline variants $\mathrm{P_s}$ and $\mathrm{P_p}$, when using the set similarity heuristic and the pattern matching heuristic in step (ii), respectively.

2.2 An ILP formulation

Crossing minimization in storylines is generally solved using ILP formulations [8, 16]. We propose two formulations to handle slices, which build on the ideas of Gronemann et al. [8]. Both formulations will give us an assignment of interactions to layers, that are already totally ordered, and an ordering of characters per layer. For each timestamp $t \in T$, let $\mathcal{L}_t$ be a set of $|\{I \mid \mathtt{time}(I) = t\}|$ layers corresponding the number of interactions at t, and let $\mathcal{L} = \bigcup_{t \in T} \mathcal{L}_t$. In the first formulation we assume that a character c is active in all layers between the first timestamp and last timestamp, inclusively, where there exists an interaction I such that $c \in \mathtt{char}(I)$. In the second formulation we will introduce additional variables that model whether a character really needs to be active, since, in fact, character curves do not need to be active before their first interaction or after their last interaction. In contrast to the pipeline approach, the presented ILP formulations are able to find the crossing-minimal solution for the explored search space.

First formulation. Let $\mathcal{C}_\ell$ be the characters that appear in layer $\ell \in \mathcal{L}$, as discussed before. First we introduce for each $t \in T$ the binary variables $y_{\ell,I}$ for $\ell \in \mathcal{L}_t$ and $I \in \mathcal{I}$ where $\mathtt{time}(I) = t$. These should be one iff interaction I is assigned to layer ℓ. This is realized by constraints of type (1). If two different interactions I and I' share a character they cannot be in the same layer, realized by type (2) constraints.

$$\sum_{\ell \in \mathcal{L}_t} y_{\ell,I} = 1 \qquad t \in T, I \in \mathcal{I}, \mathtt{time}(I) = t \tag{1}$$

$$y_{\ell,I} + y_{\ell,I'} \leq 1 \qquad \mathtt{time}(I) = \mathtt{time}(I') = t, \mathtt{char}(I) \cap \mathtt{char}(I') \neq \emptyset, \ell \in \mathcal{L}_t \tag{2}$$

Next we introduce binary ordering variables x_{ℓ,c_i,c_j} for each layer $\ell \in \mathcal{L}$ and $c_i, c_j \in \mathcal{C}_\ell$ with $i < j$. Variable x_{ℓ,c_i,c_j} should be one iff c_i comes before c_j on layer ℓ. Standard transitivity constraints (3) (see e.g. [9]) ensure that the binary variables induce a total order.

$$0 \leq x_{\ell,c_h,c_i} + x_{\ell,c_i,c_j} - x_{\ell,c_h,c_j} \leq 1 \qquad c_i, c_j, c_h \in \mathcal{C}_\ell, i < j < h \tag{3}$$

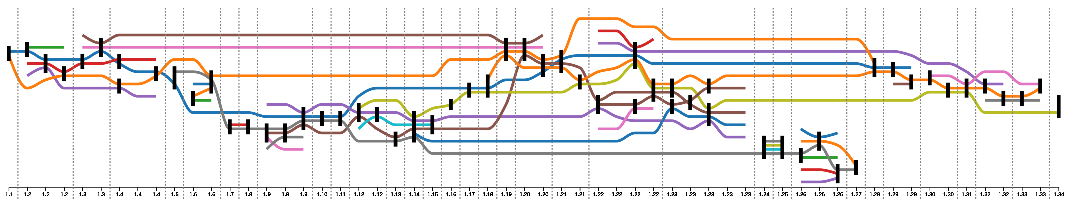


Figure 4 A storyline for a dataset corresponding to the first chapter of *Anna Karenina* created by ILP1. The x-axis is labeled by the scenes of the book, which are separated by dashed gray lines and correspond to the timestamps. Interactions are visualized with black vertical bars and correspond to the characters in the book interacting with each other shown as x-monotone curves. The storyline contains 58 layers, 34 timestamps, and 23 crossings.

The crux is now to model the assignment of some interaction I to some layer ℓ, linking the x- and y-variables together. This is done with so-called tree-constraints [8]: Let $\ell \in \mathcal{L}_t$, $I \in \mathcal{I}$ with $\texttt{time}(I) = t$ and $c_i, c_j, c_k \in \mathcal{C}_\ell$ such that $i < j$, $c_i, c_j \in \texttt{char}(I)$, and $c_k \notin \texttt{char}(I)$. If $i < j < k$ we add constraints (4) and (5), which ensure that c_k is either before or after both c_i and c_j. We elaborate on the analogous cases $k < i < j$ and $i < k < j$ in the full version [4].

$$x_{\ell,c_i,c_k} \leq x_{\ell,c_j,c_k} + 1 - y_{\ell,I} \tag{4}$$

$$x_{\ell,c_j,c_k} \leq x_{\ell,c_i,c_k} + 1 - y_{\ell,I} \tag{5}$$

Lastly, to optimize the number of crossings we have to provide an objective function. For this we introduce binary variables z_{ℓ,c_i,c_j} for all layers ℓ but the rightmost one and all $c_i, c_j \in \mathcal{C}_\ell \cap \mathcal{C}_{\ell'}$ where ℓ' is the adjacent layer of ℓ to the right. Variable z_{ℓ,c_i,c_j} should be one iff the character lines of c_i and c_j cross between layers ℓ and ℓ'. Linking variables z_{ℓ,c_i,c_j} is done by introducing the constraints corresponding to setting $z_{\ell,c_i,c_j} \geq x_{\ell,c_i,c_j} \oplus x_{\ell',c_i,c_j}$ (see the full version [4]) where $x \oplus y$ denotes the exclusive-or relation of two binary variables x and y. The objective is then to simply minimize $\sum z_{\ell,c_i,c_j}$. A solution to the ILP model is then transformed into a storyline realization of the input. We call this formulation ILP1. Figure 4 shows a storyline visualization that was computed with ILP1 and a simple post-processing method that assigns x- and y-coordinates (refer to the full version [4] for more information).

Extensions and Second Formulation In the above formulation we have one layer for each interaction, which does not utilize the potential of having multiple interactions in one layer. We can, however, minimize the number of layers beforehand, using the graph coloring problem as in Section 2.1. If we need q colors for timestamp t, we let $\mathcal{L}_t$ only consist of q layers. This can of course result in more crossings in the end. We call this adapted formulation ILP1ML.

Additionally, in the above model a character was contained in all layers of the first and last timestamp that contains an interaction, in which the character appears. By introducing further variables and adapting the constraints given in ILP1, this can be relaxed, so that a character's active range actually only spans from the first to the last interaction it appears in. This new formulation is given in the full version [4] and is referred to as ILP2. We can then introduce the fewest possible number of layers as above, resulting in formulation ILP2ML.

3 Evaluation

We evaluated our six algorithms on seven instances that were used in previous work on storylines. The number of layers that could be saved by the coloring pipeline-step, the number of crossings, and the runtimes are reported in the full paper [4] together with the complete description of instances, experimental setup, and evaluation. It also contains visualizations of storylines produced by our algorithms. Generally, the layer-minimization step of the pipeline, using graph coloring, reduces the number of layers for all but one instance, in one case even by 50%. The ILP-formulations perform better than the pipeline-approaches w.r.t. crossings, if they do not time out (3600 s). And even if they time out, the best feasible solution found by the solver is sometimes better than the solution provided by the pipeline-approaches. The ILP-approaches perform far worse w.r.t. runtime and three of the four instances timed out for all ILP-approaches. It has to be mentioned that, by construction, optimal solutions for ILP2 will always have the least crossings, and optimal solutions for ILP2ML will always have fewer crossings than all other algorithms minimizing layers. We think though, that our ILP-formulations can be further optimized for scalability.

References

1 Matthew Asquith, Joachim Gudmundsson, and Damian Merrick. An ILP for the metro-line crossing problem. In *Proc. 14th Symposium on Computing: Australasian Theory Symposium (CATS)*, volume 77 of *CRPIT*, pages 49–56, 2008.

2 Michael A. Bekos, Michael Kaufmann, Katerina Potika, and Antonios Symvonis. Line crossing minimization on metro maps. In Seok-Hee Hong, Takao Nishizeki, and Wu Quan, editors, *Proc. 15th International Symposium on Graph Drawing (GD)*, volume 4875 of *LNCS*, pages 231–242. Springer, 2007. doi:10.1007/978-3-540-77537-9_24.

3 Marc Benkert, Martin Nöllenburg, Takeaki Uno, and Alexander Wolff. Minimizing intra-edge crossings in wiring diagrams and public transportation maps. In M. Kaufmann and D. Wagner, editors, *Proc. 14th International Symposium on Graph Drawing (GD)*, volume 4372 of *LNCS*, pages 270–281. Springer-Verlag, 2007. doi:10.1007/978-3-540-70904-6_27.

4 Alexander Dobler, Martin Nöllenburg, Daniel Stojanovic, Anaïs Villedieu, and Jules Wulms. Crossing minimization in time interval storylines, 2023. doi:10.48550/ARXIV.2302.14213.

5 Martin Fink and Sergey Pupyrev. Metro-line crossing minimization: Hardness, approximations, and tractable cases. In Stephen Wismath and Alexander Wolff, editors, *Proc. 22nd International Symposium on Graph Drawing and Network Visualization (GD)*, volume 8242 of *LNCS*, pages 328–339. Springer-Verlag, 2013. doi:10.1007/978-3-319-03841-4_29.

6 M. R. Garey and D. S. Johnson. Crossing number is NP-complete. *SIAM. J. Alg. Discr. Meth.*, 4(3):312–316, 1983. doi:10.1137/0604033.

7 Emilio Di Giacomo, Walter Didimo, Giuseppe Liotta, Fabrizio Montecchiani, and Alessandra Tappini. Storyline visualizations with ubiquitous actors. In David Auber and Pavel Valtr, editors, *Proc. 28th International Symposium on Graph Drawing and Network Visualization (GD)*, volume 12590 of *LNCS*, pages 324–332. Springer, 2020. doi:10.1007/978-3-030-68766-3_25.

8 Martin Gronemann, Michael Jünger, Frauke Liers, and Francesco Mambelli. Crossing minimization in storyline visualization. In Yifan Hu and Martin Nöllenburg, editors, *Proc. 24th International Symposium on Graph Drawing and Network Visualization (GD)*, volume 9801 of *LNCS*, pages 367–381. Springer, 2016. doi:10.1007/978-3-319-50106-2_29.

9 Martin Grötschel, Michael Jünger, and Gerhard Reinelt. Facets of the linear ordering polytope. *Math. Program.*, 33(1):43–60, 1985. doi:10.1007/BF01582010.

10 Tim Herrmann. Storyline-Visualisierungen für wissenschaftliche Kollaborationsgraphen. Master's thesis, University of Würzburg, 2022.

11 Irina Kostitsyna, Martin Nöllenburg, Valentin Polishchuk, André Schulz, and Darren Strash. On minimizing crossings in storyline visualizations. In Emilio Di Giacomo and Anna Lubiw, editors, *Proc. 23rd International Symposium on Graph Drawing and Network*

Visualization (GD), volume 9411 of *LNCS*, pages 192–198. Springer, 2015. doi:10.1007/978-3-319-27261-0_16.

12 Shixia Liu, Yingcai Wu, Enxun Wei, Mengchen Liu, and Yang Liu. Storyflow: Tracking the evolution of stories. *IEEE Trans. Vis. Comput. Graph.*, 19(12):2436–2445, 2013. doi:10.1109/TVCG.2013.196.

13 Randall Munroe. Movie Narrative Charts, November 2009. URL: https://xkcd.com/657/.

14 Michael Ogawa and Kwan-Liu Ma. Software evolution storylines. In Alexandru C. Telea, Carsten Görg, and Steven P. Reiss, editors, *Proc. ACM 5th Symposium on Software Visualization (SOFTVIS)*, pages 35–42. ACM, 2010. doi:10.1145/1879211.1879219.

15 Yuzuru Tanahashi and Kwan-Liu Ma. Design considerations for optimizing storyline visualizations. *IEEE Trans. Vis. Comput. Graph.*, 18(12):2679–2688, 2012. doi:10.1109/TVCG.2012.212.

16 Thomas C. van Dijk, Martin Fink, Norbert Fischer, Fabian Lipp, Peter Markfelder, Alexander Ravsky, Subhash Suri, and Alexander Wolff. Block crossings in storyline visualizations. *J. Graph Algorithms Appl.*, 21(5):873–913, 2017. doi:10.7155/jgaa.00443.

17 Thomas C. van Dijk, Fabian Lipp, Peter Markfelder, and Alexander Wolff. Computing storyline visualizations with few block crossings. In Fabrizio Frati and Kwan-Liu Ma, editors, *Proc. 25th International Symposium on Graph Drawing and Network Visualization (GD)*, volume 10692 of *LNCS*, pages 365–378. Springer, 2017. doi:10.1007/978-3-319-73915-1_29.

The Complexity of Intersection Graphs of Lines in Space and Circle Orders

Jean Cardinal

Université libre de Bruxelles (ULB)
`jean.cardinal@ulb.be`

Abstract

We consider the complexity of the recognition problem for two families of combinatorial structures. A graph $G = (V, E)$ is said to be an intersection graph of lines in space if every $v \in V$ can be mapped to a straight line $\ell(v)$ in $\mathbb{R}^3$ so that vw is an edge in E if and only if $\ell(v)$ and $\ell(w)$ intersect. A partially ordered set $(X, \prec)$ is said to be a circle order, or a 2-space-time order, if every $x \in X$ can be mapped to a closed circular disk $C(x)$ so that $y \prec x$ if and only if $C(y)$ is contained in $C(x)$. We prove that the recognition problems for intersection graphs of lines and circle orders are both $\exists\mathbb{R}$-complete, hence polynomial-time equivalent to deciding whether a system of polynomial equalities and inequalities has a solution over the reals. The second result addresses an open problem posed by Brightwell and Luczak.

1 Introduction

The complexity class $\exists\mathbb{R}$ is the set of decision problems that are polynomial-time reducible to deciding the validity of a sentence in the existential theory of the reals. This class was popularized by Schaefer and Stefankovic [32, 35], and since then many problems have been classified as $\exists\mathbb{R}$-complete [7]. The class $\exists\mathbb{R}$ contains NP, hence $\exists\mathbb{R}$-complete problems are NP-hard. It is also known that $\exists\mathbb{R} \subseteq$ PSPACE [6].

Among known $\exists\mathbb{R}$-complete problems, we find many computational geometry problems in which we are asked to decide whether some combinatorial structure has a geometric realization, hence recognition problems for combinatorial families defined by geometric constructions. This includes, among others, realizable oriented matroids and order types [26, 27], linkages [33], simultaneously embeddable planar graphs [9, 34], and several classes of geometric intersection graphs, including segment intersection graphs [22, 23] and disk intersection graphs [19, 24]. More recently, other fundamental problems in computational geometry have been shown to be $\exists\mathbb{R}$-complete, such as the art gallery problem [1], and several families of geometric packing problems [2].

In this paper, we prove the $\exists\mathbb{R}$-completeness of the recognition problem for two families of combinatorial objects: (i) intersection graphs of lines in 3-space, and (ii) circle orders.

1.1 Lines in space

Arrangements of lines in 3-space are a classical topic in discrete geometry [14, 11, 29]. We consider the problem of deciding, given a graph G, whether we can we map each of its vertices to a line in $\mathbb{R}^3$ so that two vertices are adjacent if and only if the two corresponding lines intersect. Such graphs will be referred to as *intersection graphs of lines in space.*

Pach, Tardos, and Tóth studied complements of intersection graphs of lines in space, and proved that graphs of maximum degree three and line graphs (adjacency graphs of edges of a graph) are intersection graphs of lines in space [28]. They also considered the computational complexity of various combinatorial optimization problems on these graphs. Davies [12] showed that intersection graphs of lines in space are not χ-bounded: There exist

39th European Workshop on Computational Geometry, Barcelona, Spain, March 29–31, 2023.
This is an extended abstract of a presentation given at EuroCG'23. 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.

such graphs that have arbitrary large girth and chromatic number. On the other hand, Cardinal, Payne, and Solomon showed that they satisfy a nice Erdős-Hajnal property: They either have a clique or an independent set of size $\Omega(n^{1/3})$ [10].

Regarding the recognition problem, two closely related results to ours have been proved. First, Matoušek and Kratochvíl showed that intersection graphs of line segments in the plane are $\exists\mathbb{R}$-complete [22]. This also holds for the more restricted classes of outersegment, grounded segments, and ray intersection graphs [8]. Evans, Rzazewski, Saeedi, Shin, and Wolff [15] proved that intersection graphs of line segments in $\mathbb{R}^3$ are $\exists\mathbb{R}$-complete as well. We prove that we only need lines for this to hold.

▶ **Theorem 1.1.** *Deciding whether a graph is an intersection graph of lines in space is $\exists\mathbb{R}$-complete.*

1.2 Geometric containement orders

Geometric containment orders are posets of the form $(X, \subset)$, where X is a countable set of (usually convex) subsets of $\mathbb{R}^d$, for some constant d, and $\subset$ is the usual containment relation. Geometric containment orders are well-studied [18]. The classical *Dushnik-Miller dimension* of a poset can be defined in terms of geometric containment orders, as the smallest d such that the poset is isomorphic to a containment order of translates of the negative orthant in $\mathbb{R}^d$ [42]. *Circle orders* are containment orders of closed disks in the plane, and more generally, *d-sphere orders* are containment orders of closed balls in $\mathbb{R}^d$ [39, 17, 5]. The corresponding notion of dimension is the *Minkowski dimension* [25].

The definition of d-sphere orders is motivated by causal set theory in physics and the notion of space-time order [3, 31, 13, 40]. In 1992, Scheinerman [37] proved that circle orders are also one-to-one with parabola orders and Loewner orders for 2×2 Hermitian matrices, and gave a general equivalence statement for *Q/L orders*, an algebraically defined class of posets.

In a survey on the combinatorics of the causal set approach to quantum gravity [4], Brightwell and Luczak pose the problem of recognizing circle orders in those terms: "It is remarkable that this concrete question is apparently not easy to answer computationally; to the best of our knowledge, the complexity of determining whether a given finite partial order is (for instance) a circle order is unknown." We prove that the problem is $\exists\mathbb{R}$-complete. We recall that a poset is *bipartite* if it can be partitioned into two antichains, or subsets of pairwise incomparable elements.

▶ **Theorem 1.2.** *Deciding whether a poset is a circle order is $\exists\mathbb{R}$-complete, even when the input is restricted to bipartite posets.*

Outline. Theorem 1.1 is proved in Section 2, and Theorem 1.2 is proved in Section 3. Our proofs build on the classical complexity-theoretic interpretation of Mnëv's Universality Theorem [26, 27], and on recent hardness results from Kang and Müller [20], and Felsner and Scheucher [16]. Note that in both cases, containment in $\exists\mathbb{R}$ is easy to prove, and we concentrate on the hardness part.

Acknowledgments. The proof of Theorem 1.1 is based on an idea due to Udo Hoffmann. The author wishes to thank Till Miltzow and the EuroCG referees for their comments on a preliminary version.

2 $\exists\mathbb{R}$-Completeness of intersection graphs of lines

Our proof is by reduction from the affine realizability problem for matroids of rank 3. In short, we are given a ground set E together with a collection $\mathcal{I}$ of subsets of E defining the maximal independent sets of the matroid. In rank 3, the maximal independent sets have size three, hence $\mathcal{I}$ is a collection of triples. We wish to decide whether there exists a map $f : E \to \mathbb{R}^2$ such that for every triple $a, b, c \in E$, the points $f(a), f(b), f(c)$ form a nondegenerate triangle if and only if $\{a, b, c\} \in \mathcal{I}$. It is somehow a folklore result that this problem is $\exists\mathbb{R}$-complete, and a detailed proof was given recently by Kim, de Mesmay, and Miltzow [21].

Given a graph $G = (V, E)$, a *line realization* of G is a map ℓ from V to the set of lines in $\mathbb{R}^3$ such that $vw \in E \Leftrightarrow \ell(v) \cap \ell(w) \neq \emptyset$. For a subset $S \subseteq V$, we use the notation $\ell(S) = \{\ell(v) : v \in S\}$. The following observation will be useful.

▶ Observation 1. In a line realization of a complete graph on the vertex set V, the lines of $\ell(V)$ intersect in one point or are coplanar.

We also use the following statement, which allows us to force a collection of lines in a line realization to lie in a plane.

▶ **Lemma 2.1.** *Let G_n be the complete graph on $2n$ vertices minus a perfect matching. For $n > 3$ all lines of a line realization ℓ of G_n are coplanar.*

Proof. Let (A, B) be the bipartition of the vertices of G_n into two cliques of size n, and consider a line realization ℓ of G_n. We show that the lines of $\ell(A)$ are coplanar. For contradiction, suppose otherwise. Then from Observation 1, all lines of A must intersect in one point p. Furthermore, there are three vertices $a_1, a_2, a_3 \in A$ such that the lines $\ell(a_1), \ell(a_2), \ell(a_3) \in \ell(A)$ are not coplanar. There is also a vertex $b \in B$ adjacent to a_1, a_2 and a_3 in G_n, but not adjacent to a fourth vertex $a_4 \in A$. Hence the line $\ell(b)$ must intersect $\ell(a_1), \ell(a_2)$ and $\ell(a_3)$ but avoid $\ell(a_4)$. The only possibility to intersect all three lines is in p, but then $\ell(b)$ also intersects $\ell(a_4)$, a contradiction. By symmetry, all lines in $\ell(B)$ must be coplanar as well, and lying in the same plane as those in $\ell(A)$. ◀

Proof of Theorem 1.1. We reduce from realizability of a rank-3 matroid $M = (E, \mathcal{I})$, as defined above. It is convenient to let $E = \{1, 2, \ldots, n\}$, and consider the set system $\mathcal{S}$ on E defined as the set of rank-2 flats of M. In a realization of M, these flats correspond to all inclusionwise maximal subsets of collinear points. Note that $|\mathcal{S}| \leq \binom{n}{2}$.

We start by considering a line realization of the graph G_n defined in Lemma 2.1. We consider the bipartition of the vertices of G_n into the two cliques $A = \{a_1, \ldots, a_n\}$ and $B = \{b_1, \ldots, b_n\}$, where a_i is adjacent to b_j if $i \neq j$. Then for each set $C \in \mathcal{S}$, we add a new vertex v_C and an edge $v_C a_i$ for each $i \in C$. We let H be the resulting graph, defined on the vertex set $A \cup B \cup \{v_C : C \in \mathcal{S}\}$. We claim that H is a intersection graph of lines if and only if the input matroid M is realizable.

To construct a realization of M from a line realization ℓ of H we use planar point-line duality on the lines in $\ell(A)$. From Lemma 2.1, the lines of $\ell(A \cup B)$ all lie in a plane P. For every set $C \in \mathcal{S}$, the vertices $\{v_C\} \cup \{a_i : i \in C\}$ form a maximal clique in H. From Observation 1, this clique must be realized by an intersection point common to all lines $\{\ell(a_i) : i \in C\}$, since otherwise $\ell(v_C)$ lies in P and intersects some lines in $\ell(B)$. Point-line duality in P maps lines with a common intersection point to collinear points. We denote by $p(a_i)$ the point that is dual to the line $\ell(a_i)$ in P. Therefore, the dual points $\{p(a_i) : i \in C\}$ are collinear in P. From the maximality of the clique $\{v_C\} \cup \{a_i : i \in C\}$, the points $\{p(a_i) : i \in C\}$ form a maximal collinear set, and $p(A)$ is a realization of M in P.

On the other hand, a realization of M in the plane can be mapped to a dual arrangement of lines. Let $A = \{a_i : i \in E\}$, and $\ell(a_i)$ be the line representing element i, for every $i \in E$. Now each line $\ell(b_j) \in \ell(B)$ can be represented by a line parallel to $\ell(a_j)$ and intersecting all the other lines $\ell(a_i)$ and $\ell(b_i)$ for $i \neq j$. We embed this realization into a plane P in $\mathbb{R}^3$ and add a line $\ell(v_C)$ for each $C \in \mathcal{S}$ that pierces P in the common intersection point of the lines $\{a_i : i \in C\}$ (see Figure 1). This yields a line realization of H. ◀

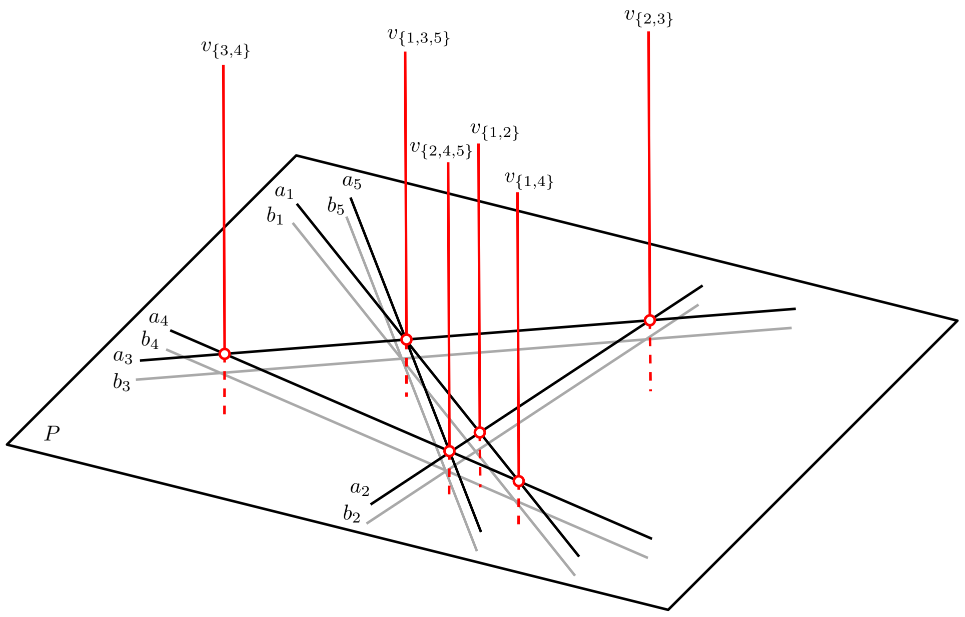


Figure 1 A line realization of the graph H constructed from a realizable rank-3 matroid on 5 elements with rank-2 flats $\mathcal{S} = \{\{1,2\},\{2,3\},\{3,4\},\{1,4\},\{1,3,5\},\{2,4,5\}\}$.

3 ∃ℝ-Completeness of circle orders

A *pseudoline arrangement* is a collection of x-monotone curves that pairwise intersect exactly once. A pseudoline arrangement is *simple* if no triple of lines intersect in the same point, and *stretchable* if it is isomorphic to an arrangement of straight lines, see Figure 2. A well-known consequence of Mnëv's Theorem – see Shor [38], and Richter-Gebert [30] – is that deciding whether a given pseudoline arrangement is stretchable is ∃ℝ-complete.

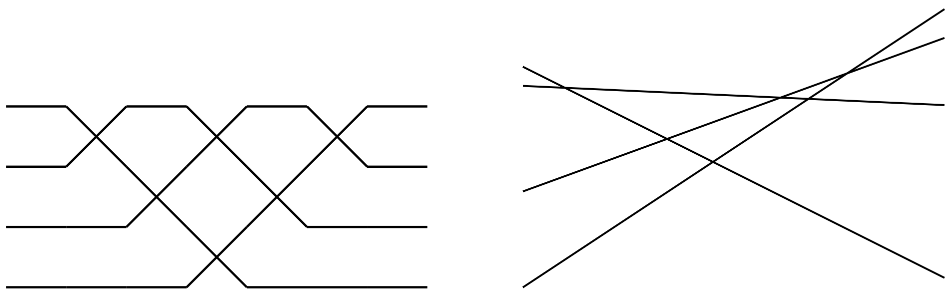


Figure 2 A pseudoline arrangement and an isomorphic line arrangement.

In order to deal with circle orders, we make use of recent analogous results on arrangement of circles due to Kang and Müller [20], and Felsner and Scheucher [16]. An arrangement of Jordan curves is said to be a *pseudocircle arrangement* if every pair of curves is either disjoint or intersects in exactly two points. A triple of pseudocircles is said to form a *Krupp* if they pairwise intersect in distinct points and their interiors have a common point. A pseudocircle arrangement is said to be a *great-pseudocircle arrangement* if every triple of pseudocircles forms a Krupp. See Figure 3 for illustrations. Similarly to pseudoline arrangements, we are only concerned with the topological structure of the arrangement. Great-pseudocircle arrangements constitute a combinatorial abstraction of arrangements of great circles on a sphere, hence of line arrangements in the projective plane. In fact, great-pseudocircle arrangements and simple pseudoline arrangements are essentially the same combinatorial family, see Felsner and Scheucher [16].

▶ **Lemma 3.1.** *Great-pseudocircle arrangements are one-to-one with simple pseudoline arrangements.*

Every great-pseudocircle arrangement realizes two copies of a pseudoline arrangement, see Figure 3. When stretchable, they correspond to realizations of pseudolines as great circles on the 2-sphere, or equivalently as lines in the projective plane.

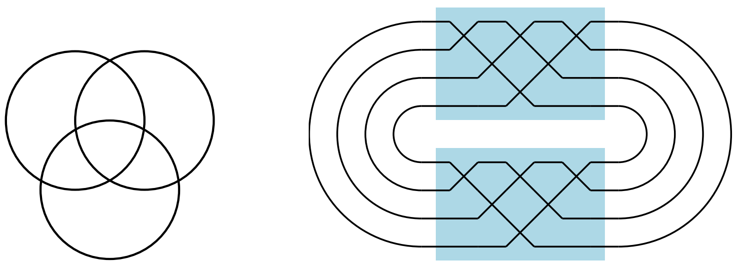


Figure 3 A Krupp, and a great-pseudocircle arrangement realizing two copies of the pseudoline arrangement of Figure 2.

We need a technical lemma on the structure of circle arrangements. A collection of n circles in the plane partitions the plane into connected subsets that we refer to as *cells*. To each cell, one can assign a binary string of length n indicating, for each circle, whether the cell lies in the interior or the exterior of the circle. We refer to this string as the *label* of the cell.

▶ **Lemma 3.2.** *The maximum number of dictinct cell labels of an arrangement of n circles is equal to the maximum number of cells of a circle arrangement, and is $n(n-1)+2$. This bound is attained by great-pseudocircle arrangements.*

Proof. One can proceed by induction on n. We let $f(n)$ be the maximum number of labels, with $f(1) = 2$. When adding the nth circle to the arrangement, the number of additional labels is at most the number of newly created cells. This number is in turn bounded by the number of new intersection points: One can charge every new cell to the intersection point on its left in a clockwise sweep around the new circle. Therefore $f(n) \leq f(n-1) + 2(n-1)$, solving to the announced bound. From this reasoning, tight examples must be such that no two cells have the same label, and one can check that this bound is attained by great-pseudocircle arrangements. ◀

The *circularizability* problem is the problem of deciding whether a pseudocircle arrangement is isomorphic to a circle arrangement. It is the analogue of the stretchability problem for pseudoline arrangements, and is in fact polynomial-time equivalent to it.

▶ **Theorem 3.3** (Kang and Müller [20], Felsner and Scheucher [16]). *Deciding whether a pseudocircle arrangement is circularizable is $\exists\mathbb{R}$-complete, even when the input is restricted to great-pseudocircle arrangements.*

With these tools at hand, we now prove our second main result.

Proof of Theorem 1.2. We reduce from the circularizability problem. Given a great-pseudocircle arrangement $\mathcal{C}$, we construct a bipartite poset $P = (X, \prec)$ that is a circle order if and only if $\mathcal{C}$ is circularizable. We define the ground set X of P as $X := C \cup F$, where C is the set of great-pseudocircles of $\mathcal{C}$, and F is the set of cell labels of $\mathcal{C}$. Both C and F induce empty posets in P, and for a pair $(c, f) \in C \times F$, we let $c \prec f$ if and only if the cell corresponding to f lies in the disk bounded by c. We show that P is a circle order if and only if $\mathcal{C}$ is circularizable.

Indeed, if $\mathcal{C}$ is circularizable, then we can consider a circle realization of $\mathcal{C}$ and add a small circle within each cell of the arrangement to realize P. For the other direction, suppose P has a realization as a circle order and consider the arrangement $\mathcal{D}$ induced by C in this realization. Since every $f \in F$ has an associated circle as well, every cell label of $\mathcal{C}$ is realized in $\mathcal{D}$. From Lemma 3.2, the number of cells is as large as can be for an arrangement of circles, hence no other cell can appear in $\mathcal{D}$. Therefore $\mathcal{D}$ must be a circularization of $\mathcal{C}$. ◀

Note that a similar proof was given by Tanenbaum, Goodrich, and Scheinerman [41] to prove hardness of point-halfspace incidence orders. Also note that from a theorem of Scheinerman [36], the vertex-edge incidence poset of a graph is a circle order if and only if the graph is planar. Hence for vertex-edge incidence posets, the problem reduces to planarity testing and can therefore be solved in linear time.

References

1 Mikkel Abrahamsen, Anna Adamaszek, and Tillmann Miltzow. The art gallery problem is $\exists\mathbb{R}$-complete. *J. ACM*, 69(1):4:1–4:70, 2022.

2 Mikkel Abrahamsen, Tillmann Miltzow, and Nadja Seiferth. Framework for ER-completeness of two-dimensional packing problems. In Sandy Irani, editor, *61st IEEE Annual Symposium on Foundations of Computer Science, FOCS 2020, Durham, NC, USA, November 16-19, 2020*, pages 1014–1021. IEEE, 2020.

3 Graham Brightwell and Ruth Gregory. Structure of random discrete spacetime. *Phys. Rev. Lett.*, 66:260–263, Jan 1991.

4 Graham Brightwell and Malwina Luczak. The mathematics of causal sets. In *Recent Trends in Combinatorics*, volume 159 of *The IMA Volumes in Mathematics and its Applications.* Springer, 2016.

5 Graham Brightwell and Peter Winkler. Sphere orders. *Order*, 6:235–240, 1989.

6 John Canny. Some algebraic and geometric computations in PSPACE. In *Symposium on Theory of Computing (STOC)*, pages 460–467. ACM, 1988.

7 Jean Cardinal. Computational geometry column 62. *SIGACT News*, 46(4):69–78, 2015.

8 Jean Cardinal, Stefan Felsner, Tillmann Miltzow, Casey Tompkins, and Birgit Vogtenhuber. Intersection graphs of rays and grounded segments. *J. Graph Algorithms Appl.*, 22(2):273–295, 2018.

9 Jean Cardinal and Vincent Kusters. The complexity of simultaneous geometric graph embedding. *J. Graph Algorithms Appl.*, 19(1):259–272, 2015.

10 Jean Cardinal, Michael S. Payne, and Noam Solomon. Ramsey-type theorems for lines in 3-space. *Discret. Math. Theor. Comput. Sci.*, 18(3), 2016.

11 Bernard Chazelle, Herbert Edelsbrunner, Leonidas J. Guibas, Micha Sharir, and Jorge Stolfi. Lines in space: Combinatorics and algorithms. *Algorithmica*, 15(5):428–447, 1996.

12 James Davies. Box and segment intersection graphs with large girth and chromatic number. *Advances in combinatorics*, 2021.

13 Fay Dowker. Introduction to causal sets and their phenomenology. *General Relativity and Gravitation*, 45(9):1651–1667, 2013.

14 Herbert Edelsbrunner. Lines in space-a collection of results. In Jacob E. Goodman, Richard Pollack, and William Steiger, editors, *Discrete and Computational Geometry: Papers from the DIMACS Special Year*, volume 6 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, pages 77–94. DIMACS/AMS, 1990.

15 William S. Evans, Pawel Rzazewski, Noushin Saeedi, Chan-Su Shin, and Alexander Wolff. Representing graphs and hypergraphs by touching polygons in 3D. In Daniel Archambault and Csaba D. Tóth, editors, *Graph Drawing and Network Visualization - 27th International Symposium, GD 2019, Prague, Czech Republic, September 17-20, 2019, Proceedings*, volume 11904 of *Lecture Notes in Computer Science*, pages 18–32. Springer, 2019.

16 Stefan Felsner and Manfred Scheucher. Arrangements of pseudocircles: On circularizability. In Therese C. Biedl and Andreas Kerren, editors, *Graph Drawing and Network Visualization - 26th International Symposium, GD 2018, Barcelona, Spain, September 26-28, 2018, Proceedings*, volume 11282 of *Lecture Notes in Computer Science*, pages 555–568. Springer, 2018.

17 Peter C. Fishburn. Interval orders and circle orders. *Order*, 5:225–234, 1988.

18 Peter C. Fishburn and William T. Trotter. Geometric containment orders: A survey. *Order*, 15(2):167–182, 1998.

19 Ross J. Kang and Tobias Müller. Sphere and dot product representations of graphs. *Discret. Comput. Geom.*, 47(3):548–568, 2012.

20 Ross J. Kang and Tobias Müller. Arrangements of pseudocircles and circles. *Discret. Comput. Geom.*, 51(4):896–925, 2014.

21 Eunjung Kim, Arnaud de Mesmay, and Tillmann Miltzow. Representing matroids over the reals is $\exists\mathbb{R}$-complete. *ArXiv e-prints*, 2023.

22 Jan Kratochvíl and Jirí Matousek. Intersection graphs of segments. *J. Comb. Theory, Ser. B*, 62(2):289–315, 1994.

23 Jiří Matoušek. Intersection graphs of segments and $\exists\mathbb{R}$. *ArXiv e-prints*, 2014.

24 Colin McDiarmid and Tobias Müller. Integer realizations of disk and segment graphs. *J. Comb. Theory, Ser. B*, 103(1):114–143, 2013.

25 David A. Meyer. Spherical containment and the Minkowski dimension of partial orders. *Order*, 10:227–237, 1993.

26 Nicolai E. Mnëv. Varieties of combinatorial types of projective configurations and convex polyhedra. *Dokl. Akad. Nauk SSSR*, 283(6):1312–1314, 1985.

27 Nicolai E. Mnëv. The universality theorems on the classification problem of configuration varieties and convex polytopes varieties. In *Topology and geometry – Rohlin seminar*, pages 527–543. Springer, 1988.

28 János Pach, Gábor Tardos, and Géza Tóth. Disjointness graphs of segments in the space. *Comb. Probab. Comput.*, 30(4):498–512, 2021.

29 Orit E. Raz. Configurations of lines in space and combinatorial rigidity. *Discret. Comput. Geom.*, 58(4):986–1009, 2017.

30 Jürgen Richter-Gebert. Mnëv's universality theorem revisited. In *Proceedings of the Séminaire Lotharingien de Combinatoire*, pages 211–225, 1995.

31 David P. Rideout and Rafael D. Sorkin. Classical sequential growth dynamics for causal sets. *Phys. Rev. D*, 61:024002, Dec 1999.

32 Marcus Schaefer. Complexity of some geometric and topological problems. In David Eppstein and Emden R. Gansner, editors, *Graph Drawing, 17th International Symposium, GD 2009, Chicago, IL, USA, September 22-25, 2009. Revised Papers*, volume 5849 of *Lecture Notes in Computer Science*, pages 334–344. Springer, 2009.

33 Marcus Schaefer. Realizability of graphs and linkages. In *Thirty Essays on Geometric Graph Theory*. Springer, 2012.

34 Marcus Schaefer. On the complexity of some geometric problems with fixed parameters. *J. Graph Algorithms Appl.*, 25(1):195–218, 2021.

35 Marcus Schaefer and Daniel Stefankovic. Fixed points, Nash equilibria, and the existential theory of the reals. *Theory Comput. Syst.*, 60(2):172–193, 2017.

36 Edward R. Scheinerman. A note on planar graphs and circle orders. *SIAM J. Discret. Math.*, 4(3):448–451, 1991.

37 Edward R. Scheinerman. The many faces of circle orders. *Order*, 9(4):343–348, 1992.

38 Peter W. Shor. Stretchability of pseudolines is NP-hard. *Applied Geometry and Discrete Mathematics–The Victor Klee Festschrift*, 4:531–554, 1991.

39 Jeffrey B. Sidney, Stuart J. Sidney, and Jorge Urrutia. Circle orders, N-gon orders and the crossing number. *Order*, 5:1–10, 1988.

40 Sumati Surya. The causal set approach to quantum gravity. *Living Reviews in Relativity*, 22(1), 2019.

41 Paul J. Tanenbaum, Michael T. Goodrich, and Edward R. Scheinerman. Characterization and recognition of point-halfspace and related orders. In Roberto Tamassia and Ioannis G. Tollis, editors, *Graph Drawing, DIMACS International Workshop, GD'94, Princeton, New Jersey, USA, October 10-12, 1994, Proceedings*, volume 894 of *Lecture Notes in Computer Science*, pages 234–245. Springer, 1994.

42 William T. Trotter. *Combinatorics and Partially Ordered Sets: Dimension Theory*. Johns Hopkins Studies in Nineteenth C Architecture Series. Johns Hopkins University Press, 1992.

Interactive Exploration of the Temporal α-Shape

Felix Weitbrecht[1]

1 **Universität Stuttgart, Germany**
weitbrecht@fmi.uni-stuttgart.de

Abstract

An interesting subcomplex of the Delaunay triangulation are α-shapes, which give a more detailed representation of the shape of point sets than the convex hull. We extend an algorithm [3, 4] which computes all Delaunay simplices over all time windows to also compute the temporal α-shape, which is a description of all α-shapes over all time windows and all values of α, in output-sensitive linear time. We present an interactive demo application based on a fast query data structure. Experimental results show that our algorithm can be used on real-world data sets and achieves a speedup of at least ~52 over the approach of [1].

Related Version https://arxiv.org/abs/2303.00878

1 Introduction

A common property of point sets to examine is their shape. The convex hull comes to mind, but a more general way to formalize the notion of shape are α-shapes [2], which generalize the convex hull to allow concavities, holes and disjointness, see Figure 1. If points are associated with timestamps it is also interesting to examine α-shapes of time windows. In interactive visualization applications, freshly computing the α-shape for every request seems wasteful and slow. If multiple α-shapes or time windows are required, a precomputed query data structure may be more efficient, like in [1], which considers storm events in the United States and visualizes their α-shape for various time windows. This approach examines all pairs of points which are close enough to admit an α-ball through their (potential) edge, so α needs to be fixed in advance and precomputation only works quickly if the value of α is chosen small enough. We show how to compute the temporal α-shape, i.e. all α-shapes over all time windows and over all values of α, in output-sensitive linear time in any fixed dimension d, based on a temporal Delaunay enumeration algorithm [3, 4]. Unlike [1], the runtime of our approach does not depend on the value(s) of α as it considers all values of α at once while still being ~52 times faster for some α-values.

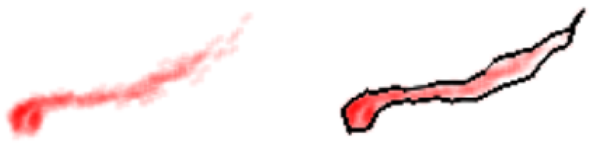


Figure 1 A point set representing a particle swarm and an α-shape describing its shape.

2 Preliminaries

We are given a point set $P = \{p_1, \dots, p_n\} \subset \mathbb{R}^d$ in general position. Point indices represent the temporal order and a point p_i exists only at time i. We say $P_{i,j}$ for $\{p_i, p_{i+1}, \dots, p_j\}$. We consider d fixed, so we refer to d-simplices as simplices and to $(d-1)$-simplices as faces.

If f is a face of a simplex s, we call s a coface of f. We consider the d-dimensional α-shape, i.e. that set of faces f with vertices in P for which an α-ball (an open hypersphere of radius α) exists which passes through the vertices of f and contains no points of P in its interior.

The Delaunay triangulation $DT(P)$ is that unique subdivision of the convex hull which consists only of d-simplices whose open circumhyperspheres contain no points of P in their interior. $T_{i,j}$ is the Delaunay triangulation of $P_{i,j}$ and T is the set of all Delaunay d-simplices occurring over all $T_{i,j}$. It can be shown that all α-faces are also Delaunay faces (Figure 2), and that every Delaunay face is an α-face for a value range of α determined by its cofaces.

The Delaunay enumeration algorithm of [3, 4] computes T in time $\mathcal{O}(|T|)$. With minor modifications it provides face-simplex associations in a well-structured order, as well as simplex lifetimes. When speaking of faces in the following, we mean faces appearing in T.

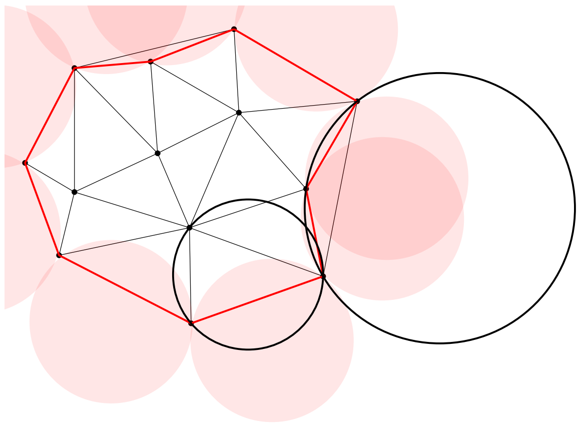


Figure 2 A point set and its Delaunay triangulation in black, and an α-shape in red.

3 Activity Spaces of Delaunay Triangulations and α-Shapes

In settings where we have geometric elements which are part of some structure (like the Delaunay triangulation or the α-shape) in certain time windows, the set of time windows for which an element is part of that structure is called its activity space, and it can be visualized in activity space diagrams, like it is done in two dimensions in [1]. To accommodate different values of α, we introduce a third dimension to activity spaces. We will now study how the activity spaces of α-faces are related to those of Delaunay simplices and Delaunay faces so we can explain exactly what is encompassed by the temporal α-shape.

3.1 Activity Spaces of Delaunay Simplices

A simplex s is a Delaunay simplex in a time window $[i, j]$ iff *(a)* all vertices of s have point indices between i and j and *(b)* the open circumsphere C of s contains no points of $P_{i,j}$. Let i_{lower} and i_{upper} be the lowest and highest point indices among the vertices of s, then *(a)* is fulfilled iff $i \leq i_{\text{lower}} \wedge i_{\text{upper}} \leq j$. For *(b)*, consider the points inside C which come before $p_{i_{\text{lower}}}$, i.e. $C \cap P_{1,i_{\text{lower}}-1}$. Let i_{before} be the largest point index among them, or $-\infty$ if no such points exist. Now *(b)* is fulfilled iff $i_{\text{before}} < i \wedge j < i_{\text{after}}$ (i_{after} defined analogously).

Since i and j are independent and both must lie within a certain interval, the set of time windows $[i, j]$ for which s is a Delaunay simplex can be represented as a rectangle $]i_{\text{before}}, i_{\text{lower}}] \times [i_{\text{upper}}, i_{\text{after}}[$, see Figure 3. The two dimensions give the range of the lower end of time windows in which s is a Delaunay simplex, and of the upper end, respectively.

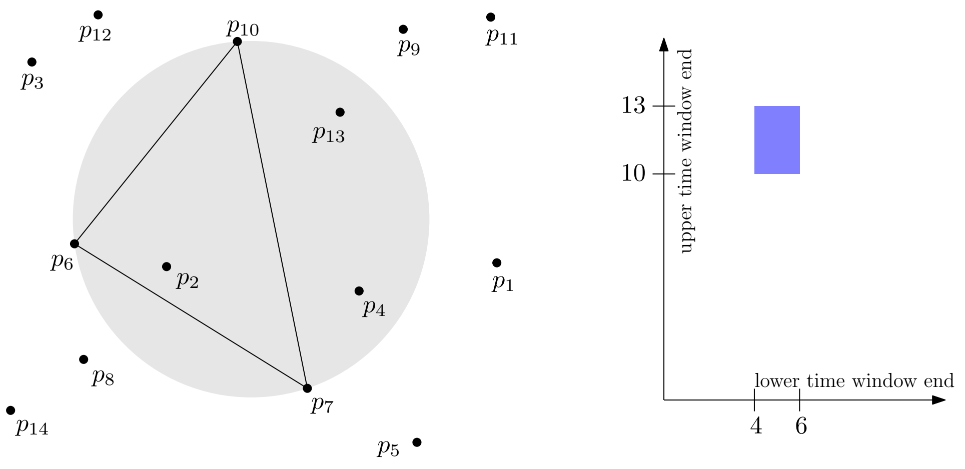


Figure 3 Left: A Delaunay simplex s and some points, some inside the circumsphere of s. **Right:** The activity space of s, bounded by $i_{\text{lower}} = 6$, $i_{\text{upper}} = 10$, $i_{\text{before}} = 4$, and $i_{\text{after}} = 13$.

3.2 Activity Spaces of Delaunay Faces

Activity spaces of Delaunay faces are not necessarily rectangular because, unlike Delaunay simplices, faces may have different empty spheres in different time windows. A Delaunay face f occurring in T is a Delaunay face in time window $[i, j]$ iff *(a)* all vertices of f have point indices between i and j and *(b)* an open sphere S exists which contains no points of $P_{i,j}$ and passes through all vertices of f. Like before, *(a)* is fulfilled iff $i \leq i_{\text{lower}} \wedge i_{\text{upper}} \leq j$.

For *(b)*, we can always find such an empty sphere for faces on the convex hull boundary by choosing its center sufficiently far away. For all other faces, such an empty sphere exists as long as points on one side of f are not too close to points on the other side of f. Otherwise there exists some pair of points which certifies that no such sphere can exist. There may be many such pairs over all time windows, and *(b)* is fulfilled iff the time window $[i, j]$ contains no such pair. Figure 4 shows an example of the resulting activity space.

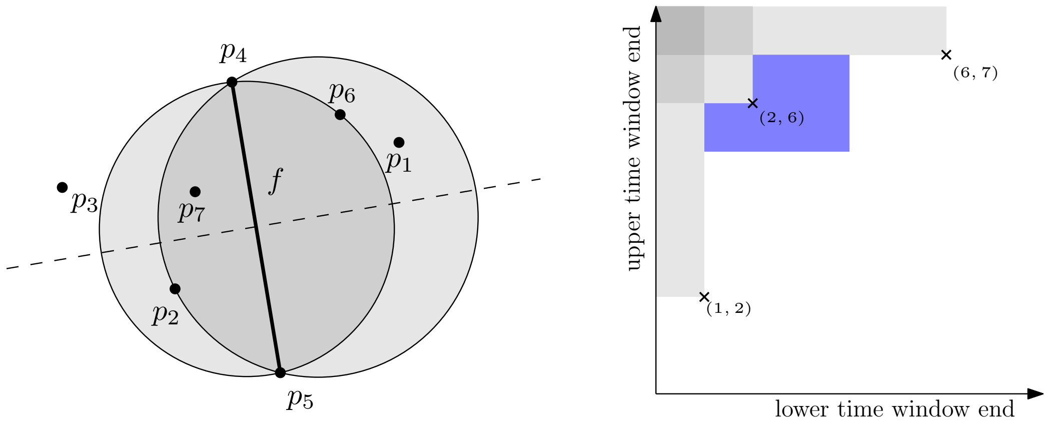


Figure 4 Left: All spheres through the vertices of Delaunay face f are centered on the dashed line. Some points form simplices with f in some time windows. The pairs $\{p_1, p_2\}$, $\{p_2, p_6\}$, and $\{p_6, p_7\}$ prevent f from being a Delaunay face. More such pairs exist, but their influence on the activity space of f is dominated by that of the 3 mentioned pairs. **Right:** Subtracting the rectangles defined by the 3 pairs (transparent grey) from $]-\infty, 4] \times [5, \infty[$ gives the activity space of f (blue).

▶ **Definition 3.1.** *The staircase property*

The activity space of a Delaunay face is shaped like a staircase, obtained by taking the rectangle $]-\infty, i_{\text{lower}}] \times [i_{\text{upper}}, \infty[$ and subtracting from it every rectangle $]-\infty, i_a] \times [i_b, \infty[$ corresponding to a pair as described above (w.l.o.g. we assume $i_a < i_b$).

3.3 Activity Spaces of α-faces

Every Delaunay face is also an α-face for certain values of α, so activity spaces of α-faces have 3 dimensions: 2 temporal dimensions as before, and a dimension for the α-value range.

An α-ball of some face f may be centered on either side of f. We will consider both sides separately and study the activity space for one side only; the activity space of f then is the union of both sides' activity spaces. Consider some Delaunay face f and fix one side of it as the front, the other as the back. We want to know what the activity space looks like if we only consider α-balls whose center is in front of f. Let s_{front} and s_{back} be the cofaces of f in $T_{i,j}$, C_{front} and C_{back} their circumspheres, and r_{front} and r_{back} their circumradii. All spheres through the vertices of f are centered on a common line, this is true in particular for C_{front} and C_{back} as well as for the α-ball centered in front of f. To ensure that the α-ball is empty, it must be centered between C_{front} and C_{back}, see Figure 5. We are only interested in α-balls in front of f, so C_{back} is irrelevant if it is centered behind f.

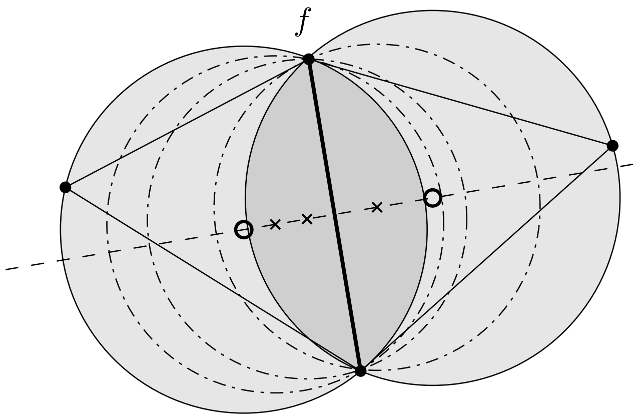


Figure 5 The cofaces of Delaunay face f determining the possible centers of empty α-balls of f. The circumspheres of f's cofaces are filled in grey, their circumcenters are shown by circles. Crosses indicate possible centers of empty α-balls, which are shown with a dash dotted outline.

We get that $r_{\text{back}} \leq \alpha \leq r_{\text{front}}$ in case C_{back} and C_{front} are centered in front of f, and $r_{\min} \leq \alpha \leq r_{\text{front}}$ if only C_{front} is centered in front of f. Here $r_{\min}$ is the radius of the smallest sphere passing through the vertices of f. If C_{front} is also centered behind f, no empty α-ball may exist in front of f and the value range of α is empty.

The α-face activity space of f is defined by its cofaces occurring in T. For every pair of cofaces s_{front} and s_{back}, we get a cuboid whose extent in the temporal dimensions is the intersection of the activity spaces of s_{front} and s_{back}, and the α-value range is determined as above. The front activity space of f is the union of these cuboids over all pairs of cofaces.

The requirement for the α-ball to be centered in front of f implies that, if we project the front activity space down into the two temporal dimensions, it may only cover a subset of the activity space of f as a Delaunay face. Still, the staircase property applies w.r.t. the temporal dimensions because the staircase is effectively shortened in one dimension. No two front cuboids overlap even when disregarding the third dimension.

4 Computing the Temporal α-Shape

We can now present our algorithm to compute the temporal α-shape. We have to compute the α-shape activity space of every Delaunay face occurring in T. Consider some Delaunay face f in T and fix one side as the front side. We describe the process for this side of f. In the following we first collect all cofaces of f in T. Then we compute the cuboids described in Section 3.3 by intersecting the activity space rectangles of all pairs of cofaces and computing the corresponding α-value range. Discarding rectangles which force an empty α-value range and choosing a sensible computation order lets us avoid explicitly computing empty cuboids.

Since f has exactly one front coface in every time window it is a Delaunay face in, we can partition the staircase representing the Delaunay face front activity space of f into rectangles which correspond to the front cofaces of f over all time windows. With some additional bookkeeping, the Delaunay enumeration algorithm [3, 4] allows us to list these cofaces in a nice order in what is essentially one pass over T. We get a partition of the Delaunay face front activity space of f into lists of rectangles. Figure 6 shows some examples.

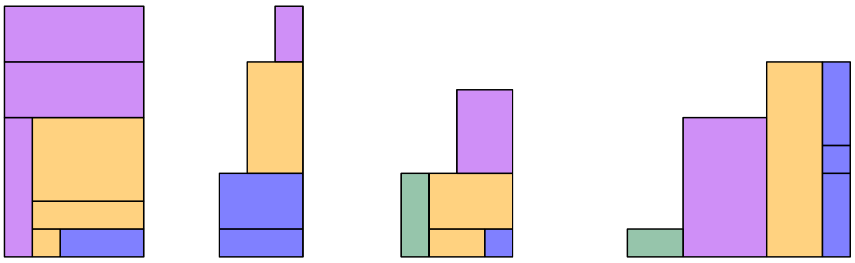


Figure 6 The front activity space of four faces represented by their front rectangle lists. For each face, the rectangles corresponding to the same rectangle list are highlighted in the same color.

We can make some useful observations about these rectangle lists.

▶ **Lemma 4.1.** *Every rectangle is aligned with the right or bottom staircase boundary.*

▶ **Lemma 4.2.** *Two rectangles have the same left boundary iff they are in the same list.*

▶ **Lemma 4.3.** *The rectangles within each list are stacked on top of each other.*

We reorganize this staircase representation into two lists, a *bottom* list and a *right* list. The *bottom* list holds all rectangles touching the bottom boundary, ordered left to right. The *right* list holds all remaining rectangles, ordered bottom to top. Front rectangles which correspond to simplices whose circumcenter lies behind f force an empty α-value range, so we remove them. This effectively cuts off some rectangles of the *bottom* list from the left, and some rectangles of the *right* list from the top, see Figure 7, top left.

To ensure minimality of the final cuboid set, we merge all back rectangles of f with no effect on the α-value range of the front into one combined rectangle (Figure 7, bottom left). That rectangle can include time windows where f is not Delaunay, but this is not an issue as the next step will intersect back and front rectangles, and the latter have not been extended.

We have worked only with Delaunay simplex activity spaces so far. To construct the cuboids representing the α-face front activity space we now overlay the front staircase and the back staircase, i.e. we intersect front rectangles with back rectangles. Each pair of cofaces (i.e. pair of front rectangle and back rectangle) determines a certain range of α as described in Section 3.3. This range, together with the intersection of the temporal dimensions of the two rectangles, produces the resulting cuboid.

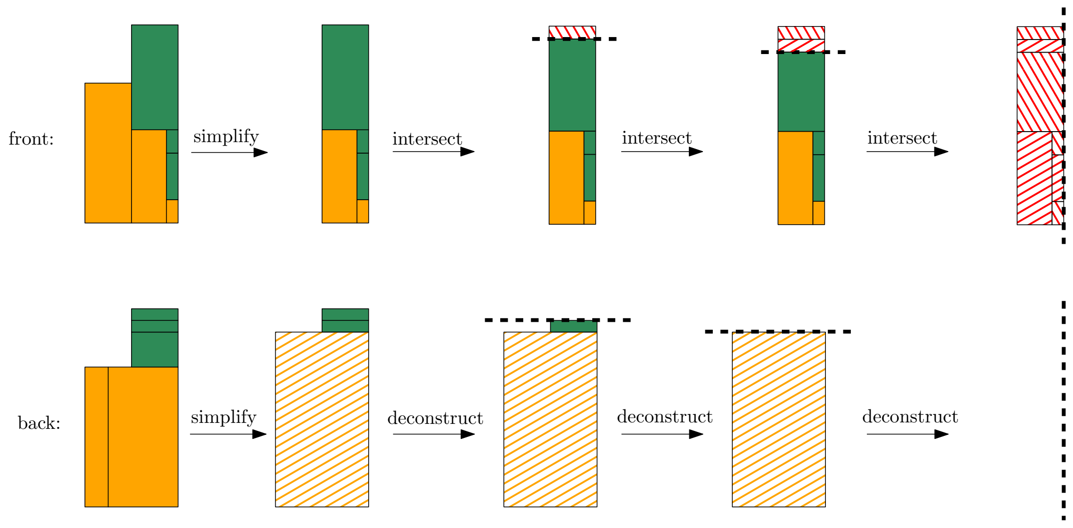


Figure 7 The *bottom* (orange) and *right* (green) lists of f's front and back as they are simplified and intersected. **Simplification:** We remove some front rectangles from the left, and merge some back rectangles from the bottom right into the rectangle shown with orange stripes. **Intersection:** The back staircase is deconstructed rectangle by rectangle, always intersecting the removed rectangle with the front rectangles. This corresponds to cutting off all remaining rectangles from the top or from the left. The resulting intersected rectangles are shown with red stripes. Together with their α-value range they form the cuboids which partition the α-face front activity space of f.

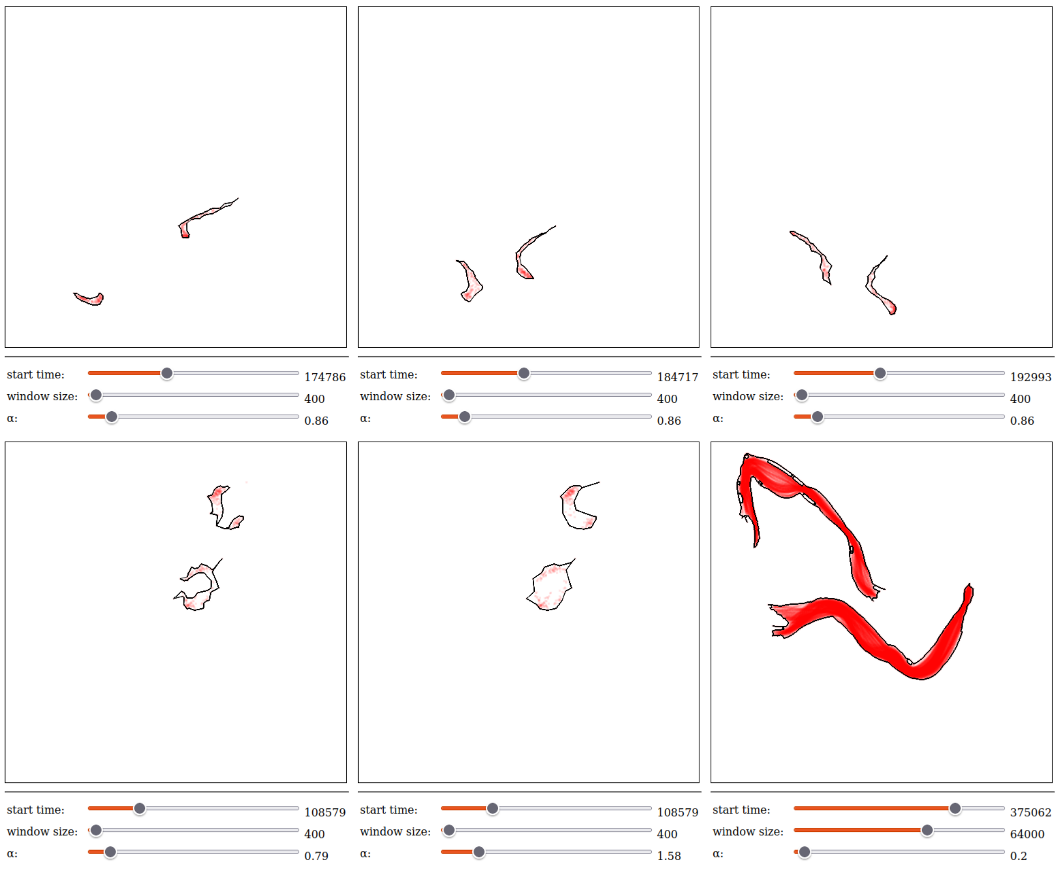
Intersections can be accomplished in output-sensitive linear time because intersections happen at the boundaries of the remaining front staircase, see Figure 7. We repeat this process for both sides of every Delaunay face f in T and get the following theorem.

▶ **Theorem 4.4.** *There exists an algorithm to compute a minimal description* A *of all α-shapes over all time windows and all values of α for a set of timestamped points in $\mathbb{R}^d$ in output-sensitive linear time $\mathcal{O}(|\mathrm{A}|)$ for arbitrary fixed d.*

Note that the size of the temporal α-shape is not necessarily $\mathcal{O}(|T|)$. Overlaying the front and back Delaunay face activity spaces of some face f may create $\Omega(n^2)$ cuboids for f, and one can easily construct families of instances where $|A| \in \Omega(n \cdot |T|)$. It is trivial to show that $|A| \in \mathcal{O}(n^2 \cdot |T|)$, but better bounds, both in terms of $|T|$ and in terms of n, are likely.

5 Demo Application

We use the temporal α-shape to interactively visualize spatio-temporal point sets. Our demo data set is a 2D particle animation with 400 particles. A copy of each particle is created for each movement step, so each of the 1,200 steps is encoded with 400 timestamps and we get 480,000 total data points. Section 6 shows that duplicating each point for each movement step only makes the temporal α-shape twice as large as necessary in this instance. Figure 8 shows our interactive visualization application on this data set. A video demo is available online at `https://youtu.be/Esh7_uzmBac`.



start time: 174786
window size: 400
α: 0.86

start time: 184717
window size: 400
α: 0.86

start time: 192993
window size: 400
α: 0.86

start time: 108579
window size: 400
α: 0.79

start time: 108579
window size: 400
α: 1.58

start time: 375062
window size: 64000
α: 0.2

Figure 8 Interactive visualization of spatio-temporal point sets using α-shapes. The particle sets in red are overlaid only for reference. **Top:** Following particle swarm movement by advancing the time window. **Bottom left and bottom middle:** Changing the α-value to control the level of detail in the outline. **Bottom right:** Revealing swarm trajectories using a larger time window.

6 Experimental Results

Using a basic 2D implementation of our algorithm on an Intel® Xeon® E5-2650 v4 CPU the data set of Section 5 takes 147.3 seconds to compute the 45,494,688 simplices of T, and 222.1 seconds to compute the 127,227,279 cuboids representing the temporal α-shape.

Short time windows and small α-values may be irrelevant for real-world applications. We restrict the time window size and α-value range to realistic values and count how many cuboids are necessary to correctly answer queries. Considering only time windows aligned with movement steps (encoded using 400 distinct timestamps), we need 49% of all cuboids. Ignoring time windows shorter than 128 movement steps we still need 21%. Restricting α to ≥ 0.1 (α-shape fragmentation occurs below 0.1), we still need 30% and 11%, respectively.

Finally we consider the data set used in [1]. The approach of [1] takes "less than two minutes" for a single small α-value and 59,789 points on an Intel® Core i7-7700T CPU, and "about 20 minutes" for a single larger α-value. Our approach, on an Intel® Core i5-10500 CPU, takes 20.7 seconds to compute the temporal α-shape. Accounting for query structure setup time in [1], this gives a speedup of at least $\sim$52 over [1], and likely much more for larger α-values, all while considering not just a single value of α, but all values at once.

7 Conclusion

We presented an algorithm to compute the temporal α-shape, which is a minimal description of all α-shapes of a spatio-temporal point set over all time windows and all values of α in output-sensitive linear time, in arbitrary fixed dimensions. Experiments verified the practicality of our algorithm and showed a speedup of at least $\sim$52 over the approach of [1] while considering not just one value of α, but all values at once. It remains to be seen how the size of the temporal α-shape can be bounded in terms of n and $|T|$.

References

1 Annika Bonerath, Benjamin Niedermann, and Jan-Henrik Haunert. Retrieving α-shapes and schematic polygonal approximations for sets of points within queried temporal ranges. In *Proceedings of the 27th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 249–258. ACM, 2019. doi:10.1145/3347146.3359087.

2 Herbert Edelsbrunner, David G. Kirkpatrick, and Raimund Seidel. On the shape of a set of points in the plane. *IEEE Transactions on Information Theory*, 29(4):551–559, 1983. doi:10.1109/TIT.1983.1056714.

3 Stefan Funke and Felix Weitbrecht. Efficiently computing all Delaunay triangles occurring over all contiguous subsequences. In *31st International Symposium on Algorithms and Computation (ISAAC 2020)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPIcs.ISAAC.2020.28.

4 Felix Weitbrecht. Linear time point location in Delaunay simplex enumeration over all contiguous subsequences. In *EuroCG*, pages 399–404, 2022. URL: http://eurocg2022.unipg.it/booklet/EuroCG2022-Booklet.pdf.

On the Number of Delaunay Simplices over all Time Windows in any Dimension

Felix Weitbrecht[1]

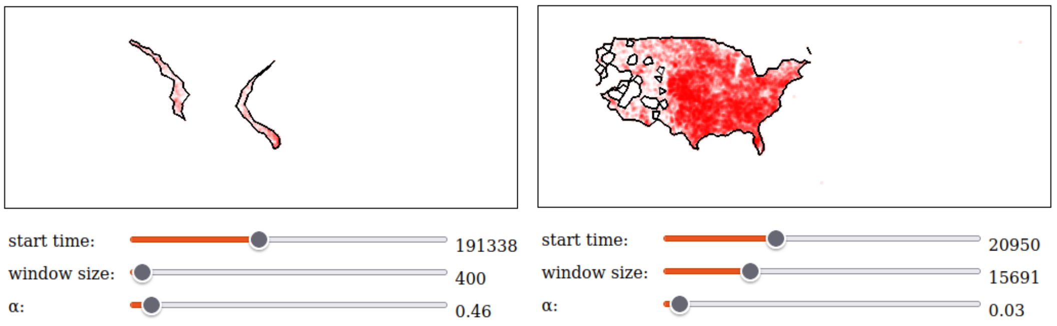
1 **Universität Stuttgart, Germany**
weitbrecht@fmi.uni-stuttgart.de

Abstract

Temporal point data can be examined in great detail by considering time windows (contiguous subsequences) within the data using the Delaunay triangulation and its subcomplexes. Setting up a query structure for not only the Delaunay triangulation of the full point set, but also those of all time windows, requires precomputing T, the set of all Delaunay simplices over all time windows. We assume points are ordered randomly and investigate how $|T|$ behaves with certain realistic assumptions on the input points. We show that many sub-quadratic bounds on the complexity of the Delaunay triangulation apply to $E[|T|]$ with only an added $\log n$ factor, and that many other bounds apply directly to $E[|T|]$. In the case where arbitrary orderings are allowed, we give asymptotic upper bounds for arbitrary d and exact upper bounds for $d \leq 2$, and we give worst-case instances.

1 Introduction

Examining time windows within spatial data sampled over time allows deeper insight, for example to visualize [6, 8] or reconstruct [1, 25] shapes. The Delaunay triangulation and its subcomplexes are valuable tools for such purposes, for example one can use α-shapes of time windows to visualize the temporal evolution of data as in [8] or [27], see Figure 1. Precomputing the set of all Delaunay simplices over all time windows, T, would be useful for this purpose. Precomputation complexity depends on $|T|$, so we consider certain classes of point sets and investigate how $|T|$ behaves if the points are ordered randomly. We show how expected complexity bounds on the Delaunay triangulation can apply to $|T|$. We study how arbitrary point sets in arbitrary order behave in the worst case.


start time: 191338
window size: 400
α: 0.46

start time: 20950
window size: 15691
α: 0.03

Figure 1 Interactive exploration of the α-shape of temporal point data representing a particle swarm (left) and storm events in the United States (right), from [27].

We consider for fixed $d \geq 1$ the d-dimensional Delaunay triangulation $DT(P)$, i.e. that (unique) subdivision of the convex hull of P which consists only of d-simplices whose open circumhyperspheres contain no points from P in their interior. We are given a point set $P = \{p_1, \ldots, p_n\} \subset \mathbb{R}^d$ in general position. Point indices represent the temporal order

39th European Workshop on Computational Geometry, Barcelona, Spain, March 29–31, 2023.
This is an extended abstract of a presentation given at EuroCG'23. 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 a point p_i exists only at time i. We write $P_{i,j}$ for $\{p_i, \ldots, p_j\}$ and $T_{i,j}$ for $DT(P_{i,j})$ $(1 \leq i < j \leq n)$. The set $T := \bigcup_{i<j} T_{i,j}$ is the set of all Delaunay simplices occurring over all time windows $P_{i,j}$.

1.1 Motivation and Related Work

Efficiently precomputing all Delaunay simplices over all time windows is possible in practice [8, 17, 26], but the output complexity depends on the input data. It was previously shown that in two dimensions, $E[|T|] = \Theta(n \log n)$ [17]. The lower bound can be shown to hold, and be tight, for $d \geq 1$, but the matching upper bound only holds for $d \leq 2$; it is not possible to give a matching upper bound in three or more dimensions without making restrictive assumptions on P. This is because the complexity of the d-dimensional Delaunay triangulation varies between $\mathcal{O}(n)$ and $\Omega(n^{\lceil d/2 \rceil})$, while in one and two dimensions it is $\Theta(n)$.

Worst-case inputs are rare in practice, so it is also important to analyze the complexity of realistic input models. Restricting the shape, distribution, or sampling condition of points can yield good upper bounds [3, 4, 5, 7, 9, 10, 11, 12, 13, 16, 19, 20] or lower bounds [11, 15]. Upper bounds are typically of more interest, so they will be our focus in the following.

1.2 Contribution

Section 2 bounds the expected complexity of T when points have a random ordering. We give universal bounds, and bounds derived from Delaunay triangulation complexity bounds where we show that many sub-quadratic bounds apply to $E[|T|]$ with only an added $\log n$ factor, and that many other bounds apply directly to $E[|T|]$. Section 3 investigates the worst case if the points are ordered arbitrarily, giving exact bounds for $d \leq 2$ and asymptotically tight bounds for $d > 2$. We provide constructions with matching complexity for all bounds in this section. Section 4 concludes with an outlook on future work.

2 On the Expected Size of T

In this section P is ordered according to a permutation chosen u.a.r. and we give bounds on $E[|T|]$, the expected number of Delaunay simplices over all time windows. We can count Delaunay simplices of T by considering the smallest time window containing their vertices:

▶ **Lemma 2.1.** *Any Delaunay simplex $s \in T$ is also a Delaunay simplex in the Delaunay triangulation of the minimal time window of P containing all vertices of s.*

We reformulate $E[|T|]$ using Lemma 2.1 and linearity of expectation, with δ_s as an indicator variable which is 1 if the simplex s has both p_i and p_j as a vertex, and 0 otherwise. The points are ordered randomly, so $E[\delta_s]$ is simply the probability of $d+1$ points (the vertices of s) randomly selected from $j-i+1$ points (those of the time window) containing two specific points. Skipping some intermediate steps for brevity of presentation, we get:

$$\begin{aligned} E[|T|] = E\left[\sum_{i=1}^{n-d} \sum_{j=i+d}^{n} \sum_{s \in T_{i,j}} \delta_s\right] &= \sum_{i=1}^{n-d} \sum_{j=i+d}^{n} E\left[\sum_{s \in T_{i,j}} \delta_s\right] \\ &= \sum_{i=1}^{n-d} \sum_{j=i+d}^{n} E[|T_{i,j}|] \frac{d(d+1)}{(j-i+1)(j-i)} \end{aligned} \tag{1}$$

We can now transform existing upper (or lower) bounds on the expected complexity of the Delaunay triangulation into bounds on $E[|T|]$, provided the bound holds for random subsets.

Table 1 Expected Delaunay triangulation complexity bounds compatible with Theorem 2.2

Points	d	$E[\|DT(P)\|]$	Citation
Distributed u.a.r. in a compact convex of fixed volume	≥ 1	$\mathcal{O}(n)$	[9, 13]
ϵ-net on a flat torus	≥ 1	$\mathcal{O}(n)$	[7]
Distributed u.a.r. on the surface of a fixed convex polytope	3	$\mathcal{O}(n)$	[20]
ϵ-net of a fixed polyhedral surface	3	$\mathcal{O}(n)$	[7]
(ϵ, κ)-sample of a fixed polyhedral surface	3	$\mathcal{O}(n\kappa^2)$	[4, 7]
Distributed u.a.r. on a cylinder of constant height	3	$\mathcal{O}(n \log n)$	[11]
Distributed u.a.r. on a fixed polyhedral surface	3	$\mathcal{O}(n \log^2 n)$	[4][1]
Distributed u.a.r. on a fixed generic smooth surface	3	$\mathcal{O}(n \log^3 n)$	[5][1]

▶ **Theorem 2.2.** *Let $\mathcal{P}$ be a family of point sets in $\mathbb{R}^d$ for fixed arbitrary d, with an expected upper bound $E[|DT(P)|] = \mathcal{O}(f(n))$ for $P \in \mathcal{P}$ which has the property that, if R is a random subset of size r, $E[|DT(R)|] = \mathcal{O}(f(r))$. Then, if P is ordered by a permutation chosen u.a.r., $E[|T|] = \mathcal{O}(f(n) \log n)$.*

Proof. Use the bound in Equation 1. We can bound $\frac{f(j-i+1)}{j-i+1}$ with $\frac{f(n)}{n}$ since $f(n) \in \Omega(n)$.

$$E[|T|] = \sum_{i=1}^{n-d} \sum_{j=i+d}^{n} E[|T_{i,j}|] \frac{d(d+1)}{(j-i+1)(j-i)} = \mathcal{O}\left(\sum_{i=1}^{n-d} \sum_{j=i+d}^{n} \frac{f(j-i+1)}{(j-i+1)(j-i)} \right)$$
$$= \mathcal{O}\left(\frac{f(n)}{n} \sum_{i=1}^{n-d} \sum_{j=i+d}^{n} \frac{1}{j-i} \right) = \mathcal{O}\left(\frac{f(n)}{n} n \log n \right) = \mathcal{O}(f(n) \log n)$$
◀

Table 1 gives an overview of some bounds compatible with Theorem 2.2. For upper bounds $f(n) \in \Omega(n^2)$ we can show $E[|T|] = \mathcal{O}(f(n))$ with the same approach. The complexity of the Delaunay triangulation is $\mathcal{O}(n^{\lceil d/2 \rceil})$, and this bound is at least quadratic for $d \geq 3$, so we get the higher-dimensional analogon of the $\mathcal{O}(n \log n)$ 2D bound of [17]:

▶ **Corollary 2.3.** *For $d \geq 3$, $E[|T|] = \mathcal{O}(n^{\lceil d/2 \rceil})$, and this bound is tight.*

The bound is tight because instances of P exist with $|DT(P)| = \Theta(n^{\lceil d/2 \rceil})$ for all n, d [22, 23].

3 On the Maximum Size of T

In this section P has an arbitrary fixed order. We derive a tight asymptotic upper bound on $|T|$ for arbitrary d in Section 3.1 and Section 3.2 gives exact upper bounds for small d.

3.1 Asymptotic Upper Bounds

Clearly $|T| = \mathcal{O}(n^{2+\lceil d/2 \rceil})$ because there are only quadratically many time windows and any one Delaunay triangulation is of size $\mathcal{O}(n^{\lceil d/2 \rceil})$, but we can do better. Theorem 3.2 shows that $|T| = \mathcal{O}(n^{1+\lceil d/2 \rceil})$, and Lemma 3.4 shows that this is tight in the worst case.

Cyclic polytopes are the basis of the proofs in this section. The upper bound theorem [24] states that cyclic polytopes maximize the number of i-dimensional faces among all simplicial

[1] By Chernoff bounds, random samples are $(\mathcal{O}(\sqrt{\frac{\log n}{n}}), \mathcal{O}(\log n))$-samples with high probability [16].

$(d-1)$-spheres for all $i < d$, i.e. the number $C_i(n, d)$ of i-faces of a cyclic polytope with n vertices in d dimensions is an upper bound on the number of i-faces of any convex polytope with n vertices in d dimensions. Cyclic polytopes with the same dimension and number of vertices are combinatorially equivalent, and one way of construction is to choose n distinct points $\{x(t_1), \ldots, x(t_n)\}$ from the moment curve $x(t) = (t, t^2, \ldots, t^d)^T$ restricted to $t > 0$.

We can identify Delaunay simplices of a d-dimensional point set with convex hull facets of a $d+1$-dimensional point set [14], and we can bound the number of such facets using the upper bound theorem. This lets us bound the number of Delaunay simplices of $T_{i,j}$ which are incident to both p_i and p_j in Lemma 3.1, which we use in Theorem 3.2 to bound $|T|$.

▶ **Lemma 3.1.** *There are $\mathcal{O}((j-i+1)^{\lfloor (d-1)/2 \rfloor})$ simplices incident to both p_i and p_j in $T_{i,j}$.*

Proof. Let $S(k, d)$ be the maximum number of Delaunay simplices incident to any two common points in the Delaunay triangulation of any k points in d dimensions. For $d \leq 2$ or $k \leq d+1$, $S(k, d)$ is $\mathcal{O}(1)$. For $d \geq 3$ and $k > d+1$, we use the lifting map to project P onto a paraboloid in $(d+1)$-space [14]. Then a $(d+1)$-subset F of P forms a Delaunay simplex if and only if the points of F lifted onto the paraboloid form a lower convex hull facet.

We bound the number of facets incident to two common points, p_a and p_b, on any convex hull $\mathcal{H}$ of $k > d+1$ points in $d+1$ dimensions in general position. This bound applies to the convex hull of the lifted points, and to the Delaunay triangulation of the original points.

The facets (i.e. d-simplices) of $\mathcal{H}$ which are incident to p_a each have exactly one $(d-1)$-simplex not incident to p_a. The set $\mathcal{K}_1$ of these $(d-1)$-simplices is a simplicial $(d-1)$-sphere with at most $k-1$ vertices. We apply the same logic to a vertex in $\mathcal{K}_1$, say p_b:

The facets (i.e. $(d-1)$-simplices) of $\mathcal{K}_1$ which are incident to p_b each have exactly one $(d-2)$-simplex not incident to p_b. The set $\mathcal{K}_2$ of these $(d-2)$-simplices is a simplicial $(d-2)$-sphere with at most $k-2$ vertices. By construction, the facets of $\mathcal{K}_2$ are in one-to-one correspondence with those facets of $\mathcal{H}$ which are incident to p_a and p_b. By the upper bound theorem [24], the number of facets in $\mathcal{K}_2$ is at most $C_{d-2}(k-2, d-1) = \mathcal{O}(k^{\lfloor (d-1)/2 \rfloor})$. ◀

▶ **Theorem 3.2.** *The number of Delaunay simplices over all time windows in d dimensions is $\mathcal{O}(n^{1+\lceil d/2 \rceil})$ for all $d \geq 1$.*

Proof. We use Lemma 2.1 and count for each time window $P_{i,j}$ the Delaunay simplices for which $P_{i,j}$ is the minimal enclosing time window, which are bounded by Lemma 3.1:

$$|T| = \sum_{i=1}^{n-d} \sum_{j=i+d}^{n} \mathcal{O}((j-i+1)^{\lfloor (d-1)/2 \rfloor}) = \mathcal{O}(n^{1+\lceil d/2 \rceil})$$

We omit the tedious derivation of explicitly summing up the $C_{d-2}(k-2, d-1)$ of Lemma 3.1, which would give an explicit upper bound of $|T| \leq \binom{n-\lfloor d/2 \rfloor}{\lceil d/2 \rceil+1} + \binom{n-\lceil d/2 \rceil}{\lfloor d/2 \rfloor+1} - n + d$. ◀

Gale's Evenness Condition characterizes cyclic polytope facets:

▶ **Theorem 3.3** (Gale's Evenness Condition [18]). *Let $P = \{x(t_1), \ldots, x(t_n)\}$ be a set of distinct points chosen from the moment curve $x(t) = (t, t^2, \ldots, t^d)^T$ restricted to $t > 0$ and ordered by increasing t_i. Then a d-subset $F \subseteq P$ forms a facet of the cyclic polytope $C_d(n) = CH(P)$ iff the following condition is satisfied: for every pair of points $(x(t_{i_1}), x(t_{i_2}))$ in $P \setminus F$ with $i_1 < i_2$, the number of points $x(t)$ in F with $t_{i_1} < t < t_{i_2}$ is even.*

We adapt a construction from [21] and use Gale's Evenness Condition in Lemma 3.4 to show that the bound given in Theorem 3.2 is asymptotically tight.

▶ **Lemma 3.4.** *Point sets P with $T \in \Omega(n^{1+\lceil d/2 \rceil})$ exist for all $n \geq d \geq 1$.*

Proof. Pick n points $P' = \{p'_1, \dots, p'_n\}$ with strictly increasing $t > 0$ on the moment curve $x(t) = (t, t^2, \dots, t^d)^T$ such that every p'_j is strictly outside the circumhypersphere of simplices formed by $d+1$ points with index $< j$. The convex hull of any subset $R = \{p'_{i_1}, \dots, p'_{i_k}\} \subseteq P'$ with $i_1 < i_2 < \dots < i_k$ and $k > d$ is a cyclic polytope, so Gale's Evenness Condition tells us which d-subsets $F \subset R$ form a facet of $CH(R)$, and thus also which F no longer form a facet after inserting a point $p'_j \in P'$ with $j > i_k$ into $CH(R)$: they are exactly those F for which Gale's Evenness Condition holds w.r.t. R, but not w.r.t. $R \cup \{p'_j\}$, i.e. $p'_{i_k} \in F$ (and $p'_{i_1} \in F$ if d is even), and all other points of F can be decomposed into $\lfloor \frac{d-1}{2} \rfloor$ pairwise disjoint pairs of points with consecutive indices in R. Roughly estimating the number of such d-subsets of R, we see that $\Omega(k^{\lfloor (d-1)/2 \rfloor})$ facets disappear when inserting p'_j.

Now consider the Delaunay triangulation of R and how it changes when inserting p'_j. Due to the choice of the p'_i, no Delaunay simplex of $DT(R)$ contains p'_j in its circumhypersphere, so each d-subset F forming a disappearing facet corresponds to a $d+1$-subset $F \cup \{p'_j\}$ forming a Delaunay simplex of $DT(R \cup \{p'_j\})$. We are now ready to construct P. Re-order the points of P' by reversing the order within the first and second half:

$$P = \{p_1, \dots, p_n\}, \text{ with } p_i = \begin{cases} p'_{\lfloor n/2 \rfloor - i + 1} & , i \leq \lfloor \frac{n}{2} \rfloor \\ p'_{n + \lfloor n/2 \rfloor - i + 1} & , i > \lfloor \frac{n}{2} \rfloor \end{cases}$$

Ordered by t-value, P would look like this: $[p_{\lfloor n/2 \rfloor}, p_{\lfloor n/2 \rfloor - 1}, \dots, p_1, p_n, p_{n-1}, \dots, p_{\lfloor n/2 \rfloor + 1}]$. Observe that for $i \leq \lfloor \frac{n}{2} \rfloor - d$, all facets of $T_{i, \lfloor n/2 \rfloor}$ that would disappear upon insertion of some p_j with $j > \lfloor \frac{n}{2} \rfloor$ have p_i as a vertex, and the resulting Delaunay simplices of $DT(P_{i, \lfloor n/2 \rfloor} \cup \{p_j\})$ all have p_j as a vertex. By construction, $p_{\lfloor n/2 \rfloor + 1}, p_{\lfloor n/2 \rfloor + 2}, \dots, p_{j-1}$ are strictly outside of the circumhypersphere of these Delaunay simplices, so they are also Delaunay simplices in $T_{i,j}$. Bound the number of Delaunay simplices in T using Lemma 2.1:

$$|T| \geq \sum_{i=1}^{\lfloor n/2 \rfloor - d} \sum_{j = \lfloor n/2 \rfloor + 1}^{n} \Omega((\lfloor \frac{n}{2} \rfloor - i + 1)^{\lfloor (d-1)/2 \rfloor}) = \Omega(n^{1+\lceil d/2 \rceil})$$ ◀

3.2 Exact Upper Bounds

We give exact upper bounds and constructions that realize them in one and two dimensions. Surprisingly, the explicit bound of Theorem 3.2 is tight in two dimensions, but not in one.

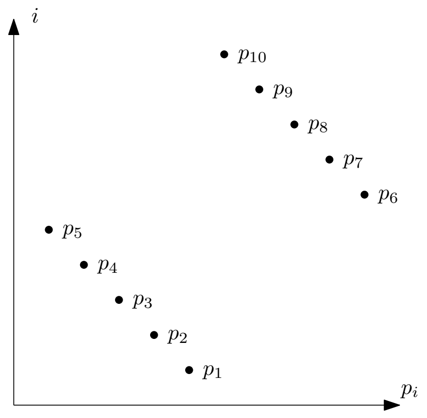


Figure 2 1D point set with point indices plotted in the second dimension.

▶ **Lemma 3.5.** *For $d = 1$ and $n \geq 2$, $|T| \leq \lfloor \frac{n^2}{4} \rfloor + n - 2$, and this bound is tight.*

Proof. The one-dimensional Delaunay triangulation is the set of segments defined by two neighboring points on a line. By Lemma 2.1, a segment p_ip_j $(i < j)$ is in T iff $P_{i,j}$ contains no points between p_i and p_j. Plot the points in a 2D coordinate system, with p_i corresponding to the x-value and the index i corresponding to the y-value, and we get that $p_ip_j \in T$ iff there exists a closed axis-aligned rectangle containing p_i and p_j but no other points from P. It is shown in [2] that at most $\lfloor \frac{n^2}{4} \rfloor + n - 2$ such pairs of points can exist.

The construction in Lemma 3.4 reaches this bound, see Figure 2: for $i \leq \lfloor \frac{n}{2} \rfloor < j$ we get $\lfloor \frac{n^2}{4} \rfloor$ segments of the type p_ip_j, plus $n - 2$ segments of the type p_ip_{i+1} for $i \notin \{\lfloor \frac{n}{2} \rfloor, n\}$. ◀

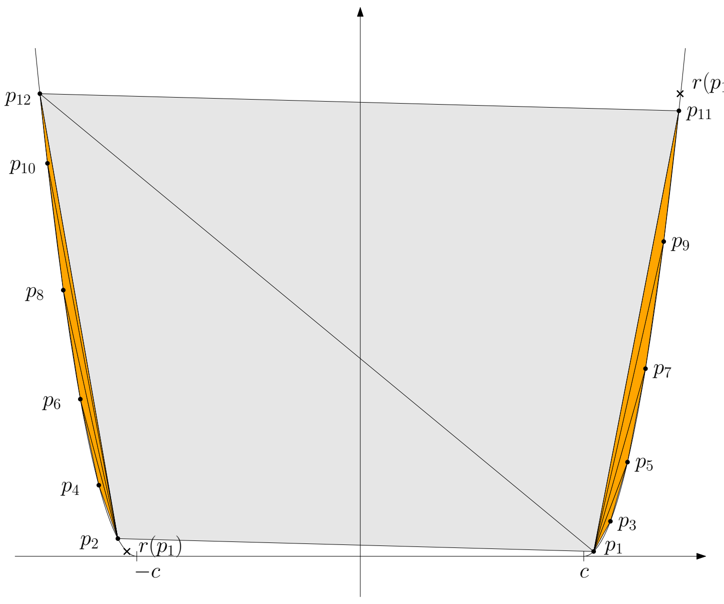


Figure 3 Split parabola construction scheme, not to scale. Triangles of either half parabola in orange, quadrilateral used in the proof in grey, and its diagonal edge.

▶ **Lemma 3.6.** *For $d = 2$ and $n \geq 2$, $|T| \leq (n - 2)^2$, and this bound is tight.*

Proof. We bound for every time window $P_{i,j}$ the number of triangles of $T_{i,j}$ incident to both p_i and p_j. This bounds $|T|$ by Lemma 2.1. Each of the $\frac{n(n-1)}{2}$ time windows contributes up to 2 triangles to the total, but the $n - 1$ time windows of length 2 admit no triangles and the $n - 2$ time windows of length 3 admit only one triangle each. We get a bound of $|T| \leq 2\frac{n(n-1)}{2} - 2(n - 1) - (n - 2) = (n - 2)^2$. It can be realized as follows, like in Figure 3:

$$P = \{p_1, \ldots, p_n\}, \text{ with } p_i = \begin{cases} (i + c, i^2)^T & , i \text{ odd} \\ (-i - c, i^2)^T & , i \text{ even} \end{cases}$$

Let $r(p)$ be the reflection over the y-axis, and $P' = P \cup \{r(p) \mid p \in P\}$, and c a constant chosen large enough so that every point of P' with negative x-value is strictly outside the circumcircle of any triangle formed by 3 points of P' with positive x-value.

We show that the edge p_ip_j exists in $T_{i,j}$ and has a triangle on both sides for all $j > i+2$. Due to the choice of c, $T_{i,j}$ consists of two separate, independently triangulated half parabolas which are connected by exactly 3 edges between their lowest and highest points. If i and j

have the same parity, the edge p_ip_j is on the convex hull boundary of one half, and thus exists in $T_{i,j}$. The edge is between both halves, so a triangle exists on both sides.

Otherwise, assume w.l.o.g. that j is even. The convex quadrilateral $p_ip_{j-1}p_jp_{i+1}$ in the middle is fixed already. A sphere morphing argument using spheres through $p_{i+1}, p_j, r(p_i)$ and $p_i, p_{j-1}, r(p_j)$ shows that p_ip_j is the diagonal edge triangulating that quadrilateral. ◀

4 Outlook

Applying other complexity bounds on the Delaunay triangulation to the number of Delaunay simplices over all time windows remains an interesting challenge, especially when point order is not randomized or points are distributed less evenly. Exact upper bounds on $|T|$ for $d \geq 3$, and finding for which d the explicit bound given in Theorem 3.2 is tight, would also be nice.

References

1 Ehsan Aganj, Jean-Philippe Pons, Florent Ségonne, and Renaud Keriven. Spatio-temporal shape from silhouette using four-dimensional Delaunay meshing. In *2007 IEEE 11th International Conference on Computer Vision*, pages 1–8. IEEE, 2007. doi:10.1109/ICCV.2007.4409016.

2 Noga Alon, Zoltán Füredi, and Meir Katchalski. Separating pairs of points by standard boxes. *European Journal of Combinatorics*, 6(3):205–210, 1985. doi:10.1016/S0195-6698(85)80028-7.

3 Nina Amenta, Dominique Attali, and Olivier Devillers. A tight bound for the Delaunay triangulation of points on a polyhedron. *Discrete & Computational Geometry*, 48(1):19–38, 2012. doi:10.1007/s00454-012-9415-7.

4 Dominique Attali and Jean-Daniel Boissonnat. A linear bound on the complexity of the Delaunay triangulation of points on polyhedral surfaces. *Discrete & Computational Geometry*, 31(3):369–384, 2004. doi:10.1007/s00454-003-2870-4.

5 Dominique Attali, Jean-Daniel Boissonnat, and André Lieutier. Complexity of the Delaunay triangulation of points on surfaces: the smooth case. In *Proceedings of the nineteenth annual symposium on Computational Geometry*, pages 201–210, 2003. doi:10.1145/777792.777823.

6 Michael J. Bannister, William E. Devanny, Michael T. Goodrich, Joseph A. Simons, and Lowell Trott. Windows into geometric events: Data structures for time-windowed querying of temporal point sets. In *Proceedings of the 26th Canadian Conference on Computational Geometry (CCCG)*, pages 11–19, 2014. URL: https://www.cccg.ca/proceedings/2014/papers/paper02.pdf.

7 Jean-Daniel Boissonnat, Olivier Devillers, Kunal Dutta, and Marc Glisse. Randomized incremental construction of Delaunay triangulations of nice point sets. *Discrete & Computational Geometry*, pages 1–33, 2020. doi:10.1007/s00454-020-00235-7.

8 Annika Bonerath, Benjamin Niedermann, and Jan-Henrik Haunert. Retrieving α-shapes and schematic polygonal approximations for sets of points within queried temporal ranges. In *Proceedings of the 27th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 249–258. ACM, 2019. doi:10.1145/3347146.3359087.

9 Olivier Devillers. Delaunay triangulation and randomized constructions. In *Encyclopedia of Algorithms*. Springer, 2016. doi:10.1007/978-1-4939-2864-4_711.

10 Olivier Devillers and Charles Duménil. A poisson sample of a smooth surface is a good sample. In *EuroCG 2019 - 35th European Workshop on Computational Geometry*, Utrecht, Netherlands, March 2019. URL: https://hal.archives-ouvertes.fr/hal-02394144.

11 Olivier Devillers, Jeff Erickson, and Xavier Goaoc. Empty-ellipse graphs. In *19th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'08)*, pages 1249–1256, 2008. URL: https://dl.acm.org/doi/10.5555/1347082.1347218.

12 Rex A Dwyer. Higher-dimensional Voronoi diagrams in linear expected time. *Discrete & Computational Geometry*, 6(3):343–367, 1991. doi:10.1007/BF02574694.

13 Rex A Dwyer. The expected number of k-faces of a Voronoi diagram. *Computers & Mathematics with Applications*, 26(5):13–19, 1993. doi:10.1016/0898-1221(93)90068-7.

14 Herbert Edelsbrunner and Raimund Seidel. Voronoi diagrams and arrangements. *Discrete & Computational Geometry*, 1(1):25–44, 1986. doi:10.1007/BF02187681.

15 Jeff Erickson. Nice point sets can have nasty Delaunay triangulations. *Discrete & Computational Geometry*, 30:109–132, 2003. doi:10.1007/s00454-003-2927-4.

16 Jeff Erickson. Dense point sets have sparse Delaunay triangulations or "... but not too nasty". *Discrete & Computational Geometry*, 33(1):83–115, 2005. doi:10.1007/s00454-004-1089-3.

17 Stefan Funke and Felix Weitbrecht. Efficiently computing all Delaunay triangles occurring over all contiguous subsequences. In *31st International Symposium on Algorithms and Computation (ISAAC 2020)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPIcs.ISAAC.2020.28.

18 David Gale. Neighborly and cyclic polytopes. In *Proc. Sympos. Pure Math*, volume 7, pages 225–232, 1963. doi:10.1090/pspum/007/0152944.

19 Mordecai J Golin and Hyeon-Suk Na. The probabilistic complexity of the Voronoi diagram of points on a polyhedron. In *Proceedings of the eighteenth annual symposium on Computational geometry*, pages 209–216, 2002. doi:10.1145/513400.513426.

20 Mordecai J Golin and Hyeon-Suk Na. On the average complexity of 3d-Voronoi diagrams of random points on convex polytopes. *Computational Geometry*, 25(3):197–231, 2003. doi:10.1016/S0925-7721(02)00123-2.

21 Bernd Gonska and Günter M. Ziegler. Inscribable stacked polytopes. *Advances in Geometry*, 13(4):723–740, 2013. doi:10.1515/advgeom-2013-0014.

22 Raimund Seidel. The complexity of Voronoi diagrams in higher dimensions. In *Proceedings of the 20th Annual Allerton Conference on Communication, Control, and Computing*, volume 300, pages 94–95, 1982.

23 Raimund Seidel. Exact upper bounds for the number of faces in d-dimensional Voronoi diagrams. In *Applied Geometry and Discrete Mathematics: The Victor Klee Festschrift*, pages 517–530, 1991. doi:10.1090/dimacs/004/40.

24 Richard P Stanley. The upper bound conjecture and cohen-macaulay rings. *Studies in Applied Mathematics*, 54(2):135–142, 1975. doi:10.1002/sapm1975542135.

25 Jochen Süßmuth, Marco Winter, and Günther Greiner. Reconstructing animated meshes from time-varying point clouds. In *Computer Graphics Forum*, volume 27, pages 1469–1476. Wiley Online Library, 2008. doi:10.1111/j.1467-8659.2008.01287.x.

26 Felix Weitbrecht. Linear time point location in Delaunay simplex enumeration over all contiguous subsequences. In *EuroCG*, pages 399–404, 2022. URL: http://eurocg2022.unipg.it/booklet/EuroCG2022-Booklet.pdf.

27 Felix Weitbrecht. Interactive exploration of the temporal α-shape. In *EuroCG*, 2023.

Vorosketch and the L_0 distance

Herman Haverkort

Institute of Computer Science, University of Bonn
haverkort@uni-bonn.de

Abstract

This paper briefly introduces Vorosketch, a simple tool that sketches Voronoi diagrams for a variety of distance measures, suitable for use in lectures, articles, and research. Experiments with Vorosketch led us to the following discovery. Consider the Voronoi diagram of a set of points in the plane when the distance between two points a and b is given by $L_p(a-b)$, where $L_p((x,y)) = (|x|^p + |y|^p)^{1/p}$. For $p = 0$, this distance is undefined. Nevertheless, the Voronoi diagram has a limit as p converges to zero from above or from below: it is the diagram that corresponds to the distance function $L_*((x,y)) := |xy|$. This suggests that L_* provides a natural definition of a *geometric* L_0 *distance*.

1 Drawing Voronoi diagrams with Vorosketch

Given a set S of n points in the plane (called *sites*) and a distance function $d : S \times \mathbb{R}^2 \to \mathbb{R}$, the Voronoi region of a point $a \in S$ is the set $\{q \in \mathbb{R}^2 \mid d(a,q) \leq \min_{s \in S} d(s,q)\}$. The Voronoi diagram of S with respect to d is a subdivision of the plane into the Voronoi regions of the sites of S (ignoring, for now, complications arising from overlap between regions). Each edge of the Voronoi diagram lies on the boundary between the regions of two sites a and b, and thus, it is a part of the *bisector* $B(a,b)$ between a and b, that is, the set of points $B(a,b) := \{q \in \mathbb{R}^2 \mid d(a,q) = d(b,q)\}$.

When d is simply the Euclidean (L_2) distance, the Voronoi diagram can be computed by various algorithms in $O(n \log n)$ time (see e.g. [3, 13]) and various efficient implementations exist. These exploit various convenient properties of Voronoi diagrams, including that all regions are simply connected and that the bisector between any pair of points from S is a straight line.

Besides Voronoi diagrams for the Euclidean distance, diagrams for many other distance functions have been considered. In general, this results in Voronoi diagrams that do not have the aforementioned convenient properties: bisectors are not straight anymore and/or regions might consist of multiple connected components. As a result, the standard algorithms to compute Voronoi diagrams can be used only with non-trivial adaptations, if at all.

Nevertheless, I wanted to study such Voronoi diagrams experimentally, and produce figures of such Voronoi diagrams to explain the topic to students. Therefore I wrote *Vorosketch* [8]: a tool that is not at the summit of speed or accuracy, but can easily be extended with novel distance functions, regardless of whether we understand the geometry of the bisectors yet. Vorosketch produces bitmap images of Voronoi diagrams. The running time is dominated by distance computations. For a bitmap of $r \times r$ pixels, $\Theta(nr^2)$ distances are computed: from each site to the centre of each pixel. However, in practice, the number of distance computations tends to be much smaller than nr^2, thanks to a two-phase approach.

In the first phase, Vorosketch subdivides the drawing area into blocks of 20×20 pixels. For each block B and for each site a, Vorosketch computes a lower bound $\ell(a,B)$ and an upper bound $u(a,B)$ such that $\ell(a,B) \leq d(a,q) \leq u(a,B)$ holds for each pixel centre $q \in B$. In the second phase, Vorosketch computes, for each pixel centre $q \in B$, the distances to only those sites a for which $\ell(a,B) \leq \min_{s \in S} u(s,B)$, and selects out of those, the site closest to q. Bisectors are detected and drawn wherever there is a square of 2×2 pixels that do not

39th European Workshop on Computational Geometry, Barcelona, Spain, March 29–31, 2023.
This is an extended abstract of a presentation given at EuroCG'23. 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.

all belong to the same region (so one has to be a bit careful: bisector parts along regions that are locally smaller or narrower than a pixel's width might be missed).

Naturally, the speed of the computation depends strongly on how tight the computed lower and upper bounds are, but the output quality does not depend on it. To experiment with a new distance measure quickly, it suffices to insert the code for the single-pixel computation in the second phase, and Vorosketch will just use trivial (correct but useless) bounds in the first phase. If desired, one can speed up the computation of diagrams for the new distance measure later by implementing the calculation of tighter bounds.

Vorosketch offers various options to control the line work and colours of the drawing. Regions may be coloured in different colours—using some rudimentary heuristic to give adjacent regions contrasting colours. Colours or shades may be set to depend on the distance to the site (see Figures 1a, 1e, 1f); saturation may be set to depend on the number of neighbours of a region (Figure 1c). One may also choose to apply shading depending on the direction of the distance-to-closest-site gradient—thus visualising the hills and valleys of the distance landscape (Figures 1a, 1c). Alternatively, or on top of that, one may draw distance contour lines. Areas that belong to two Voronoi regions are filled with a checkerboard pattern according to the colours of the two sites involved (Figures 1c, 1d). Vorosketch can also draw second-closest-site, second-order, and farthest-site Voronoi diagrams.

In the current version of Vorosketch [8], the following distance measures are supported:

- L_p distance for $p = -\infty$, all $p \in \mathbb{R}$, and $p = \infty$, using L_* for L_0 (see Section 2);
- distance when only travelling in directions that are integral multiples of 60 degrees;
- *Kalrsruhe/Moscow/Amsterdam* distance (travel only on lines through or circles around the origin) [12], and the novel *Köln/Cologne* distance (like Karlsruhe, but measuring travel time when speed is proportional to the distance from the origin, as if modelling congestion in the centre—see Figure 1a);
- distance when the plane is interpreted as an equal-area azimuthal, elliptical (see Figure 1b) or cylindrical projection of a sphere;
- distance when the plane is interpreted as one of various models of the hyperbolic plane;
- the energy required to send a spaceship from any point into an orbit that reaches the site (or vice versa), subject to gravity towards the centre of the coordinate system (see the Vorosketch website [8] for details on its definition and computation);
- L_1 and L_2 distance with unrestricted-access highways (the distance between two points is the travel time, when speed is 1 when going cross-country, while travel along specified line segments is faster; see Figure 1c); for these distance measures, Vorosketch includes a preprocessing phase that exploits geometric properties of shortest paths in this setting [7];
- combinations of distance measures that can be obtained from the previous by standard arithmetic operations; this includes distance measures calculated according to one function (for example, Euclidean), subject to constraints on another distance function (for example, turn angle, as with half-plane Voronoi diagrams [5, 6]);
- subtractively/additively and divisively/multiplicatively weighted versions of the above (including power diagrams [2, 11]).

Moreover, for line segment or polylinear sites, the following distance measures are included:

- L_1 (see Figure 1d) and L_2 distance;
- *angular-size* distance (the distance between a point q and a site s is 2π divided by the angular size of s as seen from q, minus one; see Figure 1e);
- *detour* distance (the distance to a point q from a line segment site with endpoints s and t is $||sq|| + ||qt|| - ||st||$), and *dilation* distance (detour distance divided by $||st||$).

For point sites that each have an associated vector (direction, and, in some cases, magnitude):

- *turn* distance: the distance to a point q from a site s is the angle between the site's associated vector and the segment sq (see [9]);
- *left-turn* distance: same as above, but always measuring the angle from the site vector to sq in counterclockwise direction (as studied by Alegría et al. [1]);
- *Dubins* distance: the distance to a point q from a site s is the distance that a car would have to drive from s to q, without reversing, given its initial position and direction vector and with minimum turn radius equal to the magnitude of the vector [4];
- the *"catch"* distance for moving sites (see Figure 1f): the distance from a point q to a site s is defined as the distance one has to travel from q (at a fixed speed) to meet the moving site s as soon as possible, given its initial position, direction and speed[1].

The examples in Figures 1e and 1f in particular raise the question what properties the curvy bisectors have with these distance measures—where the distance contours are simply composed of (non-concentric) circular arcs.

2 The geometric L_0 distance

The L_p distance between two points a and b is given by $L_p(a - b)$ where $L_p((x, y)) = (|x|^p + |y|^p)^{1/p}$. The L_p distances are widely studied in computational geometry, in particular with $p \geq 1$, in which case the distance constitutes a metric. With the help of Vorosketch, Rolf Klein and I studied what happens if p drops to zero or even becomes negative [10]. We see that the values of the L_p distance function differ depending on whether p tends to zero from above or from below: If $x \neq 0$ and $y \neq 0$, then $\lim_{p\downarrow 0} L_p((x, y)) = \infty$ whereas $\lim_{p\uparrow 0} L_p((x, y)) = 0$. To verify this, define $r = y/x$; then $\lim_{p\to 0} L_p((x, y)) = x \lim_{p\to 0}(1 + r^p)^{1/p} = x \lim_{p\to 0} 2^{1/p}$. Nevertheless, we found that, at least for points in general position, the limit of the Voronoi diagram under the L_p distance as p approaches zero is well-defined. In other words, even though, for small values of $|p|$, the L_p and $L_{(-p)}$ distances have very different values, they induce almost identical Voronoi diagrams.

To be precise, observe that the terms $|x|^p$ and $|y|^p$ in the definition of L_p are undefined if $p < 0$ and $x = 0$ or $y = 0$, respectively. We extend the function L_p, for $p < 0$, to a continuous function on all arguments (x, y) by setting its value to zero if x or y is equal to zero. Thus, given two points a and b in the plane, we can calculate and compare their L_p distances to any other point in the plane, for any $p \neq 0$.

Figure 2 shows an example of the Voronoi diagram of two points under the L_p distance for two values of p very close to zero. The figure shows, in particular, the bisector $B_p(a, b)$ of two points $a = (a_x, a_y)$ and $b = (b_x, b_y)$, that is, it shows the set of points in the plane that are equidistant to a and b under the L_p distance. The bisector $B_p(a, b)$ separates the set $V_p(a, b)$ of points that are closer to a from the set $V_p(b, a)$ of points that are closer to b. We obtained the following result:

[1] The special case in which the sites have the same speed as the catchers, also has an interpretation in terms of illumination. In that case the distance is proportional to the Euclidean distance times the secant of the angle between sq and the site vector (provided the angle is less than $\pi/2$). This can be interpreted as the size, as seen from q, of a small unit-size diffuse linear light source at s whose normal is the site vector. The Voronoi diagram models from which site each part of the plane receives most light.

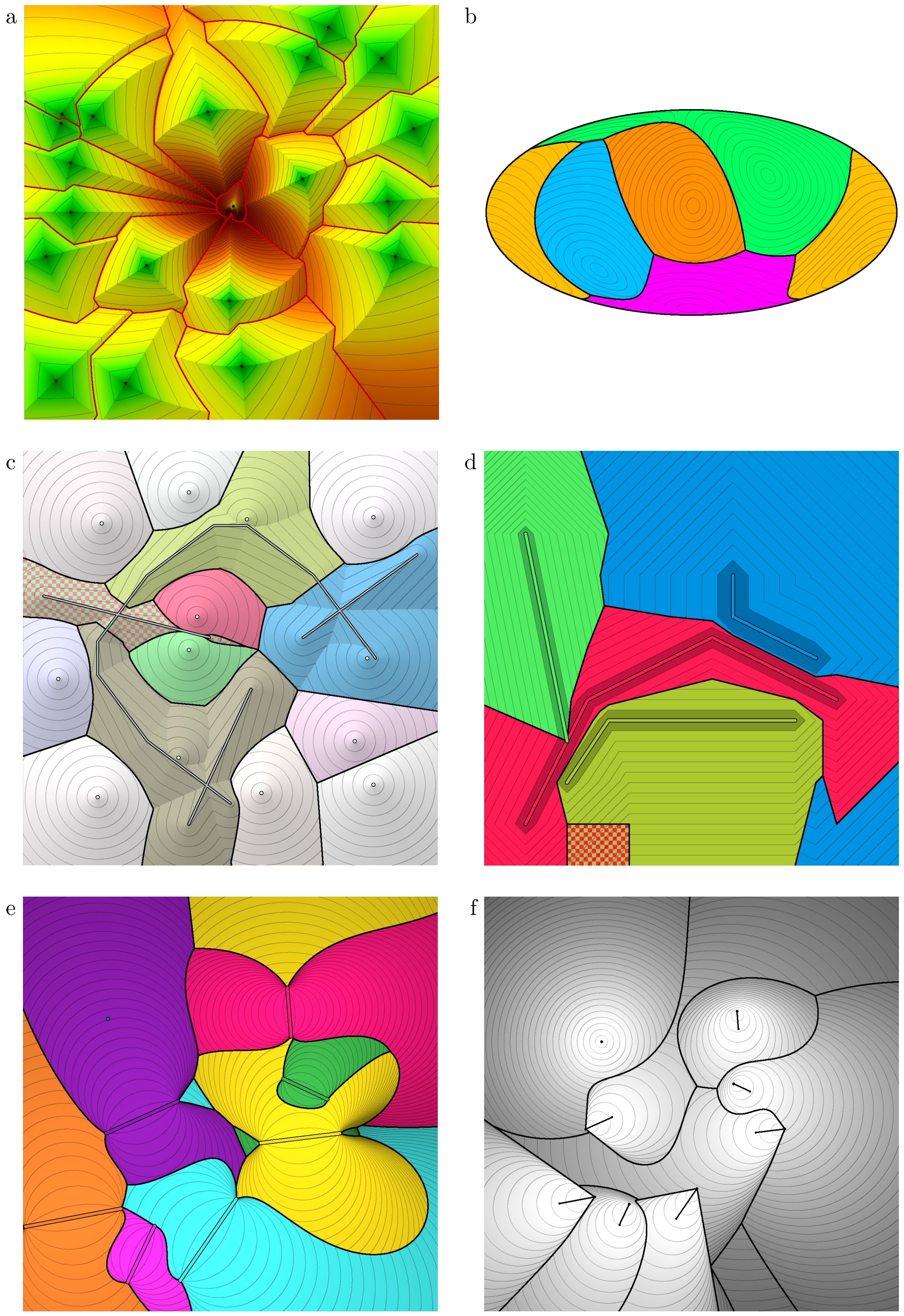


Figure 1 Voronoi diagrams of: (a) points by Köln distance; (b) points on a sphere in equal-area elliptical projection; (c) points by L_2 distance with highways; (d) multiplicatively weighted polylines by L_1 distance; (e) line segments by angular size; (f) moving points by catch distance.

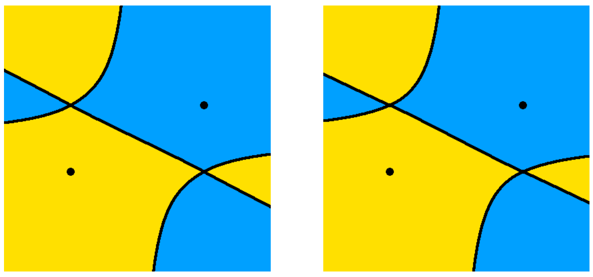


Figure 2 A bisector (in black) of two points under the L_p distance for $p = -0.05$ (left) and $p = 0.05$ (right), as computed by Vorosketch. The bisector divides the plane into two regions (points closer to a and points closer to b) that each seem to consist of three faces.

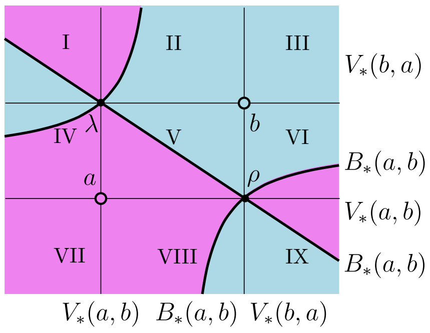


Figure 3 A bisector of two points a and b under the L_* distance.

▶ **Theorem 1.** *Let a and b be two point sites in the real plane with different X- and Y-coordinates. Then their bisector $B_p(a,b)$ and the sets $V_p(a,b)$ and $V_p(b,a)$ under the L_p distance converge, as p tends to zero from above or from below, to their bisector $B_*(a,b)$ and the sets $V_*(a,b)$ and $V_*(b,a)$ under the L_* distance as defined by $L_*((x,y)) = |xy|$.*

Note that $L_*(a-b)$ is simply the area of the axis-parallel bounding box of a and b. Let λ and ρ denote the other two vertices of their axis-parallel bounding box. The bisector $B_*(a,b)$ consists of the line through λ and ρ and of two hyperbola branches through λ and ρ, respectively, whose asymptotes are the vertical and horizontal lines through the bounding box's centre; see Figure 3. This can easily be verified by solving the equation $L_*(a-q) = L_*(b-q)$ for q in each of the nine regions that result from subdividing the plane by the axis-parallel lines through a and b. The bisector $B_*(a,b)$ divides the plane into six faces, such that each face is entirely contained in either $V_*(a,b)$ or $V_*(b,a)$.

It follows that for point sites in general position (that is, if no two points are on a common horizontal or vertical line[2]), the limit of their L_p Voronoi diagram as p tends to zero is well-defined and it equals the L_* Voronoi diagram. Thus, it appears that defining $L_0((x,y))$ as $L_*((x,y)) = |xy|$, which equals $\exp(\ln|x| + \ln|y|)$ if $x, y \neq 0$, constitutes a natural interpretation of $L_p((x,y)) = (|x|^p + |y|^p)^{1/p}$ for $p = 0$. Therefore we propose to call the L_* distance measure the *geometric L_0 distance*, and the resulting Voronoi diagram the *geometric L_0 Voronoi diagram.* Note that this definition is notably distinct from other,

[2] For a discussion of non-general position, see [10].

unrelated, definitions of L_0 that have been proposed[3], and whose Voronoi diagrams would be very different from the L_* Voronoi diagram. Figure 4 shows, for comparison, the L_p Voronoi diagram of the same sites for several values of p.

Our proof of Theorem 1 is based on the following approach. Assume that $|p| < 1$ is sufficiently small, but not zero. Consider two point sites a and b that do not lie on a common horizontal or vertical line. If we subject these points to translation, reflection in a coordinate axis, or reflection in the line $y = x$, the bisector under the L_p distance undergoes the same transformation. Therefore, to investigate the shape of the bisector, it suffices to consider two sites $a = (-u, -1)$ and $b = (u, 1)$ where $u \geq 1$. Figure 3 shows how the plane is divided into nine regions by the horizontal and vertical lines through a and b. Regions III and VII do not contain any point of $B_p(a, b)$. For each of the other regions, we can analyse, for any vertical line ℓ, the vertical distance between any point of $B_p(a, b) \cap \ell$ and the unique point of $B_*(a, b) \cap \ell$ within that region, and prove that the limit of this distance is zero as p tends to 0 from above or from below. Our proof can be found in detail in our manuscript on arXiv [10].

References

1 Carlos Alegría, Ioannis Mantas, Evanthia Papadopoulou, Marko Savic, Hendrik Schrezenmaier, Carlos Seara and Martin Suderland. The Voronoi diagram of rotating rays with applications to floodlight illumination. *Proc. 29th Eur. Symp. on Algorithms (ESA 2021)*, LIPIcs 204:5.1–16, 2021.

2 Franz Aurenhammer. Power diagrams: properties, algorithms and applications. *SIAM J. Comput.* 16(1):78–96, 1987.

3 Mark de Berg, Otfried Cheong, Marc van Kreveld and Mark Overmars. *Computational Geometry: algorithms and applications.* Springer, 2008.

4 Xuân-Nam Bui and Jean-Daniel Boissonnat. *Accessibility region for a car that only moves forwards along optimal paths.* Technical report RR-2181, inria-00074491, INRIA, 1994.

5 Yongxi Cheng, Bo Li and Yinfeng Xu. Semi Voronoi diagrams. *Proc. 9th Int. Conf. Comp. Geom., Graphs and Appl. (CGGA 2010)*, LNCS 7033:19–26, 2011.

6 Chenglin Fan, Jun Luo, Jinfei Liu and Yinfeng Xu. Half-plane Voronoi diagram. *Proc. 8th Int. Symp. Voronoi Diagrams in Sci. and Eng. (ISVD 2011)*, pages 127–133, 2011.

7 Laxmi P. Gewali, Alex C. Meng, Joseph S. B. Mitchell, and Simeon Ntafos. Path planning in $0/1/\infty$ weighted regions with applications. *INFORMS J. Comput.* 3(2):253–272, 1990.

8 Herman Haverkort. *Vorosketch*, version 0.31β.
From `http://herman.haverkort.net/vorosketch/`, 3 March 2023.

9 Herman Haverkort and Rolf Klein. Hyperbolae are the locus of constant angle difference. *CoRR (arXiv)* abs/2112.00454, 2021.

10 Herman Haverkort and Rolf Klein. The limit of L_p Voronoi diagrams as $p \to 0$ is the bounding-box-area Voronoi diagram. *CoRR (arXiv)* abs/2207.07377, 2022.

11 Hiroshi Imai, Masao Iri, and Kazuo Murota. Voronoi diagram in the Laguerre geometry and its applications. *SIAM J. Comput.* 14(1):93–105, 1985.

12 Rolf Klein. Voronoi diagrams in the Moscow metric. *Proc. 14th Int. Workshop Graph-Theoretic Concepts in Computer Sci. (WG 1988)*, LNCS 344:434–441, 1988.

13 Rolf Klein, Anne Driemel and Herman Haverkort. *Algorithmische Geometrie: Grundlagen, Methoden, Anwendungen.* Springer, 2022.

[3] `https://en.wikipedia.org/wiki/Lp_space#When_p_=_0`, retrieved 10 January 2023.

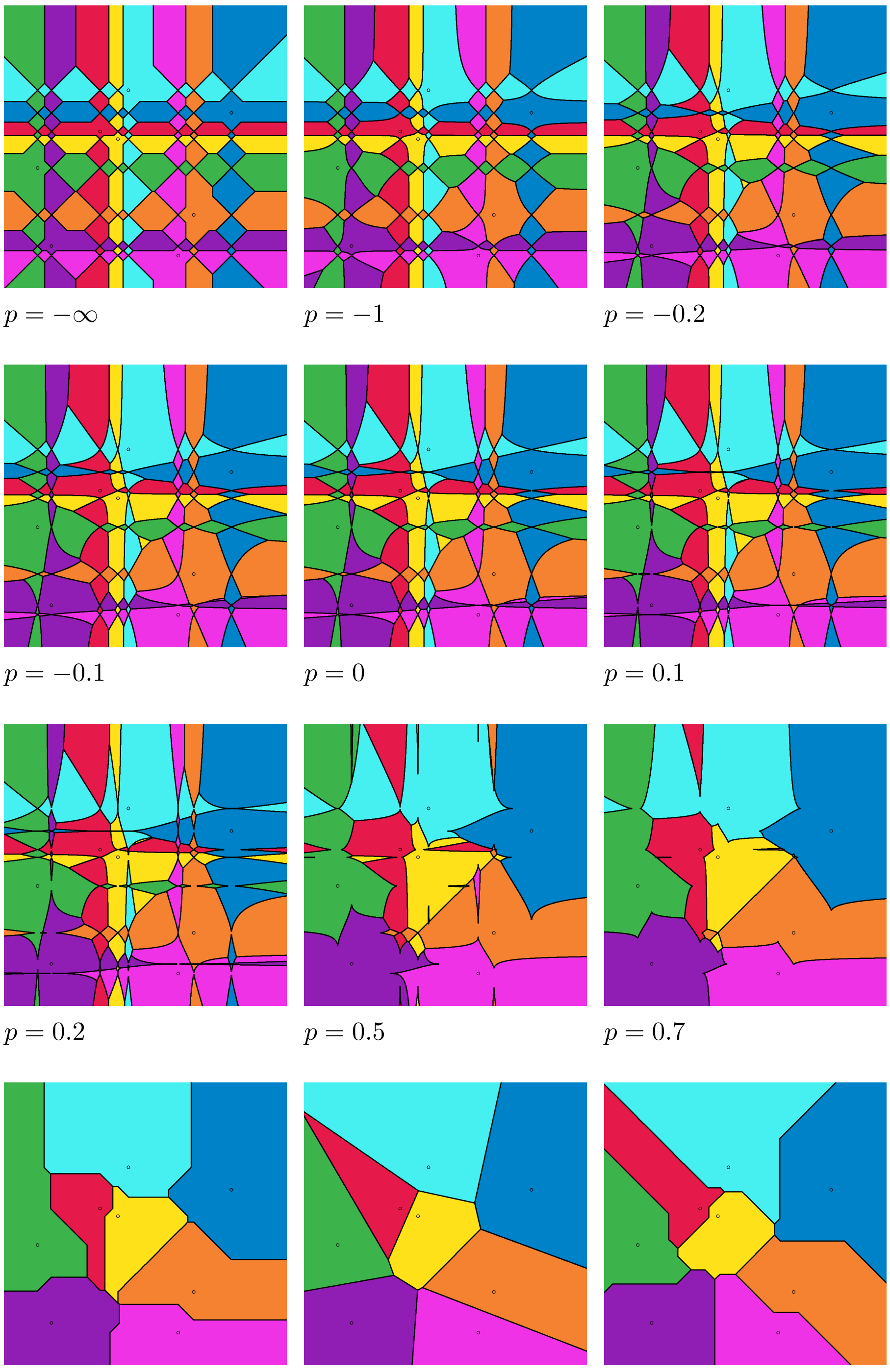


Figure 4 Voronoi Diagrams under the L_p distance for various values of p. Note that features narrower than a pixel might not have been detected completely.

Applying The Pebble Game Algorithm to Rod Configurations [*]

Signe Lundqvist[1], Klara Stokes[1], and Lars-Daniel Öhman[1]

1 **Department of Mathematics and Mathematical Statistics, Umeå University**
`signe.lundqvist@umu.se`
`klara.stokes@umu.se`
`lars-daniel.ohman@umu.se`

Abstract

We present results on rigidity of structures of rigid rods connected in joints: rod configurations. The underlying combinatorial structure of a rod configuration is an incidence structure. Our aim is to find simple ways of determining which rod configurations admit non-trivial motions, using the underlying incidence structure.

Rigidity of graphs in the plane is well understood. Indeed, there is a polynomial time algorithm for deciding whether most realisations of a graph are rigid. One of the results presented here equates rigidity of sufficiently generic rod configurations to rigidity of a related graph. As a consequence, it is possible to determine the rigidity of rod configurations using the previously mentioned polynomial time algorithm. We use this to show that all v_3-configurations on up to 15 points and all triangle-free v_3-configurations on up to 20 points are rigid in regular position, if such a realisation exists. We also conjecture that the smallest v_3-configuration that is flexible in regular position is a previously known 28_3-configuration.

Related Version arXiv:2112.01960

1 Introduction

A rod configuration is a realisation of a hypergraph as points and lines in the plane. We are interested in motions of rod configurations, where the motions are assumed to preserve collinearity of points incident to the same line, and the pairwise distance between points incident to the same line. Theorem 2.1 says that sufficiently generic rod configurations realising an incidence geometry are rigid if and only if almost all realisations of an associated graph are rigid. Determining whether almost all realisations of a given graph are rigid can be done in polynomial time, using for example the pebble game algorithm [7]. As a consequence of our result, determining the rigidity of sufficiently generic rod configurations can also be done efficiently using the pebble game algorithm.

We consider only the two-dimensional setting. Throughout, $S = (P, L, I)$ will denote a connected rank two incidence geometry, with point set P, line set L and incidence set I. For basic definitions and more on incidence geometries, see for example [3], where rank two incidence geometries are referred to as point-line geometries. All graphs are assumed to be simple. A *linear realisation* of an incidence geometry is a realisation of an incidence geometry as points and straight lines in the plane. Let $S = (P, L, I)$ be an incidence geometry, and ρ an assignment of a line slope f_j to each element $\ell_j \in L$. A linear realisation of an incidence geometry $S = (P, L, I)$ with line slopes given by ρ is an assignment of a pair of

[*] This work has been supported by the Knut and Alice Wallenberg Foundation Grant 2020.0001 and 2020.0007.

point coordinates (x_i, y_i) to each element $p_i \in P$ and a y-intercept h_j to each element $\ell_j \in L$ such that

$$f_j x_i + y_i + h_j = 0 \tag{1}$$

whenever $(p_i, \ell_j) \in I$. Finding a linear realisation of an incidence geometry with an assigned set of line slopes ρ amounts to solving a system of I equations of the form (1). Let $M(S, \rho)$ denote that system of equations. A linear realisation is *trivial* if all points are assigned the same point coordinates. There are trivial linear realisations of any incidence geometry with any line slopes. A linear realisation is *proper* if all points are assigned distinct coordinates. Not all incidence geometries have proper linear realisations. The Fano plane, for instance, does not have proper linear realisations for any choice of line slopes.

In 1989, Whiteley proved that an incidence geometry has proper linear realisations for almost all choices of line slopes if and only if for any subset $I' \subseteq I$

$$|I'| \leq |L'| + 2|P'| - 3 \tag{2}$$

where $P' \times L' \subseteq P \times L$ is the support of I' [11]. Specifically, incidence geometries with sets of incidences that satisfy (2) have proper linear realisations if the line slopes are chosen to be algebraically independent over $\mathbb{Q}$. Incidence geometries with sets of incidences that do not satisfy (2) can have proper linear realisations for some choices of line slopes ρ, however, the line slopes given by ρ must then be algebraically dependent over $\mathbb{Q}$. Furthermore, the rows of $M(S, \rho)$ would necessarily be dependent [11]. We say that a proper linear realisation of an incidence geometry $S = (P, L, I)$ is *regular* if every subset of I' satisfying (2) corresponds to independent rows of $M(S, \rho)$. Intuitively, a *rod configuration* is a proper linear realisation where the lines move like rigid bodies. We will give a more formal definition in Section 2. For the remainder of this section we will focus on the special case of graphs.

Given a graph $G = (V, E)$, any map $f : V \to \mathbb{R}^2$ gives a proper linear realisation of G. Linear realisations of graphs are more commonly known as *bar-and-joint frameworks* or *frameworks.* A *continuous motion* of a bar-and-joint framework is a motion of the vertices which preserves the distance between any pair of adjacent vertices. A bar-and-joint framework is *rigid* if the only continuous motions of the framework are translations and rotations, otherwise it is *flexible.* An *infinitesimal motion* of a bar-and-joint framework is an assignment $m : V \to \mathbb{R}^2$ such that $\langle m(i) - m(j), f(i) - f(j) \rangle = 0$, whenever $(i, j) \in E$, where $\langle -, - \rangle$ denotes the standard scalar product in $\mathbb{R}^2$. A bar-and-joint framework is *infinitesimally rigid* if the only infinitesimal motions of the framework are linearisations of translations and rotations. Otherwise it is *infinitesimally flexible.*

It is known that almost all realisations of a given graph will have the same rigidity properties [5]. In particular, all bar-and-joint framework of a graph such that the vertices are given coordinates that are algebraically independent over $\mathbb{Q}$ (*generic* frameworks) have the same rigidity properties. We can therefore say that a graph is *generically rigid* in the plane if all its generic frameworks are infinitesimally rigid, otherwise we say that the graph is *generically flexible.*

A graph is $(2, 3)$-sparse if $|E'| \leq 2|V'| - 3$ for all subsets $E' \subseteq E$, where V' is the set of vertices generated by E'. A graph is $(2, 3)$-tight if it is $(2, 3)$-sparse and $|E| = 2|V| - 3$. A classical result in rigidity theory says that a graph is generically rigid in $\mathbb{R}^2$ if and only if it has a $(2, 3)$-tight spanning subgraph [8]. The $(2, 3)$-tight graphs, or Laman graphs, are *minimally rigid*, in the sense that they are generically rigid, but removing any edge results in a generically flexible graph.

There are several algorithms for determining generic rigidity, or $(2,3)$-sparsity of graphs. For the experiment in Section 4 we implemented the pebble game, which is briefly outlined below. However, there is an algorithm which is faster at determining $(2,3)$-sparsity [4]. The pebble game algorithm, introduced by Jacobs and Hendrickson, can determine generic rigidity of graphs [7]. The input of the pebble game algorithm is a graph $G = (V, E)$ and an ordering of E. The output of the algorithm is a set of accepted edges, which generate a spanning $(2,3)$-sparse subgraph. In the specified ordering, the algorithm checks whether the graph generated by the current edge and the previously accepted edges is $(2,3)$-sparse. If it is, then the current edge is accepted. From the end state of the algorithm, it is possible to determine whether the graph is generically flexible or generically rigid. Furthermore, it is possible to tell from the end state of the algorithm whether the graph is $(2,3)$-tight.

2 Theoretical Results

Given an incidence geometry $S = (P, L, I)$ we define a *cone graph* of S, denoted $G^C(S)$, to be a graph with a vertex for each $p \in P$ and a vertex for each $\ell \in L$, and edge set consisting of edges from the vertex representing a line ℓ to all vertices representing points incident to ℓ, and edges $\{(p, q)\}_{(q,\ell)\in I}$, where p is some chosen vertex representing a point incident to ℓ. We can also define an incidence geometry, the cone incidence geometry, denoted $S^C = (P^C, L^C, I^C)$, by $P^C = P \cup L$ and $L^C = L \cup \{(p, v_\ell) \mid (p, \ell) \in I\}$, where v_ℓ is the element of P^C corresponding to ℓ. The incidences in I^C are defined by inclusion, i.e. $p \in P^C$ is incident to $\ell \in L^C$ if $p \in \ell$. Cone graphs are not uniquely defined. However, if one cone graph of an incidence geometry S is generically rigid in the plane, then all cone graphs of the incidence geometry are generically rigid in the plane [10].

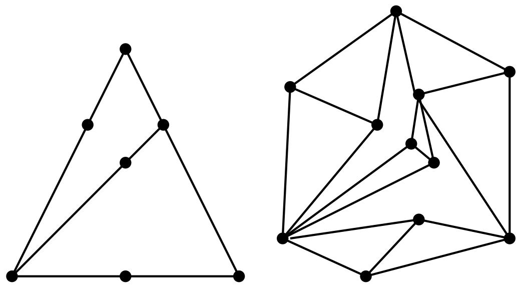


Figure 1 A rod configuration realising an incidence geometry S and a cone graph $G^C(S)$ of S.

Suppose that there is some proper linear realisation of S with line slopes given by ρ. A proper linear realisation of S assigns point coordinates to elements of P. Define a (non-generic) bar-and-joint framework of a cone graph $G^C(S)$ by assigning coordinates to the vertices of $G^C(S)$ representing elements of L, so that the vertices corresponding to elements of L are not on the lines representing elements of L in the linear realisation. Note that such an assignment defines a proper linear realisation of S^C with line slopes given by ρ', such that ρ' restricted to L is ρ. A *rod configuration* is a proper linear realisation ρ of S, which is *continuously* or *infinitesimally rigid* if some cone graph of S has a continuously or infinitesimally rigid bar-and-joint framework in the special position defined by ρ respectively. Otherwise it is *infinitesimally flexible*. A rod configuration is *minimally rigid* if it is rigid, but removing any rod results in a flexible rod configuration. We say that a rod configuration is *regular* if some linear realisation of S^C with line slopes restricting to ρ on L is regular.

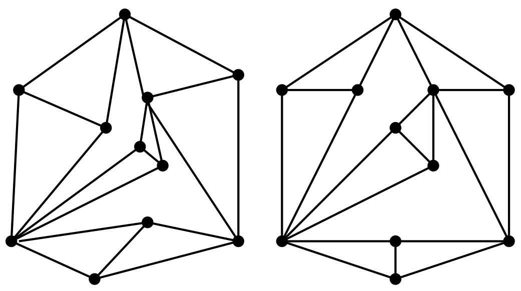


Figure 2 The cone graph $G^C(S)$ from Figure 1 and the non-generic bar-and-joint framework of $G^C(S)$ given by the linear realisation of S drawn in Figure 1.

▶ **Theorem 2.1.** *Let $S = (P, L, I)$ be an incidence geometry such that S^C has a regular proper linear realisation. Then all cone graphs of S are rigid in $\mathbb{R}^2$ if and only if all regular rod configurations realising S in $\mathbb{R}^2$ are infinitesimally rigid.*

The full proof of Theorem 2.1 can be found in a recent preprint [10], but we give a very brief sketch here. That $G^C(S)$ is rigid in the plane if all regular rod configurations realising S are infinitesimally rigid follows from the fact that if some framework of a graph is infinitesimally rigid, then all generic frameworks of that graph are rigid [5]. To prove the other direction of Theorem 2.1, we recursively construct a spanning $(2, 3)$-sparse subgraph of a cone graph $G^C(S)$. If $G^C(S)$ is generically rigid in $\mathbb{R}^2$, then the constructed $(2, 3)$-sparse subgraph is $(2, 3)$-tight and therefore minimally rigid in generic position. To complete the proof, we prove, using a result of Whiteley [11], that the constructed minimally rigid subgraph of $G^C(S)$ remains infinitesimally rigid in the non-generic position defined by any regular proper linear realisation of S^C. Since a spanning subgraph is infinitesimally rigid in this position, so is $G^C(S)$.

3 An algorithm for testing rigidity of rod configurations

As a consequence of Theorem 2.1, we get an algorithm for determining infinitesimal rigidity of regular rod configurations realising incidence geometries.

▶ **Algorithm 3.1.** *Given an incidence geometry S, create a cone graph $G^C(S)$ by adding a vertex v_p for each $p \in P$ and a vertex v_ℓ for each $\ell \in L$. Then add an edge (v_p, v_l) whenever $(p, \ell) \in I$. Finally pick an arbitrary ordering of P. Add edges $\{(v_p, v_q)\}_{(q,\ell)\in I}$, where p is the first element of P in the chosen ordering to be incident to ℓ.*

We can then apply the pebble game algorithm to $G^C(S)$. The output of the pebble game algorithm is a spanning $(2, 3)$-sparse subgraph G' of $G^C(S)$. If G' is $(2, 3)$-tight, then $G^C(S)$ is generically rigid, so regular rod configurations realising S are infinitesimally rigid, by Theorem 2.1. If G' is not $(2, 3)$-tight, then it follows from Theorem 2.1 that regular rod configurations realising S are infinitesimally flexible.

Whiteley proved that if $G^C(S)$ is generically minimally rigid, then regular rod configurations realising that incidence geometry are infinitesimally rigid [11]. If $G^C(S)$ is not minimally rigid, then testing minimal rigidity can be done by in turn removing each line of the rod configuration, and running Algorithm 3.1 to test whether the rod configuration remains rigid when the line is removed.

The pebble game algorithm runs in $\mathcal{O}(ve)$-time, where v and e are the numbers of vertices and edges of the input graph respectively [7]. The cone graph of an incidence geometry $S = (P, L, I)$ has one vertex for each element in P, and one vertex for each element of L. The cone graph S has an edge (p, c_ℓ) for each incidence (p, ℓ), and $P(\ell) - 1$ edges corresponding to a line $\ell \in L$, where $P(\ell)$ denotes the number of points incident to ℓ. In total, the cone graph has $|P| + |L|$ vertices, and $2|I| - |L|$ edges. Algorithm 3.1 therefore runs in $\mathcal{O}((|P| + |L|)(2|I| - |L|))$-time. Since it is possible to bound both the number of incidences and the number of points by constants times the number of lines, the algorithm runs in $\mathcal{O}(|L|^2)$-time.

To determine whether a rod configuration is *minimally* rigid, we may need to run Algorithm 3.1 $|L|$ times. With this approach, determining whether a rod configuration is minimally rigid can be done in $\mathcal{O}(|L|^3)$-time.

4 A Computational Experiment

A v_3-configuration is an incidence geometry with v points and v lines, such that each line is incident to 3 points and each point is incident to 3 lines. For more on v_k-configurations see for example [6].

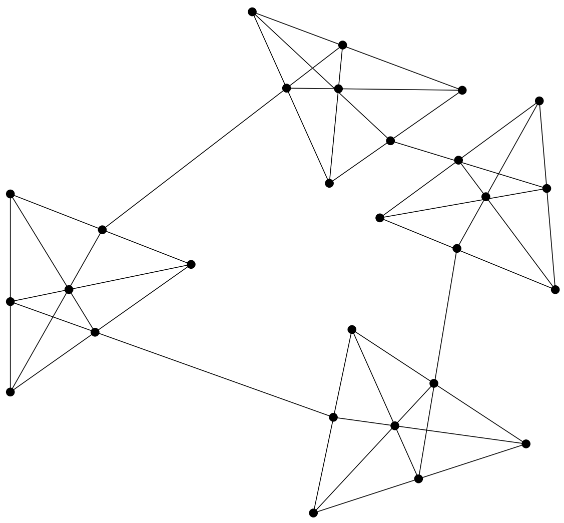


Figure 3 The smallest known flexible v_3-configuration.

In previous work, we found an infinite family of flexible v_3-configurations [9]. The smallest flexible v_3-configuration we found is a 28_3-configuration (see Figure 3). To test whether there exist smaller flexible v_3-configurations, we applied Algorithm 3.1 to small v_3-configurations. First, we used Orbiter [2] to generate all combinatorial v_3-configurations for $9 \leq v \leq 15$. There were 269120 v_3-configurations generated for $9 \leq v \leq 15$, 245342 of which were 15_3-configurations. The 16_3-configurations were too many to handle. We also used Orbiter to generate all triangle-free v_3-configurations for $16 \leq v \leq 20$. There were 181 triangle-free v_3-configurations for $16 \leq v \leq 20$, 162 of which were 20_3-configurations. We then created a cone graph of each v_3-configuration and applied the pebble game to it, in order to test whether regular rod configurations realising the v_3-configuration are infinitesimally flexible. Orbiter generates also disconnected incidence geometries, which are flexible in the trivial way that the two components can move in relation to each other. We are interested in flexible connected incidence geometries.

Out of the v_3-configurations that we applied the algorithm to, the only ones that had flexible cone graphs were a 14_3-configuration, which consisted of two copies of the Fano plane, and a 15_3-configuration which consisted of one copy of the Fano plane and one copy of the Möbius-Kantor configuration. Both of these v_3-configurations are disconnected. Hence, out of the connected v_3-configurations that we tested, none have flexible realisations as regular rod configurations. We did not manage to test all v_3 configurations for $9 \leq v \leq 27$, but we think it is reasonable to believe that the smallest flexible v_3-configuration is the flexible 28_3-configuration that we previously found.

5 Conclusion

Theorem 2.1 gives an efficient way of determining rigidity of regular rod configurations. However, in general there is no known way to efficiently determine whether an incidence geometry has realisations as regular rod configurations. Finding a maximum size subset of incidences satisfying (2) is known to be NP-hard , which suggests that deciding whether a proper linear realisation is regular is also difficult [1]. In general, even finding proper linear realisations of incidence geometries is not easy. Finding regular proper linear realisations of incidence geometries is therefore an interesting, but potentially difficult problem. We also have no examples of incidence geometries that have proper linear realisations, but no *regular* proper linear realisations. Finding such examples would be interesting.

The algorithm to determine minimal rigidity of rod configurations is based on the algorithm for determining ridigity of rod configurations. While determining minimal rigidity can still be done fairly efficiently, it might still be interesting to get a better understanding of minimal rigidity of rod configurations.

Acknowledgments. We thank Anton Betten for his valuable help with Orbiter.

References

1 Alex R. Berg and Tibor Jordán. Algorithms for graph rigidity and scene analysis. In *Algorithms—ESA 2003*, volume 2832 of *Lecture Notes in Comput. Sci.*, pages 78–89. Springer, Berlin, 2003. URL: `https://doi.org/10.1007/978-3-540-39658-1_10`, `doi:10.1007/978-3-540-39658-1\_10`.

2 Anton Betten. The Orbiter ecosystem for combinatorial objects. In *ISSAC 2020—Proceedings of the 45th International Symposium on Symbolic and Algebraic Computation*, pages 30–37. ACM, New York, 2020. `doi:10.1145/3373207.3403984`.

3 Bart De Bruyn. *An introduction to incidence geometry.* Frontiers in Mathematics. Birkhäuser/Springer, Cham, 2016. `doi:10.1007/978-3-319-43811-5`.

4 Harold N. Gabow and Herbert H. Westermann. Forests, frames, and games: algorithms for matroid sums and applications. *Algorithmica*, 7(5-6):465–497, 1992. `doi:10.1007/BF01758774`.

5 Herman Gluck. Almost all simply connected closed surfaces are rigid. In *Geometric topology (Proc. Conf., Park City, Utah, 1974)*, Lecture Notes in Math., Vol. 438, pages 225–239. Springer, Berlin, 1975.

6 Branko Grünbaum. *Configurations of points and lines*, volume 103 of *Graduate Studies in Mathematics*. American Mathematical Society, Providence, RI, 2009. `doi:10.1090/gsm/103`.

7 Donald J. Jacobs and Bruce Hendrickson. An algorithm for two-dimensional rigidity percolation: the pebble game. *J. Comput. Phys.*, 137(2):346–365, 1997. `doi:10.1006/jcph.1997.5809`.

8 Gerard Laman. On graphs and rigidity of plane skeletal structures. *J. Engrg. Math.*, 4:331–340, 1970. `doi:10.1007/BF01534980`.

9 Signe Lundqvist, Klara Stokes, and Lars-Daniel Öhman. Exploring the infinitesimal rigidity of planar configurations of points and rods. preprint, 2021. https://arxiv.org/pdf/2112.01960.pdf.

10 Signe Lundqvist, Klara Stokes, and Lars-Daniel Öhman. When is a rod configuration infinitesimally rigid? preprint, 2022. https://arxiv.org/pdf/2112.01960.pdf.

11 Walter Whiteley. A matroid on hypergraphs, with applications in scene analysis and geometry. *Discrete Comput. Geom.*, 4(1):75–95, 1989. `doi:10.1007/BF02187716`.

The Number of Edges in Maximal 2-planar Graphs[*]

Michael Hoffmann and Meghana M. Reddy

Department of Computer Science, ETH Zürich, Switzerland,
`{hoffmann, meghana.mreddy}@inf.ethz.ch`

Abstract

A graph is 2-*planar* if it has local crossing number two, that is, it can be drawn in the plane such that every edge has at most two crossings. A graph is *maximal* 2-*planar* if no edge can be added such that the resulting graph remains 2-planar. A 2-planar graph on n vertices has at most $5n-10$ edges, and some (maximal) 2-planar graphs—referred to as *optimal* 2-*planar*—achieve this bound. However, in strong contrast to maximal planar graphs, a maximal 2-planar graph may have fewer than the maximum possible number of edges. In this paper, we determine the minimum edge density of maximal 2-planar graphs by proving that every maximal 2-planar graph on $n \geq 5$ vertices has at least $2n$ edges. We also show that this bound is tight, up to an additive constant.

1 Introduction

Maximal planar graphs a.k.a. (combinatorial) triangulations are a rather important and well-studied class of graphs with a number of nice and useful properties. To begin with, the number of edges is uniquely determined by the number of vertices, as every maximal planar graph on $n \geq 3$ vertices has $3n-6$ edges. It is natural to wonder if a similar statement can be made for the various families of near-planar graphs, which have received considerable attention over the past decade; see, e.g. [7, 8].

In this paper we focus on k-planar graphs, specifically for $k = 2$. These are graphs with local crossing number at most k, that is, they admit a drawing in $\mathbb{R}^2$ where every edge has at most k crossings. The maximum number of edges in a k-planar graph on n vertices increases with k, but the exact dependency is not known. A general upper bound of $O(\sqrt{k}n)$ is known due to Ackerman and Pach and Tóth [1, 11] for graphs that admit a *simple* k-plane drawing, that is, a drawing where every pair of edges has at most one common point. A 1-planar graph on n vertices has at most $4n-8$ edges and there are infinitely many *optimal* 1-planar graphs that achieve this bound, as shown by Bodendiek, Schumacher, and Wagner [5]. A 2-planar graph on n vertices has at most $5n-10$ edges and there are infinitely many *optimal* 2-planar graphs that achieve this bound, as shown by Pach and Tóth [11]. In fact, there are complete characterizations, for optimal 1-planar graphs by Suzuki [13] and for optimal 2-planar graphs by Bekos, Kaufmann, and Raftopoulou [4].

Much less is known about *maximal* k-planar graphs, that is, graphs for which adding any edge results in a graph that is not k-planar anymore. In contrast to planar graphs, where maximal and optimal coincide, the difference between maximal and optimal can be quite large for k-planar graphs, even—perhaps counterintuitively—maximal k-planar graphs for $k \geq 1$ may have fewer edges than maximal planar graphs on the same number of vertices. Hudák, Madaras, and Suzuki [9] describe an infinite family of maximal 1-planar graphs with only $8n/3 + O(1) \approx 2.667n$ edges. An improved construction with $45n/17 + O(1) \approx 2.647n$ edges was given by Brandenburg, Eppstein, Gleißner, Goodrich, Hanauer, and Reislhuber [6]

[*] M. Hoffmann and M. M. Reddy were supported by the Swiss National Science Foundation within the collaborative DACH project *Arrangements and Drawings* as SNSF Project 200021E-171681.

who also established a lower bound by showing that every maximal 1-planar graph has at least $28n/13 - O(1) \approx 2.153n$ edges. Later, this lower bound was improved to $20n/9 \approx 2.22n$ by Barát and Tóth [3].

Maximal 2-planar graphs were studied by Auer, Brandenburg, Gleißner, and Hanauer [2] who constructed an infinite family of maximal 2-planar graphs with n vertices and $387n/147 + O(1) \approx 2.63n$ edges.[1] We are not aware of any nontrivial lower bounds on the number of edges in maximal k-planar graphs, for $k \geq 2$.

Results. In this paper, we give tight bounds on the number of edges in maximal 2-planar graphs, up to an additive constant.

▶ **Theorem 1.** *Every maximal 2-planar graph on $n \geq 5$ vertices has at least $2n$ edges.*

▶ **Theorem 2.** *There exists a constant $c \in \mathbb{N}$ such that for every $n \in \mathbb{N}$ there exists a maximal 2-planar graph on n vertices with at most $2n + c$ edges.*

2 Preliminaries

A drawing is *simple* if every pair of edges has at most one common point. A drawing is *k-plane*, for $k \in \mathbb{N}$, if every edge has at most k crossings. A graph is *k-planar* if it admits a k-plane drawing. A graph is *maximal k-planar* if no edge can be added to it so that the resulting graph is still k-planar.

To analyze a k-planar graph one often analyzes one of its k-plane drawings. It is, therefore, useful to impose additional restrictions on this drawing if possible. One such restriction is to consider a *crossing-minimal* k-plane drawing, that is, a drawing that minimizes the total number of edge crossings among all k-plane drawings of the graph. For small k, such a drawing is always simple; for $k \geq 4$ this is not the case in general [12, Footnote 112].

▶ **Lemma 3** (Pach, Radoičić, Tardos, and Tóth [10, Lemma 1.1])**.** *For $k \leq 3$, every crossing-minimal k-plane drawing is simple.*

In figures, we use the following convention to depict edges: Uncrossed edges are shown green, singly crossed edges are shown purple, doubly crossed edges are shown blue, and edges for which the number of crossings is undetermined are shown black.

3 The Lower Bound

In this section we briefly describe our lower bound on the edge density of maximal 2-planar graphs by analyzing the distribution of vertex degrees. As we aim for a lower bound of $2n$ edges, we want to show that the average vertex degree is at least four. Then, the density bound follows by the handshaking lemma. However, maximal 2-planar graphs may contain vertices of degree less than four. By the following property (whose proof is deferred to the full version), we know that the degree of every vertex is at least two. But degree two vertices, so-called *hermits*, may exist, as well as vertices of degree three.

▶ **Lemma 4.** *For $k \leq 2$, every maximal k-planar graph on $n \geq 3$ vertices is 2-connected.*

[1] Maximality is proven via uniqueness of the 2-plane drawing of the graph. However, there is no explicit proof of the uniqueness in this short abstract.

In order to lower bound the average degree by four, we employ a charging scheme where we argue that every *low-degree* vertex, that is, every vertex of degree two and three claims a certain number of halfedges at an adjacent *high-degree* vertex, that is, a vertex of degree at least five. Claims are exclusive, that is, every halfedge at a high-degree vertex can be claimed at most once. We use the term *halfedge* because the claim is not on the whole edge but rather on its incidence to one of its high-degree endpoints. The incidence at the other endpoint may or may not be claimed independently (by another vertex). For an edge uv we denote by $\overrightarrow{uv}$ the corresponding halfedge at v and by $\overrightarrow{vu}$ the corresponding halfedge at u. Vertices of degree four have a special role, as they are neither low– nor high-degree. However, a vertex of degree four that is adjacent to a hermit is treated like a low-degree vertex. More precisely, our charging scheme works as follows:

(C1) Every hermit claims two halfedges at each high-degree neighbor.
(C2) Every degree three vertex claims three halfedges at some high-degree neighbor.
(C3) Every degree four vertex that is adjacent to a hermit h claims two halfedges at some neighbor v of degree ≥ 6. Further, the vertices h and v are adjacent, so h also claims two halfedges at v by (C1). If $\deg(v) = 6$, then v is adjacent to exactly one hermit.
(C4) At most one vertex claims (one or more) halfedges at a degree five vertex.

We state some useful properties of low-degree vertices. Then we present the proof of Theorem 1 in Section 3.3. The validity of our charging scheme is deferred to the full version.

3.1 Hermits and degree four vertices

▶ **Lemma 5.** *Let h be a hermit and let x, y be its neighbors in G. Then x and y are adjacent in G and all three edges xy, hx, hy are uncrossed in D. Further,* $\deg(x) \geq 4$ *and* $\deg(y) \geq 4$.

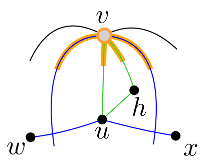
We refer to the edge xy as the *base* of the hermit h, which *hosts* h.

▶ **Lemma 6.** *Let G be a maximal 2-planar graph on $n \geq 5$ vertices. Every edge of G hosts at most one hermit. Further, a vertex of degree i in G is adjacent to at most $\lfloor i/3 \rfloor$ hermits.*

By Lemma 5, both neighbors of a hermit have degree at least four. A vertex is of type *T4-H* if it has degree four and it is adjacent to a hermit. The following lemma characterizes these vertices and ensures that every hermit has at least one high-degree neighbor.

▶ **Lemma 7.** *Let u be a T4-H vertex with neighbors h, v, w, x in G such that h is a hermit and v is the second neighbor of h. Then both uw and ux are doubly crossed in D, and the two faces of $D \setminus h$ incident to uv are triangles that are bounded by (parts of) edges incident to u and doubly crossed edges incident to v. Furthermore, we have* $\deg(v) \geq 6$, *and if* $\deg(v) = 6$, *then h is the only hermit adjacent to v in G.*

In our charging scheme, each hermit h claims two halfedges at each high-degree neighbor v: the halfedge $\overrightarrow{hv}$ and the halfedge $\overrightarrow{uv}$, where uv denotes the edge that hosts h. Each T4-H vertex u claims the two doubly crossed halfedges at v that bound the triangular faces incident to uv in D.

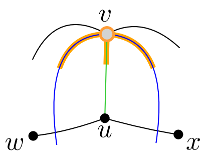


3.2 Degree three vertices

We distinguish four different types of degree three vertices in G, depending on their neighborhood and on the crossings on their incident edges in D. Consider a degree three vertex u in G. Every vertex is incident to at least one uncrossed edge in D (the proof is deferred to the full version).

T3-1: exactly one uncrossed edge. The two other edges incident to u are crossed.

▶ **Lemma 8.** *Let u be a T3-1 vertex with neighbors v, w, x in G such that the edge uv is uncrossed in D. Then the two faces of D incident to uv are triangles that are bounded by (parts of) edges incident to u and doubly crossed edges incident to v. Furthermore, we have* $\deg(v) \geq 5$.

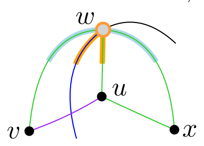


In our charging scheme, each T3-1 vertex u claims three halfedges at its adjacent high-degree vertex v: the uncrossed halfedge $\overrightarrow{uv}$ along with the two neighboring halfedges at v, which are doubly crossed by Lemma 8.

T3-2: exactly two uncrossed edges. The third edge incident to u is crossed.

▶ **Lemma 9.** *Let u be a T3-2 vertex with neighbors v, w, x s.t. the edge uv is crossed. Then uv is singly crossed by a doubly crossed edge wb in D,* $\deg(w) \geq 5$ *and* $\min\{\deg(v), \deg(x)\} \geq 4$.

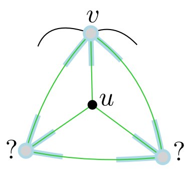
A halfedge $\overrightarrow{wx}$ is *peripheral* for a vertex u of G if (1) u is a common neighbor of w and x; (2) $\deg(w) \geq 5$; and (3) $\deg(x) \geq 4$. In our charging scheme, every T3-2 vertex u claims three halfedges at the adjacent high-degree vertex w: the halfedge $\overrightarrow{uw}$, the doubly crossed halfedge $\overrightarrow{bw}$, and one of the uncrossed peripheral halfedges $\overrightarrow{vw}$ or $\overrightarrow{xw}$. While the former two are closely tied to u, the situation is more complicated for the latter two halfedges. Eventually, we argue that u can exclusively claim (at least) one of the two peripheral halfedges. But for the time being we say that it *assesses* both of them and these edges are depicted in lightblue.



T3-3: all three incident edges uncrossed. We say that such a vertex is of type T3-3. As an immediate consequence of Lemma **??** each T3-3 vertex u together with its neighbors $\mathrm{N}(u)$ induces a plane K_4 in D. We further distiguish two subtypes of T3-3 vertices.

The first subtype accounts for the fact that there may be two adjacent T3-3 vertices in D. We refer to such a pair as an *inefficient hermit* and a T3-3 vertex that is part of an inefficient hermit is called a *T3-3 hermit.* T3-3 hermits behave similar to hermits, we defer the details of T3-3 hermits to the full version. The second subtype is formed by those T3-3 vertices that are not T3-3 hermits; we call them *T3-3 minglers.* All neighbors of a T3-3 mingler have degree at least four.

▶ **Lemma 10.** *Let u be a T3-3 mingler in D, and let v, w, x be its neighbors. Then each of v, w, x has degree at least four. Further, at least one vertex among v, w, x has degree at least six, or at least two vertices among v, w, x have degree at least five.*



Let Q denote the plane K_4 induced by u, v, w, x in D. The T3-3 mingler u claims the three halfedges of Q at one of its high-degree neighbors. That is, the vertex u assesses all of its peripheral halfedges at high-degree neighbors.

3.3 Proof of Theorem 1

Let G be a maximal 2-planar graph on $n \geq 5$ vertices, and let m denote the number of edges in G. We denote by v_i the number of vertices of degree i in G. By Lemma 4 we know that G

is 2-connected and, therefore, we have $v_0 = v_1 = 0$. Thus, we have

$$n = \sum_{i=2}^{n-1} v_i \text{ and by the Handshaking Lemma } 2m = \sum_{i=2}^{n-1} i \cdot v_i. \tag{1}$$

Vertices of degree four or higher can be adjacent to hermits. Let $v_i^{\mathrm{h}j}$ denote the number of vertices of degree i incident to j hermits in G. By Lemma 6 we have

$$v_i = \sum_{j=0}^{\lfloor i/3 \rfloor} v_i^{\mathrm{h}j} \qquad \text{for all } i \geq 3. \tag{2}$$

By Lemma 5 both neighbors of a hermit have degree at least four. Thus, double counting the edges between hermits and their neighbors we obtain

$$2v_2 \leq v_4^{\mathrm{h1}} + v_5^{\mathrm{h1}} + v_6^{\mathrm{h1}} + 2v_6^{\mathrm{h2}} + v_7^{\mathrm{h1}} + 2v_7^{\mathrm{h2}} + 2v_8 + v_9^{\mathrm{h1}} + 2v_9^{\mathrm{h2}} + 3v_9^{\mathrm{h3}} + \sum_{i=10}^{n-1} \lfloor i/3 \rfloor v_i. \tag{3}$$

If a vertex u claims halfedges at a vertex v, we say that v *serves* u. According to (C2), every vertex of degree three claims three halfedges at a high-degree neighbor. Every degree four vertex that is adjacent to a hermit together with this hermit claims four halfedges at a high-degree neighbor by (C3). We sum up the number of these claims and assess how many of them can be served by the different types of high-degree vertices.

In general, a high-degree vertex of degree $i \geq 5$ can serve at most $\lfloor i/3 \rfloor$ such claims. For $i \in \{5, 6, 7, 9\}$, we make a more detailed analysis, taking into account the number of adjacent hermits. Specifically, by (C3) and (C4) a degree five vertex serves at most one low-degree vertex, which is either a hermit or a degree three vertex. A degree six vertex can serve two degree three vertices but only if it is not adjacent to a hermit. If a degree six vertex serves a degree four vertex, it is adjacent to exactly one hermit by (C3). In particular, a degree six vertex that is adjacent to two hermits does not serve any degree three or degree four vertex. Altogether we obtain the following inequality:

$$v_3 + v_4^{\mathrm{h1}} \leq v_5^{\mathrm{h0}} + 2v_6^{\mathrm{h0}} + v_6^{\mathrm{h1}} + 2v_7^{\mathrm{h0}} + 2v_7^{\mathrm{h1}} + v_7^{\mathrm{h2}} + 2v_8 + 3v_9^{\mathrm{h0}} + 2v_9^{\mathrm{h1}} + 2v_9^{\mathrm{h2}} + v_9^{\mathrm{h3}} + \sum_{i=10}^{n-1} \lfloor i/3 \rfloor v_i. \tag{4}$$

The combination $((3) + (4))/2$ together with (2) yields

$$v_2 + \frac{1}{2}v_3 \leq \frac{1}{2}v_5 + v_6 + \frac{3}{2}v_7 + 2v_8 + 2v_9 + \sum_{i=10}^{n-1} \lfloor i/3 \rfloor v_i. \tag{5}$$

Now, using these equations and inequalities, we can prove that $m - 2n \geq 0$, to complete the proof of Theorem 1. Let us start from the left hand side, using (1).

$$\begin{aligned} m - 2n &= \frac{1}{2}\sum_{i=2}^{n-1} i v_i - 2\sum_{i=2}^{n-1} v_i = \sum_{i=2}^{n-1} \frac{i-4}{2} v_i \\ &= -v_2 - \frac{1}{2}v_3 + \frac{1}{2}v_5 + v_6 + \frac{3}{2}v_7 + 2v_8 + \frac{5}{2}v_9 + \sum_{i=10}^{n-1} \frac{i-4}{2} v_i \end{aligned}$$

By (5) the right hand side is nonnegative, quod erat demonstrandum.

4 The Upper Bound: Proof outline of Theorem 2

We illustrate a family of maximal 2-planar graphs with $2n + c$ edges in Figure 1. The graphs can roughly be described as braided cylindrical grids where each layer consists of a cycle on ten vertices and every pair of consecutive layers have edges between them. The number of layers in the graph can be increased arbitrarily, and the gadget graph is attached to each of the green edges of the innermost and the outermost cycles. The graph is maximal 2-planar and has $2n + c$ edges, where $c = 350$. The details are deferred to the full version.

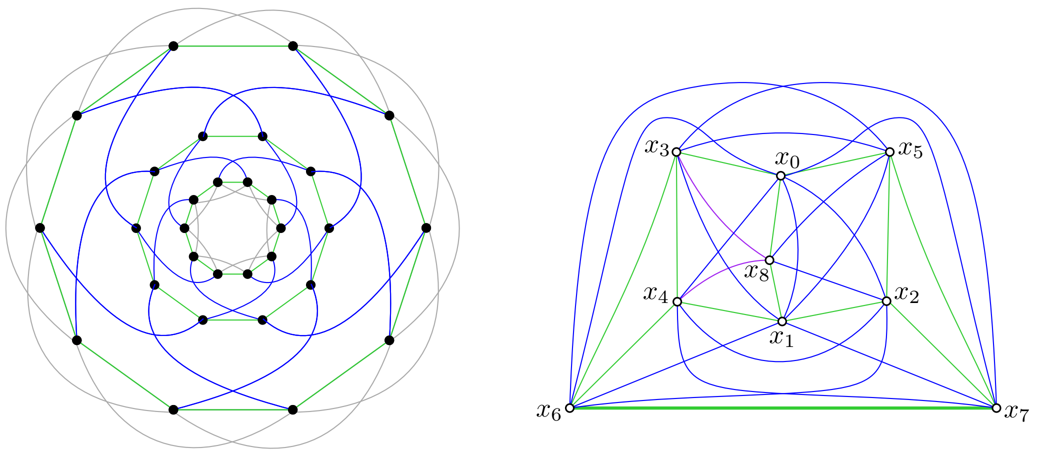


Figure 1 The layered graph (left); the gadget that we attach to the extreme green edges (right).

5 Conclusions

We have obtained tight bounds on the number of edges in maximal 2-planar graphs, up to an additive constant. Naturally, one would expect that our approach can also be applied to other families of near-planar graphs, specifically, to maximal 1- and 3-planar graphs. Intuitively, for k-planar graphs the challenge with increasing k is that the structure of the drawings gets more involved, whereas with decreasing k we aim for a higher bound.

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 Christopher Auer, Franz-Josef Brandenburg, Andreas Gleißner, and Kathrin Hanauer. On sparse maximal 2-planar graphs. In *Proc. 20th Int. Sympos. Graph Drawing (GD 2012)*, volume 7704 of *Lecture Notes Comput. Sci.*, pages 555–556. Springer, 2012. URL: https://doi.org/10.1007/978-3-642-36763-2_50, doi:10.1007/978-3-642-36763-2_50.

3 János Barát and Géza Tóth. Improvements on the density of maximal 1-planar graphs. *J. Graph Theory*, 88(1):101–109, 2018. doi:10.1002/jgt.22187.

4 Michael A. Bekos, Michael Kaufmann, and Chrysanthi N. Raftopoulou. On optimal 2- and 3-planar graphs. In *Proc. 33rd Internat. Sympos. Comput. Geom. (SoCG 2017)*, volume 77 of *LIPIcs*, pages 16:1–16:16. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017. URL: https://doi.org/10.4230/LIPIcs.SoCG.2017.16, doi:10.4230/LIPIcs.SoCG.2017.16.

5 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.

6 Franz-Josef Brandenburg, David Eppstein, Andreas Gleißner, Michael T. Goodrich, Kathrin Hanauer, and Josef Reislhuber. On the density of maximal 1-planar graphs. In *Proc. 20th Int. Sympos. Graph Drawing (GD 2012)*, volume 7704 of *Lecture Notes Comput. Sci.*, pages 327–338. Springer, 2012. URL: https://doi.org/10.1007/978-3-642-36763-2_29, doi:10.1007/978-3-642-36763-2_29.

7 Walter Didimo, Giuseppe Liotta, and Fabrizio Montecchiani. A survey on graph drawing beyond planarity. *ACM Comput. Surv.*, 52(1):1–37, 2020. doi:10.1145/3301281.

8 Seok-Hee Hong and Takeshi Tokuyama, editors. *Beyond Planar Graphs*. Springer, Singapore, 2020. doi:10.1007/978-981-15-6533-5.

9 Dávid Hudák, Tomaš Madaras, and Yusuke Suzuki. On properties of maximal 1-planar graphs. *Discussiones Mathematicae*, 32:737–747, 2012. doi:10.7151/dmgt.1639.

10 János Pach, Radoš Radoičić, Gábor Tardos, and Géza Tóth. Improving the crossing lemma by finding more crossings in sparse graphs. *Discrete Comput. Geom.*, 36(4):527–552, 2006. doi:10.1007/s00454-006-1264-9.

11 János Pach and Géza Tóth. Graphs drawn with few crossings per edge. *Combinatorica*, 17(3):427–439, 1997. URL: https://doi.org/10.1007/BF01215922, doi:10.1007/BF01215922.

12 Marcus Schaefer. The graph crossing number and its variants: A survey. *The Electronic Journal of Combinatorics*, 20, 2013. Version 7 (April 8, 2022). doi:10.37236/2713.

13 Yusuke Suzuki. Optimal 1-planar graphs which triangulate other surfaces. *Discr. Math.*, 310(1):6–11, 2010. doi:10.1016/j.disc.2009.07.016.

Parameterized Complexity of Vertex Splitting to Pathwidth at most 1

Jakob Baumann[1], Matthias Pfretzschner[1], and Ignaz Rutter[1]

1 **University of Passau**
`{baumannjak,pfretzschner,rutter}@fim.uni-passau.de`

Abstract

Motivated by the planarization of 2-layered straight-line drawings, we consider the problem of modifying a graph such that the resulting graph has pathwidth at most 1. The problem Pathwidth-One Vertex Explosion (POVE) asks whether such a graph can be obtained using at most k vertex explosions, where a *vertex explosion* replaces a vertex v by $\deg(v)$ degree-1 vertices, each incident to exactly one edge that was originally incident to v. For POVE, we give an FPT algorithm with running time $O(4^k \cdot m)$ and a quadratic kernel. Similarly, a *vertex split* replaces a vertex v by two distinct vertices v_1 and v_2 and distributes the edges originally incident to v arbitrarily to v_1 and v_2. Analogously to POVE, we define the problem variant Pathwidth-One Vertex Splitting (POVS) that uses the split operation instead of vertex explosions. Here we obtain a linear kernel and a branching algorithm with running time $O((6k + 12)^k \cdot m)$.

Related Version arXiv:2302.14725

1 Introduction

Crossings are one of the main aspects that negatively affect the readability of drawings [14]. It is therefore natural to try and modify a given graph in such a way that it can be drawn without crossings while preserving as much of the information as possible. We consider three different operations.

A *deletion operation* simply removes a vertex from the graph. A *vertex explosion* replaces a vertex v by $\deg(v)$ degree-1 vertices, each incident to exactly one edge that was originally incident to v. Finally, a *vertex split* replaces a vertex v by two distinct vertices v_1 and v_2 and distributes the edges originally incident to v arbitrarily to v_1 and v_2.

Nöllenburg et al. [12] have recently studied the vertex splitting problem, which is known to be NP-complete [7]. In particular, they gave a non-uniform FPT-algorithm for deciding whether a given graph can be planarized with at most k splits. We observe that, since degree-1 vertices can always be inserted into a planar drawing, the vertex explosion model and the vertex deletion model are equivalent for planar graphs. The latter problem, also known as Vertex Planarization, has been studied extensively in the literature [8, 9, 10, 11, 15]. In particular, Jansen et al. [8] gave an FPT-algorithm with running time $O(2^{O(k \log k)} \cdot n)$.

Ahmed et al. [2] investigated the problem of splitting the vertices of a bipartite graph so that it admits a 2-layered drawing without crossings. They assume that the input graph is bipartite and only the vertices of one of the two sets in the bipartition may be split. Under this condition, they give an $O(k^6)$-kernel for the vertex explosion model, which results in an $O(2^{O(k^6)}m)$-time algorithm. They ask whether similar results can be obtained in the vertex splitting model. Figure 1 illustrates the three operations in the context of 2-layered drawings.

We note that a graph admits a 2-layer drawing without crossings if and only if it has pathwidth at most 1, i.e., it is a disjoint union of caterpillars [3, 6], where a caterpillar consists of an induced path with an arbitrary number of adjacent degree-1 vertices. Motivated by this, we more generally consider the problem of turning a graph $G = (V, E)$ into a graph of

39th European Workshop on Computational Geometry, Barcelona, Spain, March 29–31, 2023.
This is an extended abstract of a presentation given at EuroCG'23. 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.

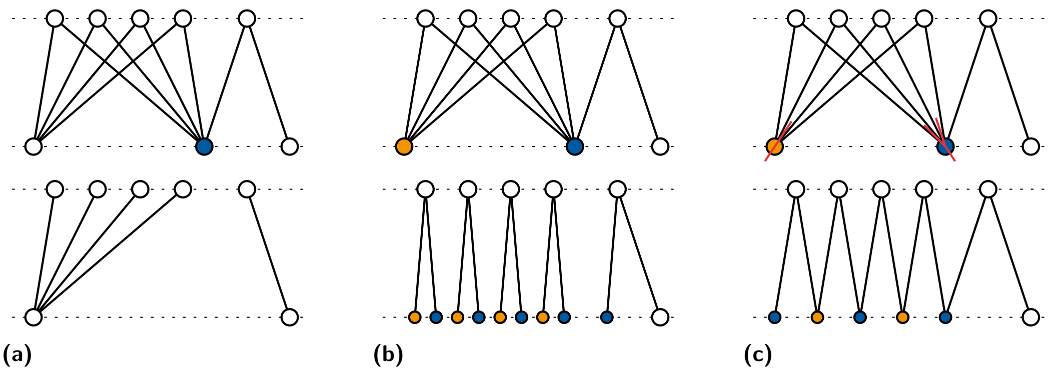


Figure 1 Given the shown bipartite graph, a crossing-free 2-layered drawing can be obtained using one vertex deletion (a), two vertex explosions (b), or three vertex splits (c).

pathwidth at most 1 by the above operations. In order to model the restriction of Ahmed et al. [2] that only one side of their bipartite input graph may be split, we further assume that we are given a subset $S \subseteq V$, to which we may apply modification operations as part of the input.

More formally, we consider the following two problems, both of which have been shown to be NP-hard [1]. Given as input an undirected graph $G = (V, E)$, a set $S \subseteq V$, and a positive integer k, the problem Pathwidth-One Vertex Explosion (POVE) asks whether there is a set $W \subseteq S$ with $|W| \leq k$ such that the graph resulting from exploding all vertices in W has pathwidth at most 1. Analogously, we define the problem Pathwidth-One Vertex Splitting (POVS), which asks for a sequence of at most k vertex splits. For both problems, the new vertices resulting from the respective operation are also included in S.

We note that the analogous problem with the deletion operation has been studied extensively [5, 13, 16]. Here, an FPT algorithm with running time $O(3.888^k \cdot n^{O(1)})$ [16] and a quadratic kernel [5] are known. In this work, we show the following results.

For POVE, we develop a quadratic kernel and an algorithm with running time $O(4^k \cdot m)$, thereby improving over the results of Ahmed et al. [2] in a more general setting. For POVS, we give a linear kernel and an algorithm with running time $O((6k + 12)^k \cdot m)$. This answers the open question of Ahmed et al. [2]. For detailed proofs, we refer to the full version of the paper [4].

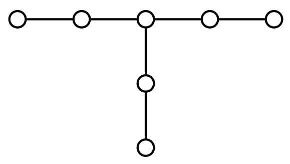
2 Preliminaries

Given a graph G, we let n and m denote the number of vertices and edges of G, respectively. A *path decomposition* of a graph $G = (V, E)$ is a sequence $P = X_1, \ldots, X_l$ of subsets of V, called *bags*, such that

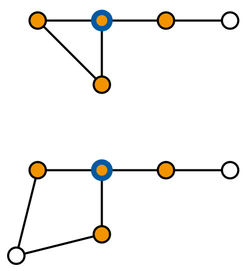
1. $\bigcup\limits_{1 \leq i \leq l} X_i = V$,
2. for every edge $\{u, v\} \in E$, there exists an $i \in \{1, \ldots, l\}$ such that $u, v \in X_i$, and
3. for every vertex $v \in V$, the bags containing v are a contiguous subsequence of P.

The *width* of a path decomposition is one less than the size of its largest bag. The *pathwidth* of G is the minimum width of a path decomposition of G.

We refer to vertices of degree 1 as *pendant* vertices. A graph is a *caterpillar* (respectively a *pseudo-caterpillar*), if it consists of a path (a simple cycle) with an arbitrary number of adjacent pendant vertices. The path (the cycle) is called the *spine* of the (pseudo-)caterpillar.



(a)



(b)

Figure 2 (a) The graph T_2. (b) Two graphs that do not contain T_2 as a subgraph, but both contain N_2 (marked in orange; the root is additionally highlighted in blue) as a substructure.

Philip et al. [13] mainly characterized the graphs of pathwidth at most 1 as the graphs containing no cycles and no T_2 (see Figure 2a) as a subgraph. We additionally use slightly different sets of forbidden substructures. An N_2 *substructure* consists of a *root* vertex r adjacent to three distinct vertices of degree at least 2. Note that every T_2 contains an N_2 substructure, however, the existence of an N_2 substructure does not generally imply the existence of a T_2 subgraph; see Figure 2b. In the following proposition, we state the different characterizations for graphs of pathwidth at most 1 that we use.

▶ **Proposition 2.1.** *For a graph G, the following statements are equivalent.*
1. *G has pathwidth at most 1*
2. *every connected component of G is a caterpillar*
3. *G is acyclic and contains no N_2 substructure*
4. *G contains no N_2 substructure and no connected component that is a pseudo-caterpillar.*

For a vertex $v \in V(G)$, we define $\deg^*(v)$ as the number of non-pendant neighbors of v. If $\deg^*(v) = d$, we say that v has *degree* d*. Additionally, we let $\mu(v) := \max(\deg^*(v) - 2, 0)$ denote the *potential* of v. The *global potential* $\mu(G) := \sum_{v \in V(G)} \mu(v)$ is defined as the sum of the potentials of all nodes in G.

3 FPT Algorithms for Pathwidth-One Vertex Explosion

We start by sketching a simple branching algorithm for POVE, similar to the algorithm by Philip et al. [13] for the deletion variant of the problem.

▶ **Theorem 3.1.** *The problem POVE can be solved in time $O(4^k \cdot m)$.*

Proof. For an N_2 substructure X, observe that exploding vertices not contained in X cannot eliminate X, because the degrees of the vertices in X remain the same due to the new degree-1 vertices resulting from the explosion. To obtain a graph of pathwidth at most 1, it is therefore always necessary to explode one of the four vertices of every N_2 substructure by Proposition 2.1. We can thus define a branching rule that first picks an arbitrary N_2 substructure X from the instance and then branches on which of the four vertices of X should be exploded. Note that we only branch for vertices of X that are also contained in S, i.e., vertices that are allowed to be exploded. Each branch reduces the parameter by 1, the corresponding search tree thus consists of $O(4^k)$ nodes and spends $O(m)$ time at each node to identify the next N_2 substructure and to explode the chosen vertex. We remark that exploding a degree-1 vertex has no effect, it is therefore not necessary to include the new vertices resulting from a vertex explosion in the set S.

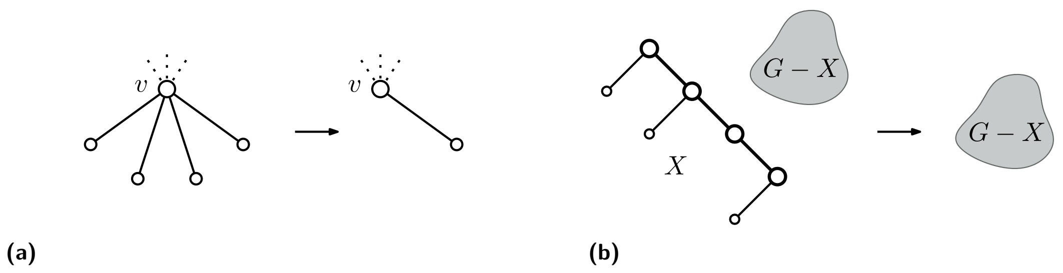


Figure 3 Reductions removing most pendant neighbors (a) and all caterpillars (b).

After exhaustively applying this branching rule, the resulting instance contains no N_2 substructures. By Proposition 2.1, it thus only remains to eliminate connected components that are a pseudo-caterpillar. Since a pseudo-caterpillar can (only) be turned into a caterpillar by exploding a vertex of its spine, the remaining instance (in each leaf of the search tree) can be solved in $O(m)$ time. ◀

We now turn to our kernelization algorithm for POVE. To obtain a quadratic kernel, we first show with the following lemma that we can use reduction rules to obtain an equivalent graph G whose size is bounded linearly in the global potential $\mu(G)$. Subsequently, our goal is to develop an upper bound for $\mu(G)$.

▶ **Lemma 3.2.** *Given an instance of POVE, an equivalent instance with $|V(G)| \leq 8 \cdot \mu(G)$ can be computed in $O(m)$ time.*

Sketch of Proof. We sketch how an equivalent instance of size $|V(G)| \in O(\mu(G))$ can be obtained using reduction rules. To obtain the bound $|V(G)| \leq 8 \cdot \mu(G)$, additional reduction rules and a more refined analysis are necessary, for which we refer to the full version [4].

Our first reduction rule removes all but one pendant neighbors of a vertex v; see Figure 3a. Roughly speaking, the correctness of this reduction follows from the observation that all N_2 substructures and cycles of the instance are preserved. Note that we cannot remove the last pendant neighbor of v, since this could unintentionally remove N_2 substructures from the graph.

The next two reduction rules eliminate all connected components that form a (pseudo-)caterpillar from the instance. Since a caterpillar has pathwidth at most 1, we can safely remove the corresponding component; see Figure 3b. Given a pseudo-caterpillar X, we need to explode one arbitrary vertex of its spine to obtain a caterpillar. If the spine of X contains a vertex of S, we can thus remove X from the instance and decrease the parameter by 1. Otherwise, we reduce to a trivial no-instance; see Figure 4.

Our next reduction rule shortens paths of degree*-2 vertices to constant size. Given such a path of length at least 3, observe that only the first and the last vertex of the path can be contained in an N_2 substructure, because the inner vertices have less than three neighbors of degree at least 2. These inner vertices are thus only contained in a minimum solution if exploding them breaks a cycle. Note that, in this case, exploding any vertex of the path is sufficient to break all cycles the path is contained in. We can thus safely shorten such a path to length 3 by ensuring that the inner vertex of the new path is contained in S if and only if the old path has an inner vertex contained in S; see Figure 5.

After exhaustively applying the reduction rules described above, we can show that $|V(G)| \in O(\mu(G))$. Let V_3 denote the vertices of G of degree* at least 3. Since all vertices of

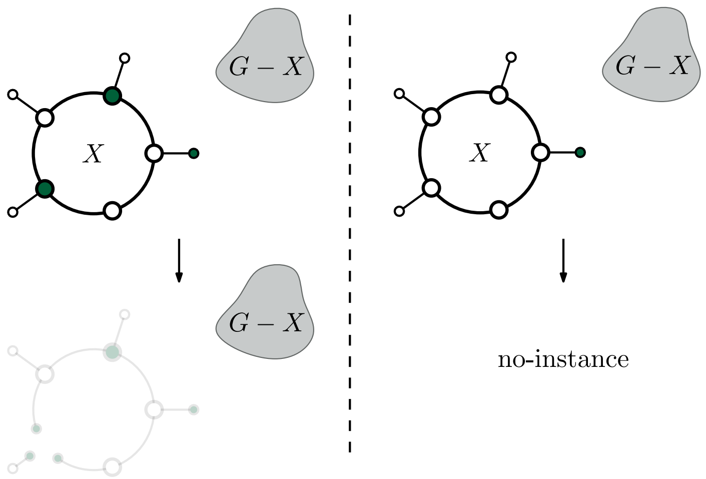


Figure 4 The reduction eliminating pseudo-caterpillars. The vertices of S are marked in green.

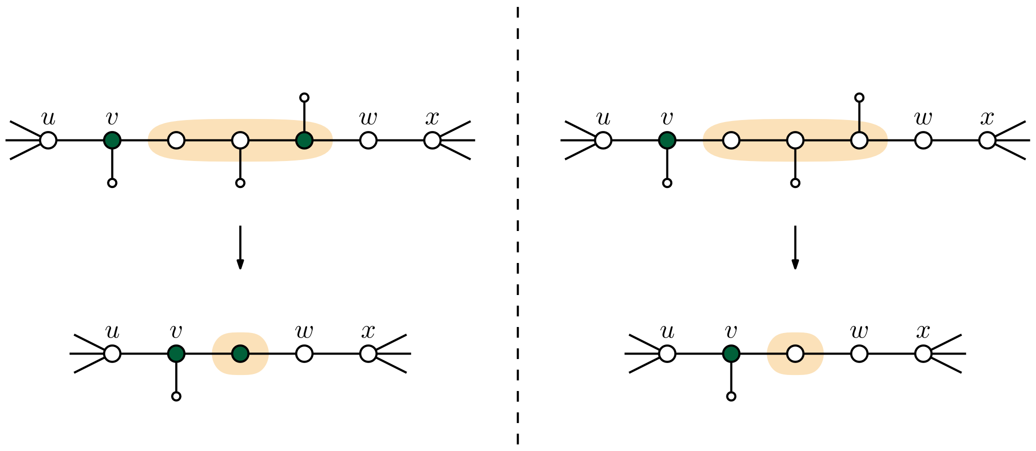


Figure 5 The reduction shortening degree*-2 paths. The inner vertices are highlighted in orange.

degree* at most 2 have potential 0, we can rewrite $\mu(G)$ as

$$\mu(G) = \sum_{v \in V_3} (\deg^*(v) - 2) = \sum_{v \in V_3} \deg^*(v) - 2 \cdot |V_3|.$$

Note that every vertex in V_3 contributes at least 1 to the global potential, thus $|V_3| \leq \mu(G)$. Using this bound in the equation above, we get $\sum_{v \in V_3} \deg^*(v) \leq 3 \cdot \mu(G)$. Note that this bounds exactly the number of non-pendant neighbors of the vertices in V_3. Since we have reduction rules removing all components that form a (pseudo-)caterpillar, every component must contain a vertex of V_3. Because we additionally used a reduction rule to shorten paths of degree*-2 vertices, it is possible to use a handshaking-argument to bound the number of all non-pendant vertices in G in $O(\mu(G))$. Since our first reduction rule ensures that each vertex has at most one pendant neighbor, we consequently have $|V(G)| \in O(\mu(G))$. ◀

To obtain a kernel for POVE, it thus suffices to reduce to an instance with bounded $\mu(G)$. Consider a vertex v of G with $\mu(v) > k$. Since exploding a vertex of $V(G) \setminus \{v\}$ decreases $\mu(v)$ by at most 1, after exploding at most k vertices in $V(G) \setminus \{v\}$ we still have $\mu(v) > 0$. Because $\mu(v) > 0$ implies that G contains an N_2 substructure, it is therefore always necessary to explode vertex v. If $v \in S$, we thus explode v, and we reduce to a trivial no-instance otherwise. Subsequently, every vertex v has $\mu(v) \leq k$, hence exploding v decreases its own potential by at most k, and the potential of each of its at most $k+2$ non-pendant neighbors by at most 1. A sequence of k explosions can thus decrease the global potential by at most $2k^2+2k$. If $\mu(G) > 2k^2+2k$, our final reduction rule therefore reduces to a trivial no-instance. Using Lemma 3.2 with $\mu(G) \leq 2k^2+2k$, we finally obtain the following result.

▶ **Theorem 3.3.** *POVE admits a kernel of size $16(k^2+k)$. It can be computed in time $O(m)$.*

4 FPT Algorithms for Pathwidth-One Vertex Splitting

Linear Kernel. One can prove that the reduction rules for reducing pendant vertices, removing (pseudo-)caterpillars, and shortening degree*-2 paths we used for POVE are also safe for the problem POVS. Since only these are needed to establish the upper bound of $|V(G)| \leq 8 \cdot \mu(G)$ in Lemma 3.2, the lemma also applies for POVS.

The main difference to the kernelization of POVE lies in the way the global potential changes due to splits. While a vertex explosion can decrease the global potential linearly in k, we can show that a single vertex split decreases $\mu(G)$ by at most 2. If $\mu(G) > 2k$, we can thus again reduce to a trivial no-instance. Using Lemma 3.2 with $\mu(G) \leq 2k$, we obtain the following result.

▶ **Theorem 4.1.** *POVS admits a kernel of size $16k$. It can be computed in time $O(m)$.*

Branching Algorithm. As in Section 3, our branching algorithm for POVS eliminates every N_2 substructure of G by branching on which of its four vertices should be split. In this case, however, we need to additionally consider the possible ways to split a single vertex. The following lemma helps us limit the number of suitable splits.

▶ **Lemma 4.2.** *For every instance of POVS, there exists a minimum sequence of splits such that every split operation splits off at most two edges.*

▶ **Theorem 4.3.** *The problem POVS can be solved in time $O((6k+12)^k \cdot m)$.*

Sketch of Proof. From the kernelization, we use the reduction rule that reduces pendant vertices, and the rule that yields the upper bound $\mu(G) \leq 2k$. Together, these two rules ensure that each vertex has degree at most $2k+3$. We now branch on the way of splitting an N_2 substructure with root r and neighbors $\{x, y, z\}$ as above (see Figure 6). If we split r, then, by Lemma 4.2, we may assume that we split off one of the neighbors $\{x, y, z\}$, together with at most one other neighbor of r; these are $3 \cdot (2k+3) = 6k+9$ choices. If we split a vertex $v \in \{x, y, z\}$, then it is necessary that we only split off the edge rv at v to eliminate the N_2 substructure, thus there is only one possibility for each of them, totaling three additional possible choices. Overall, we thus find a branching vector of size $6k+12$. Recall that, by definition, the new vertices resulting from a split operation are included in the set S of splittable vertices. This is important, since it may be necessary to split these vertices again. ◀

References

1 Reyan Ahmed, Patrizio Angelini, Michael A. Bekos, Giuseppe Di Battista, Michael Kaufmann, Philipp Kindermann, Stephen G. Kobourov, Martin Nöllenburg, Antonios Symvonis, Anaïs Villedieu, and Markus Wallinger. Splitting vertices in 2-layer graph drawings. *CoRR*, abs/2301.10872, 2023. `arXiv:2301.10872`, `doi:10.48550/arXiv.2301.10872`.

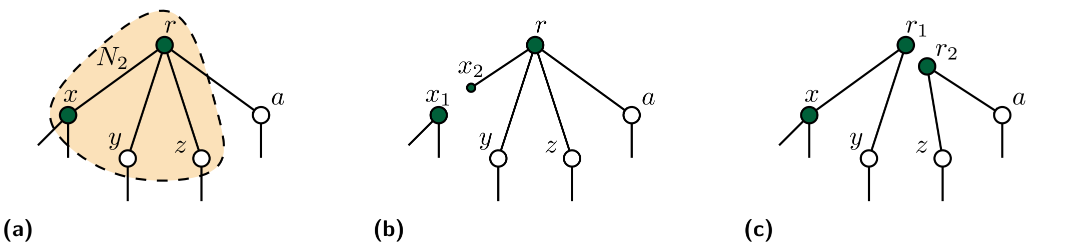


Figure 6 (a) An N_2 substructure $\{r, x, y, z\}$. (b)-(c) Two possible branches eliminating the N_2 substructure. The former splits off edge rx at x, the latter splits off the edges rz and ra at r.

2 Reyan Ahmed, Stephen G. Kobourov, and Myroslav Kryven. An FPT algorithm for bipartite vertex splitting. In Patrizio Angelini and Reinhard von Hanxleden, editors, *Graph Drawing and Network Visualization - 30th International Symposium, GD 2022*, volume 13764 of *Lecture Notes in Computer Science*, pages 261–268. Springer, 2022. doi:10.1007/978-3-031-22203-0_19.

3 Stefan Arnborg, Andrzej Proskurowski, and Detlef Seese. Monadic second order logic, tree automata and forbidden minors. In Egon Börger, Hans Kleine Büning, Michael M. Richter, and Wolfgang Schönfeld, editors, *Computer Science Logic, CSL '90*, volume 533 of *Lecture Notes in Computer Science*, pages 1–16. Springer, 1990. doi:10.1007/3-540-54487-9_49.

4 Jakob Baumann, Matthias Pfretzschner, and Ignaz Rutter. Parameterized complexity of vertex splitting to pathwidth at most 1. *CoRR*, abs/2302.14725, 2023. arXiv:2302.14725, doi:10.48550/ARXIV.2302.14725.

5 Marek Cygan, Marcin Pilipczuk, Michal Pilipczuk, and Jakub Onufry Wojtaszczyk. An improved FPT algorithm and a quadratic kernel for pathwidth one vertex deletion. *Algorithmica*, 64(1):170–188, 2012. doi:10.1007/s00453-011-9578-2.

6 Peter Eades, Brendan D McKay, and Nicholas C Wormald. On an edge crossing problem. In *Proc. 9th Australian Computer Science Conference*, volume 327, page 334, 1986.

7 Luérbio Faria, Celina M. H. de Figueiredo, and Candido Ferreira Xavier de Mendonça Neto. Splitting Number is NP-complete. In Juraj Hromkovic and Ondrej Sýkora, editors, *Graph-Theoretic Concepts in Computer Science, WG '98*, volume 1517 of *Lecture Notes in Computer Science*, pages 285–297. Springer, 1998. doi:10.1007/10692760_23.

8 Bart M. P. Jansen, Daniel Lokshtanov, and Saket Saurabh. A near-optimal planarization algorithm. In Chandra Chekuri, editor, *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1802–1811. SIAM, 2014. doi:10.1137/1.9781611973402.130.

9 Ken-ichi Kawarabayashi. Planarity allowing few error vertices in linear time. In *50th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2009*, pages 639–648. IEEE Computer Society, 2009. doi:10.1109/FOCS.2009.45.

10 John M. Lewis and Mihalis Yannakakis. The node-deletion problem for hereditary properties is NP-complete. *J. Comput. Syst. Sci.*, 20(2):219–230, 1980. doi:10.1016/0022-0000(80)90060-4.

11 Dániel Marx and Ildikó Schlotter. Obtaining a planar graph by vertex deletion. In Andreas Brandstädt, Dieter Kratsch, and Haiko Müller, editors, *Graph-Theoretic Concepts in Computer Science, WG 2007*, volume 4769 of *Lecture Notes in Computer Science*, pages 292–303. Springer, 2007. doi:10.1007/978-3-540-74839-7_28.

12 Martin Nöllenburg, Manuel Sorge, Soeren Terziadis, Anaïs Villedieu, Hsiang-Yun Wu, and Jules Wulms. Planarizing graphs and their drawings by vertex splitting. In Patrizio Angelini and Reinhard von Hanxleden, editors, *Graph Drawing and Network Visualization - 30th*

International Symposium, GD 2022, volume 13764 of *Lecture Notes in Computer Science*, pages 232–246. Springer, 2022. doi:10.1007/978-3-031-22203-0_17.

13 Geevarghese Philip, Venkatesh Raman, and Yngve Villanger. A quartic kernel for pathwidth-one vertex deletion. In Dimitrios M. Thilikos, editor, *Graph Theoretic Concepts in Computer Science, WG 2010*, volume 6410 of *Lecture Notes in Computer Science*, pages 196–207, 2010. doi:10.1007/978-3-642-16926-7_19.

14 Helen C. Purchase, Robert F. Cohen, and Murray I. James. Validating graph drawing aesthetics. In Franz-Josef Brandenburg, editor, *Symposium on Graph Drawing, GD '95*, volume 1027 of *Lecture Notes in Computer Science*, pages 435–446. Springer, 1995. doi:10.1007/BFb0021827.

15 Neil Robertson and Paul D. Seymour. Graph minors .xiii. the disjoint paths problem. *J. Comb. Theory, Ser. B*, 63(1):65–110, 1995. doi:10.1006/jctb.1995.1006.

16 Dekel Tsur. Faster algorithm for pathwidth one vertex deletion. *Theor. Comput. Sci.*, 921:63–74, 2022. doi:10.1016/j.tcs.2022.04.001.

Circling a Square: The Lawn Mowing Problem Is Algebraically Hard

Sándor P. Fekete[1], Dominik Krupke[1], Michael Perk[1], Christian Rieck[1], and Christian Scheffer[2]

1 **Department of Computer Science, TU Braunschweig**
s.fekete@tu-bs.de, d.krupke@tu-bs.de, perk@ibr.cs.tu-bs.de, rieck@ibr.cs.tu-bs.de

2 **Faculty of Electrical Engineering and Computer Science, Bochum University of Applied Sciences, Bochum, Germany**
christian.scheffer@hs-bochum.de

Abstract

For a given polygonal region P, the Lawn Mowing Problem (LMP) asks for a shortest tour T that gets within Euclidean distance 1 of every point in P; this is equivalent to computing a shortest tour for a unit-disk cutter D that covers all of P. We show that the LMP is algebraically hard: it is not solvable by radicals over the field of rationals, even for a simple case in which P is a 4×4 square. This implies that it is impossible to compute exact optimal solutions under models of computation that rely on elementary arithmetic operations and the extraction of kth roots.

1 Introduction

Long before the invention of computers, geometry already faced unsolvable algorithmic problems. This hardness was not rooted in the asymptotic complexity of finding the best of a finite number of candidates, but in the impossibility of obtaining solutions with a given set of construction tools: Computing the length of the diagonal of a square is not possible with only rational numbers; trisecting any given angle cannot be done with ruler and compass, nor can a square be computed whose area is equal to that of a given circle.

In the following, we consider the *Lawn Mowing Problem* (LMP), in which we are given a (not necessarily simple or even connected) polygonal region P and a disk cutter D of radius 1; the task is to find a closed roundtrip (a *tour*) of minimum Euclidean length, such that the cutter “mows” all of P, i.e., a shortest tour that moves the center of D within distance 1 from every point in P. The LMP naturally occurs in a wide spectrum of practical applications, such as robotics, manufacturing, farming, quality control, and image processing, so it is of both theoretical and practical importance. As a generalization of the classic Traveling Salesman Problem (TSP), the LMP is also NP-hard; however, while the TSP has shown to be amenable to exact methods for computing provably optimal solutions even for large instances, the LMP has defied such attempts, with only some moderate recent progress [20].

The main result of this paper is to establish a fundamental reason for this perceived difficulty: Computing an optimal lawn mowing tour is not only NP-hard, but also algebraically hard, even for the seemingly harmless case of mowing a 4×4 square by a unit-radius disk, as it requires computing zeroes of high-order irreducible polynomials. As a consequence, computing even near-optimal solutions for the LMP requires dealing with algebraic issues of numerical approximation and accuracy, making the LMP fundamentally more challenging than its special case, the discrete (Euclidean) TSP.

39th European Workshop on Computational Geometry, Barcelona, Spain, March 29–31, 2023.

Related Work There is a wide range of practical applications for the LMP, including manufacturing [3, 23, 24], cleaning [9], robotic coverage [10, 11, 22, 27], inspection [15], CAD [14], farming [4, 12, 30] and pest control [6]. In Computational Geometry, the Lawn Mowing Problem was first introduced by Arkin et al. [1], who later gave the currently best approximation algorithm with a performance guarantee of $2\sqrt{3}\alpha_{\text{TSP}} \approx 3.46\alpha_{\text{TSP}}$ [2], where α_{TSP} is the performance guarantee for an approximation algorithm for the TSP.

Optimally covering even relatively simple regions by a set of n unit disks has received considerable attention, but is excruciatingly difficult; see [7, 8, 21, 25, 28, 29]. As recently as 2005, Fejes Tóth [16] established optimal values for $n = 8, 9, 10$. Progress on covering by (not necessarily equal) disks has been achieved by Fekete et al. [17, 18].

A seminal result for understanding algebraic aspects of geometric optimization problems was achieved by Bajaj [5], who established algebraic hardness for the Fermat-Weber problem of finding a point in $\mathbb{R}^2$ that minimizes the sum of Euclidean distances to all points in a given set. Note, however, that the Fermat-Weber problem is relatively benign in practical difficulty, as it amounts to minimizing a smooth, convex function over a compact set, which can be achieved with high accuracy by using a numerical approach such as Newton's method. This was exploited for algorithmic purposes by Fekete et al. [19].

The use of straight-edge and compass is known to be equivalent to the use of $(+, -, *, /, \sqrt{})$ over $\mathbb{Q}$ [13]. Our main result implies that the Lawn Mowing Problem is not solvable by radicals over $\mathbb{Q}$, i.e., a solution is not expressible in terms of $(+, -, *, /, \sqrt[k]{})$ over $\mathbb{Q}$.

2 Optimal Tours in Rectangles

Recent work by Fekete et al. [20] shows that when mowing a triangle, optimal tours may need to contain vertices with irrational coordinates. In the following we show even if P is a 4×4 square, an optimal tour may involve coordinates that cannot be described with radicals.

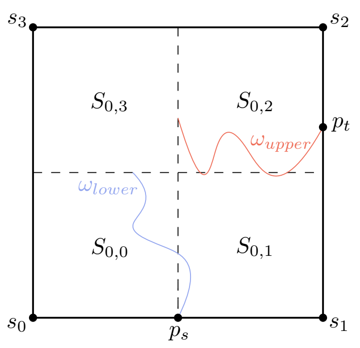
▶ **Theorem 2.1.** *For any rational height $h \geq 4$, there are rectangles P with height h and rational vertex coordinates for which the Lawn Mowing Problem is not solvable by radicals.*

See Figure 3 for the structure of optimal trajectories. A key observation is that covering each of the four corners $(0, -2), (4, -2), (4, 2), (0, 2)$ of a 4×4 square S requires the disk center to leave the subsquare λ with vertices $\lambda_0 = (1, -1)$, $\lambda_1 = (3, -1)$, $\lambda_2 = (3, 1)$, $\lambda_3 = (1, 1)$, obtained by offsetting the boundary of P by the unit radius of D, which is the locus of all disk centers for which λ stays inside P. However, covering the area close to the center of P also requires keeping the center of D within λ; as we argue in the following, this results in a trajectory as shown in Figure 3a, with a "long" portion (shown vertically in the figure) for which the disk covers the center of P and the boundary of D traces the boundary of P, and a "short" portion for which D only dips into λ without tracing the boundary of P.

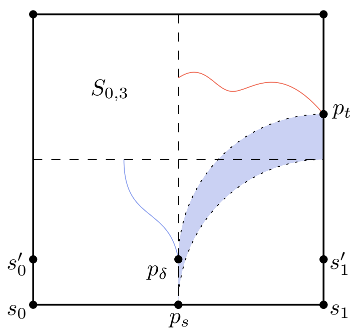
We start our proof by considering an optimal lawn mowing tour for a rectangle and then argue why no solution can be obtained in terms of $(+, -, *, /, \sqrt[k]{})$ over $\mathbb{Q}$.

2.1 Properties of Optimal Tours

For the 4×4 square S, consider the upper left 2×2 subsquare S_0 with corners $(0, 0)$, $(0, 2)$, $(2, 2)$, $(0, 2)$, further subdivided into four 1×1 quadrants $S_{0,0}, \ldots, S_{0,3}$, as shown in Figure 1a, and an optimal path ω that enters S_0 at the bottom and leaves it to the right. Let $p_s = (p_s^x, 0), p_t = (2, p_t^y)$ be the points where ω enters and leaves S_0, respectively. For the following lemmas, we assume that a covering path exists that obeys the above conditions. We will later determine that path and show that it covers S_0.



(a) The upper left 2×2 subsquare S_0 of S.



(b) Visualization of Lemma 2.3.

Figure 1 Computing an optimal path ω through the square S_0.

▶ **Lemma 2.2.** *$p_s^x \leq 1$ and $p_t^y \geq 1$ and either $p_s^x = 1$ or $p_t^y = 1$.*

Proof. To cover s_1, ω must intersect a unit circle centered in s_1. Any path with p_s right of $(1, 0)$ or p_t below $(2, 1)$ can be made shorter by shifting the point p_s to $(1, 0)$ or p_t to $(2, 1)$. Any path with p_s left of $(1, 0)$ and p_t above $(2, 1)$ must enter $S_{0,1}$, resulting in a detour. ◀

Without loss of generality, we assume that $p_s^x = 1$. The next step is to find the optimal position of p_t. As an optimal path ω must enter the quadrant $S_{0,3}$ once, we can subdivide the path into two parts. We denote the part from p_s to $S_{0,3}$ as the *lower portion* and from $S_{0,3}$ to p_t as the *upper portion* of ω. For some $\delta > 0$, let $p_t^y = 1 + \delta$ and $p_\delta = (1, \delta)$.

▶ **Lemma 2.3.** *For any $\delta > 0$, ω has a subpath $p_s p_\delta$.*

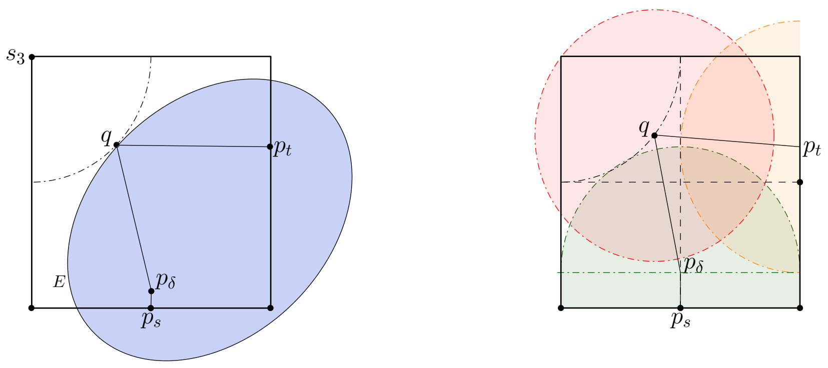
Proof. Let $s'_1 = (2, \delta)$ and $\varepsilon = s_1 s'_1$. Segment ε must be covered by ω. We distinguish two cases; (i) ε is covered by the lower portion of ω or (ii) ε is covered by the upper portion of ω. For case (i), let us assume that ε is covered by the lower portion of ω. When ω would enter $S_{0,1}$ it would also have to enter $S_{0,0}$ to cover the left side of $S_{0,0}$. It is clear that traversing the segment $p_s p_\delta$ of length δ is the best way to cover the lower portion of $S_{0,0}, S_{0,1}$, as any other path would need additional segments in x-direction, see Figure 1b. Any path that obeys case (ii) is suboptimal, as it has to cover ε from within $S_{0,2}$, for a detour of at least 2δ. ◀

We can now define the optimal path ω, which has four vertices. The exact coordinates are defined in the proof of Lemma 2.4.

▶ **Lemma 2.4.** *The unique optimal lawn mowing path between two adjacent sides of S_0 is $\omega = (p_s, p_\delta, q, p_t)$ and has length $L_{S_0} \approx 2.618$.*

Proof. We now identify a shortest path for visiting one point q in the unit circle C centered in s_3 dependent on δ, which is a necessary condition for a feasible path. Let $c = d(p_\delta, q) + d(q, p_t)$ be the distance from both points to C. Consider an ellipse E with foci p_δ, p_t which touches C in a single point, see Figure 2a. By the definition of an ellipse, the intersection point q minimizes the distance c. For any $\delta \in [0, 1]$ we want to find a minimum distance c that allows E to have a single intersection point with C. Let $p_c = (p_c^x, p_c^y)$ be the center point of E and d_E be the distance from the center point of E and a, b the major/minor axis.

$$p_c^x = \frac{3}{2} \quad p_c^y = \frac{1}{2} + \delta \quad d_E = d(p_\delta, p_c) = \frac{1}{\sqrt{2}} \quad a = \frac{1}{2} d_E \quad b = \sqrt{a^2 - d_E^2} \tag{1}$$



(a) Any $0 \leq \delta \leq 1$ defines p_δ, p_t, q and ellipse E. **(b)** The optimal path ω through S_0.

Figure 2 Visualizations for Lemma 2.4.

The ellipse can now be defined with its center point p_c, the major/minor axis a, b and the angle θ, which is the angle between a line through p_δ, p_t and the x-axis. We formulate the shortest path problem as a minimization problem while inserting Equation (1).

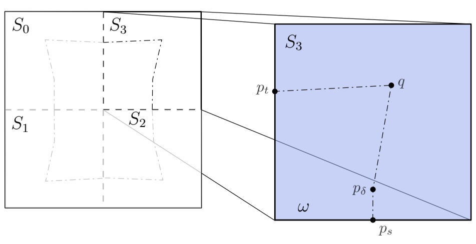
$$\begin{aligned}
\min \quad & c + \delta \\
\text{s.t.} \quad & x^2 + (y-2)^2 - 1 && = 0 \\
& \frac{((x - p_c^x)\cos(\theta) + (y - p_c^y)\sin(\theta))^2}{a^2} + \frac{((x - p_c^x)\sin(\theta) - (y - p_c^y)\cos(\theta))^2}{b^2} && = 1 \\
& \sqrt{(x-1)^2 + (y-\delta)^2} + \sqrt{(x-2)^2 + (y-1-\delta)^2} - c && = 0
\end{aligned}$$

The objective minimizes the total length of the path ω with variables that encode the exact coordinates of p_δ, q, p_t. An intersection point of E and C with center $s_3 = (0, 2)$ is a solution to the first and second constraints, respectively. An exact optimization approach using Mathematica reveals that δ, q^x, q^y can only be expressed as the first, third, and first roots of three irreducible high-degree polynomials $f_\delta, f_{q^x}, f_{q^y}$, see Equations (2) to (4).

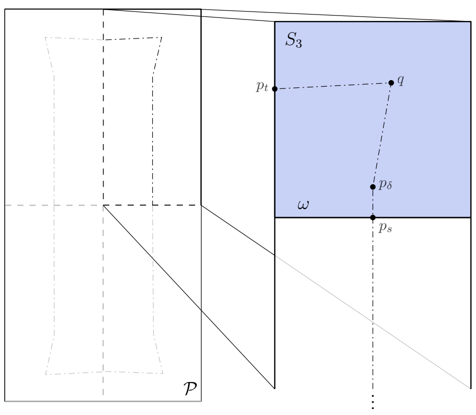
$$\begin{aligned}
f_\delta(x) = {} & 9x^{16} - 216x^{15} + 2514x^{14} - 18\,846x^{13} + 101\,755x^{12} - 418\,512x^{11} + \\
& 1\,350\,994x^{10} - 3\,475\,302x^9 + 7\,165\,772x^8 - 11\,828\,976x^7 + 15\,512\,224x^6 - \\
& 15\,916\,002x^5 + 12\,459\,638x^4 - 7\,145\,094x^3 + 2\,800\,022x^2 - 656\,964x + 67\,417 \qquad (2)\\
f_{q^x}(x) = {} & 256x^{16} - 1792x^{14} + 5312x^{12} - 8768x^{10} + 384x^9 + 8544x^8 - 1632x^7 - \\
& 3648x^6 + 1200x^5 - 152x^4 + 288x^3 + 252x^2 - 324x + 81 \qquad (3)\\
f_{q^y}(x) = {} & 256x^{16} - 8192x^{15} + 122\,624x^{14} - 1\,139\,712x^{13} + 7\,361\,472x^{12} - \\
& 35\,034\,880x^{11} + 127\,069\,376x^{10} - 358\,188\,736x^9 + 792\,777\,952x^8 - \\
& 1\,381\,642\,752x^7 + 1\,888\,549\,824x^6 - 2\,001\,789\,968x^5 + 1\,611\,461\,512x^4 - \\
& 951\,341\,552x^3 + 387\,921\,820x^2 - 97\,469\,232x + 11\,350\,269 \qquad (4)
\end{aligned}$$

The value for $\delta \approx 0.335752$ defines the points p_δ and p_t. Together with the values for q^x, q^y, we can define all points in ω as follows:

$$p_s = (1, 0) \quad p_\delta = (1, \delta) \approx (1, 0.336) \quad q \approx (0.772, 1.365) \quad p_t = (2, 1 + \delta) \approx (2, 1.336) \qquad (5)$$



(a) Optimal lawn mowing tour for a 4×4 square.



(b) Optimal lawn mowing tour for a rectangle.

Figure 3 Optimal lawn mowing tours for a 4×4 square and a $4 \times h$ rectangle.

The combined length of the path is $\delta + c \approx 2.617676448$. As ω contains a subpath that crosses the full height of $S_{0,0}$ and another subpath that crosses the full width of $S_{0,2}$, both quadrants are covered by ω, see Figure 2b. By construction, the bottom right quadrant is covered by the segment $p_s p_\delta$ and the point p_t. The top left quadrant is covered by q, because $S_{0,3}$ is fully contained in a unit disk centered in q. Therefore, ω is a feasible path between two adjacent edges of S_0 with a length of $L \approx 2.618$. ◀

▶ **Lemma 2.5.** *A square P of side length 4 has a unique optimal lawn mowing tour T of length $L = 4L_{S_0}$, where $L_{S_0} \approx 2.618$.*

Proof. We start by subdividing P by its vertical and horizontal center line into four quadrants (squares) $S_0, \dots, S_3$ with side length 2. To cover the center point of each quadrant, a lawn mowing tour has to intersect it at least once. As P is convex, T cannot leave P at any point. Finally, T is symmetric with respect to the vertical and horizontal lines because otherwise, the quadrant subpaths could be replaced by the shortest one. By Lemma 2.4, there is a unique optimal lawn mowing path through each quadrant yielding an optimal tour of length $L = 4L_{S_0} \approx 4 \cdot 2.618 \approx 10.472$, see Figure 3a. ◀

The optimal path from Lemma 2.4 can be used more extensively on rectangles with fixed width 4 and arbitrary height $h \geq 4$. For this, we extend the path from p_s outwards perpendicular to the 2×2 square S_0. One can use a similar construction as in Lemma 2.5 to obtain optimal tours for arbitrary rectangles, refer to Figure 3b.

▶ **Corollary 2.6.** *Any rectangle P with width 4 and height $h \geq 4$ has a unique optimal lawn mowing tour T of length $L = 4L_{S_0} + 2h - 8$.*

2.2 Algebraic Hardness of the LMP

As our next step, we show that the coordinates of the optimal path ω are not solvable by radicals. For this, we employ a similar technique as Bajaj [5] for the generalized Weber problem. A *field* K is said to be an *extension* (written as $K/\mathbb{Q}$) of $\mathbb{Q}$ if K contains $\mathbb{Q}$. Given a polynomial $f(x) \in \mathbb{Q}[x]$, a finite extension K of $\mathbb{Q}$ is a *splitting field* over $\mathbb{Q}$ for $f(x)$ if it can be factorized into linear polynomials $f(x) = (x - a_1) \cdots (x - a_k) \in K[x]$ but not over any proper subfield of K. Alternatively, K is a splitting field of $f(x)$ of degree n over $\mathbb{Q}$

if K is a minimal extension of $\mathbb{Q}$ in which $f(x)$ has n roots. Then the *Galois group* of the polynomial f is defined as the Galois group of $K/\mathbb{Q}$. In principle, the Galois group is a certain permutation group of the roots of the polynomial. From the fundamental theorem of Galois theory, one can derive a condition for solvability by radicals of the roots of $f(x)$ in terms of algebraic properties of its Galois group. We now state three additional theorems from Galois theory and Bajaj's work. The proofs can be found in [26, 5].

▶ **Lemma 2.7** ([26]). *$f(x) \in \mathbb{Q}[x]$ is solvable by radicals over $\mathbb{Q}$ iff the Galois group over $\mathbb{Q}$ of $f(x)$, $Gal(f(x))$, is a solvable group.*

▶ **Lemma 2.8** ([26]). *The symmetric group S_n is not solvable for $n \geq 5$.*

▶ **Lemma 2.9** ([5]). *If $n \equiv 0 \mod 2$ and $n > 2$ then the occurrence of an $(n-1)$-cycle, an n-cycle, and a permutation of type $2 + (n-3)$ on factoring the polynomial $f(x)$ modulo primes that do not divide the discriminant of $f(x)$ establishes that $Gal(f(x))$ over $\mathbb{Q}$ is the symmetric group S_n.*

▶ **Theorem 2.1.** *For any rational height $h \geq 4$, there are rectangles P with height h and rational vertex coordinates for which the Lawn Mowing Problem is not solvable by radicals.*

Proof. It suffices to show that f_δ is not solvable by the radicals as it describes the y-coordinates of two points in the solution. We provide three factorizations of f_δ modulo three primes that do not divide the discriminant $disc(f_\delta(x))$.

$$\begin{aligned}
f_\delta(x) \equiv\ & 9(x^{16} + 22x^{15} + 11x^{14} + 22x^{13} + 8x^{12} + 20x^{11} + 15x^{10} + 10x^9 + 11x^8 + 12x^7 + \\
& 9x^6 + 10x^5 + x^4 + 7x^3 + 13x^2 + 6x + 3) \mod 23 \\
f_\delta(x) \equiv\ & 9(x+41)(x^2 + 21x + 15)(x^{13} + 8x^{12} + 4x^{11} + 46x^{10} + 4x^9 + 14x^8 + 32x^7 + 14x^5 + \\
& 31x^4 + 41x^3 + 37x^2 + 32x + 41) \mod 47 \\
f_\delta(x) \equiv\ & 9(x+19)(x^{15} + 16x^{14} + 54x^{13} + 7x^{12} + 9x^{11} + 36x^{10} + 45x^9 + x^8 + 45x^7 + 3x^6 + \\
& 22x^5 + 36x^4 + 26x^3 + 22x^2 + 54x + 23) \mod 59
\end{aligned}$$

For the good primes $p = 23, 47$, and 59 the degrees of the irreducible factors of $f_\delta(x)$ mod p gives us an $16 - cycle$, a $2 + 13$ permutation and a 15-cycle, which is enough to show with Lemma 2.9 and $n = 16$ that $Gal(f_\delta) = S_{16}$. By Lemma 2.8, S_{16} is not solvable; with Lemma 2.7, this proves the theorem. ◀

3 Conclusion

We have shown that the Lawn Mowing Problem is algebraically hard, even when mowing a 4×4 square P by a unit-radius cutter D. This implies that computing provably optimal tours (such as for the TSP) would involve complicated coordinates; even good approximations (such as a PTAS) require good numerical approximations of the involved algebraic terms.

While our proof makes intricate use of the underlying structure of optimal tours, it is conceivable that similar techniques may help to better understand the difficulty of other excruciatingly hard optimization problems, such as disk packing or covering.

References

1 Esther M. Arkin, Sándor P. Fekete, and Joseph S. B. Mitchell. The lawnmower problem. In *Canadian Conference on Computational Geometry (CCCG)*, pages 461–466, 1993.

2 Esther M. Arkin, Sándor P. Fekete, and Joseph S. B. Mitchell. Approximation algorithms for lawn mowing and milling. *Computational Geometry*, 17:25–50, 2000. `doi:10.1016/S0925-7721(00)00015-8`.

3 Esther M. Arkin, Martin Held, and Christopher L. Smith. Optimization problems related to zigzag pocket machining. *Algorithmica*, 26(2):197–236, 2000. `doi:10.1007/s004539910010`.

4 Rik Bähnemann, Nicholas Lawrance, Jen Jen Chung, Michael Pantic, Roland Siegwart, and Juan Nieto. Revisiting Boustrophedon coverage path planning as a generalized traveling salesman problem. In *Field and Service Robotics*, pages 277–290, 2021. `doi:10.1007/978-981-15-9460-1_20`.

5 Chanderjit Bajaj. The algebraic degree of geometric optimization problems. *Discrete & Computational Geometry*, 3(2):177–191, 1988.

6 Aaron T. Becker, Mustapha Debboun, Sándor P. Fekete, Dominik Krupke, and An Nguyen. Zapping Zika with a mosquito-managing drone: Computing optimal flight patterns with minimum turn cost. In *Symposium on Computational Geometry (SoCG)*, pages 62:1–62:5, 2017. `doi:10.4230/LIPIcs.SoCG.2017.62`.

7 Károly Bezdek. *Körök optimális fedései (Optimal Covering of Circles)*. PhD thesis, Eötvös Lorand University, 1979.

8 Károly Bezdek. Über einige optimale Konfigurationen von Kreisen. *Ann. Univ. Sci. Budapest Rolando Eötvös Sect. Math*, 27:143–151, 1984.

9 Richard Bormann, Joshua Hampp, and Martin Hägele. New brooms sweep clean–An autonomous robotic cleaning assistant for professional office cleaning. In *International Conference on Robotics and Automation (ICRA)*, pages 4470–4477, 2015. `doi:10.1109/ICRA.2015.7139818`.

10 Tauã M. Cabreira, Lisane B. Brisolara, and Paulo R. Ferreira Jr. Survey on coverage path planning with unmanned aerial vehicles. *Drones*, 3(1):4, 2019. `doi:10.3390/drones3010004`.

11 Howie Choset. Coverage for robotics–a survey of recent results. *Annals of Mathematics and Artificial Intelligence*, 31(1):113–126, 2001. `doi:10.1023/A:1016639210559`.

12 Howie Choset and Philippe Pignon. Coverage path planning: The Boustrophedon cellular decomposition. In *Field and Service Robotics*, pages 203–209, 1998. `doi:10.1007/978-1-4471-1273-0_32`.

13 Richard Courant. *What is mathematics?* Oxford University Press, 1941.

14 Gershon Elber and Myung-Soo Kim. Offsets, sweeps and Minkowski sums. *Computer-Aided Design*, 31(3), 1999. `doi:10.1016/S0010-4485(99)00012-3`.

15 Brendan Englot and Franz Hover. Sampling-based coverage path planning for inspection of complex structures. In *International Conference on Automated Planning and Scheduling (ICAPS)*, volume 22, pages 29–37. AAAI, 2012. `doi:10.1609/icaps.v22i1.13529`.

16 Gábor Fejes Tóth. Thinnest covering of a circle by eight, nine, or ten congruent circles. *Combinatorial and Computational Geometry*, 52:361–376, 2005.

17 Sándor P. Fekete, Utkarsh Gupta, Phillip Keldenich, Christian Scheffer, and Sahil Shah. Worst-case optimal covering of rectangles by disks. In *Symposium on Computational Geometry (SoCG)*, pages 42:1–42:23, 2020. `doi:10.4230/LIPIcs.SoCG.2020.42`.

18 Sándor P. Fekete, Phillip Keldenich, and Christian Scheffer. Covering rectangles by disks: The video. In *Symposium on Computational Geometry (SoCG)*, pages 71:1–75:5, 2020. `doi:10.4230/LIPIcs.SoCG.2020.75`.

19 Sándor P. Fekete, Henk Meijer, André Rohe, and Walter Tietze. Solving a “hard” problem to approximate an “easy” one: Heuristics for maximum matchings and maximum Traveling Salesman problems. *ACM Journal of Experimental Algorithms*, 7, 2002. 21 pages. `doi:10.1145/944618.944629`.

20 Sándor P. Fekete, Dominik Krupke, Michael Perk, Christian Rieck, and Christian Scheffer. A closer cut: Computing near-optimal lawn mowing tours. In *2023 Proceedings of the Symposium on Algorithm Engineering and Experiments (ALENEX)*, pages 1–14, 2023. `doi:10.1137/1.9781611977561.ch1`.

21 Erich Friedman. Circles covering squares web page. `https://erich-friedman.github.io/packing/circovsqu`, 2014. Online, accessed January 10, 2023.

22 Enric Galceran and Marc Carreras. A survey on coverage path planning for robotics. *Robotics and Autonomous Systems*, 61(12):1258–1276, 2013. `doi:10.1016/j.robot.2013.09.004`.

23 Martin Held. *On the Computational Geometry of Pocket Machining*, volume 500 of *LNCS*. Springer, 1991. `doi:10.1007/3-540-54103-9`.

24 Martin Held, Gábor Lukács, and László Andor. Pocket machining based on contour-parallel tool paths generated by means of proximity maps. *Computer-Aided Design*, 26(3):189–203, 1994. `doi:10.1016/0010-4485(94)90042-6`.

25 Aladár Heppes and Hans Melissen. Covering a rectangle with equal circles. *Periodica Mathematica Hungarica*, 34(1-2):65–81, 1997. `doi:10.1023/A:1004224507766`.

26 Israel N. Herstein. *Topics in algebra.* John Wiley & Sons, 1991.

27 Katharin R. Jensen-Nau, Tucker Hermans, and Kam K. Leang. Near-optimal area-coverage path planning of energy-constrained aerial robots with application in autonomous environmental monitoring. *IEEE Transactions on Automation Science and Engineering*, 18(3):1453–1468, 2021. `doi:10.1109/TASE.2020.3016276`.

28 Johannes B. M. Melissen and Peter C. Schuur. Covering a rectangle with six and seven circles. *Discrete Applied Mathematics*, 99(1-3):149–156, 2000. `doi:10.1016/S0166-218X(99)00130-4`.

29 Eric H. Neville. On the solution of numerical functional equations. *Proceedings of the London Mathematical Society*, 2(1):308–326, 1915. `doi:10.1112/plms/s2_14.1.308`.

30 Timo Oksanen and Arto Visala. Coverage path planning algorithms for agricultural field machines. *Journal of Field Robotics*, 26(8):651–668, 2009. `doi:10.1002/rob.20300`.

Simply Realising an Imprecise Polyline is NP-hard

Thijs van der Horst[1,2], Tim Ophelders[1,2], and Bart van der Steenhoven[2]

1 **Department of Information and Computing Sciences, Utrecht University, the Netherlands**
`{t.w.j.vanderhorst|t.a.e.ophelders}@uu.nl`

2 **Department of Mathematics and Computer Science, TU Eindhoven, the Netherlands**
`b.j.v.d.steenhoven@student.tue.nl`

Abstract

We consider the problem of deciding, given a sequence of regions, if there is a choice of points, one for each region, such that the induced polyline is simple or weakly simple, meaning that it can touch but not cross itself. Specifically, we consider the case where each region is a translate of the same shape. We show that the problem is NP-hard when the shape is a unit-disk or unit-square. We argue that the problem is is NP-complete when the shape is a vertical unit-segment.

1 Introduction

Finding a planar drawing of a graph has been studied widely due to its many applications, for example in floor-planning and visualisations [7]. Often when the graphs come from geometric data, such as road networks, vertices already have predetermined locations. In these cases, if edges must be drawn as straight line segments, there is a unique corresponding drawing. Real-world data such as GPS trajectories are often imprecise. Imprecision can be modeled by assigning to each vertex an imprecision region. A geometric graph induced by assigning each vertex to a point in its imprecision region is called a *realisation*.

Algorithms for imprecise data have been considered in the past [5, 8]. We consider the problem of deciding whether imprecise data admits a realisation without self-intersections. Godau [4] showed that this problem is NP-hard if the imprecision regions are closed disks of varying radius. In follow-up work, Angelini et al. [2] showed that the problem remains NP-hard when the regions are pairwise-disjoint unit-disks, or unit-squares. In this work, we consider the problem for a very restricted graph class, namely path-graphs. That is, we are looking for a realisation that is a simple polyline.

Efficient algorithms exist to determine whether a polygon or polyline is weakly simple [1, 3]. When the polygon is imprecise however, Löffler [6] showed that determining whether a weakly simple realisation exists when the imprecision regions are scaled translates of a fixed shape, such as a segment or circle, is NP-hard. Finally, Silveira et al. [9] show that the problem is NP-complete for straight-line drawings of graphs where each vertex has degree exactly one (i.e. the graph is a matching), using unit-length vertical segments as imprecision regions.

Definitions and problem statement. A *polygonal chain* or *polyline* is a piecewise-linear path in the plane, given by a sequence $P = (p_i)_{i=0}^n$ of points called vertices. With slight abuse of notation, we interpret P as the polyline, instead of its vertices. An *imprecise polyline* is given by a sequence of regions $R = (r_i)_{i=0}^n$ in the plane. We call the polyline $(p_i)_{i=0}^n$ a *realisation* of R if $p_i \in r_i$ for each i.

39th European Workshop on Computational Geometry, Barcelona, Spain, March 29–31, 2023.
This is an extended abstract of a presentation given at EuroCG'23. 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 polyline is *simple* (or plane) if it does not cross or touch itself. An *ε-perturbation* of a polyline $P = (p_i)_{i=0}^n$ is a polyline $P' = (p'_i)_{i=0}^n$ such that $\|p_i - p'_i\| \leq \varepsilon$ for all i. A polyline $(p_i)_{i=0}^n$ is *weakly simple* if for every $\varepsilon > 0$ there is an ε-perturbation that is simple. We are interested in the problem of deciding whether an imprecise polyline R admits a weakly simple realisation.

Results and organisation. We show that the problem is NP-hard already when each region is a translate of the same shape S. This setting differs from [6] in the sense that now all regions must have uniform size. In Section 2 we prove this for the case where S is a unit disk, and discuss how the construction can be adapted to show NP-hardness for unit squares. Hence, computing the infimum ε for which a polyline has a simple ε-perturbation under the L_1, L_2, and L_∞ norms is NP-hard. In the full version we additionally show NP-completeness if S is a unit-length vertical segment.

2 Unit disks as regions

We prove by reduction from planar monotone 3SAT that the problem is NP-hard when all regions are unit disks. We first introduce planar monotone 3SAT. A monotone 3CNF formula is a formula $\varphi(x_1, \ldots, x_m) := \bigwedge_i C_i$ with Boolean variables $x_1, \ldots, x_m$, where each clause C_i is either positive (of the form $(x_j \vee x_k \vee x_l)$) or negative (of the form $(\neg x_j \vee \neg x_k \vee \neg x_l)$). A monotone rectilinear layout of a monotone 3CNF formula is a plane drawing of the incidence graph between its clauses and variables, in which variables are drawn as disjoint horizontal segments on the x-axis, all positive clauses lie above the x-axis, all negative clauses lie below the x-axis, and edges are rectilinear, do not cross the x-axis, and have at most one bend. We may assume for each clause that its central edge does not bend, see Figure 1. Planar monotone 3SAT is an NP-complete problem that asks, given a monotone 3CNF formula φ with a given monotone rectilinear layout, whether φ is satisfiable. For our reduction we construct gadgets consisting of imprecise polylines corresponding to variables, clauses, and edges of the layout. We then show how to connect these gadgets into a single imprecise polyline that has a weakly simple realisation if and only if the formula φ if satisfiable.

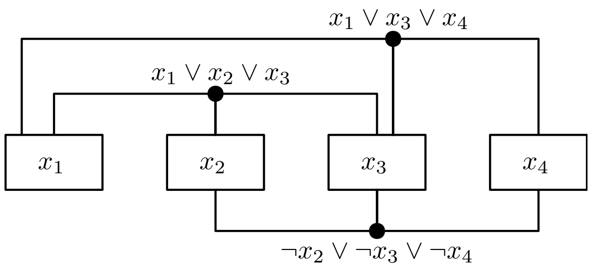


Figure 1 Example of an induced variable-clause graph for planar monotone 3SAT.

Pivot gadget. Before we construct the gadgets for the variables and clauses, we introduce an auxiliary gadget: the *pivot gadget*. The purpose of the pivot gadget is to force a given edge of any weakly simple realisation to go through some fixed point of our choosing. The gadget is illustrated in Figure 2 and consists of three components. Two of these components together form the pivot so that the edge of the third component must pass through the

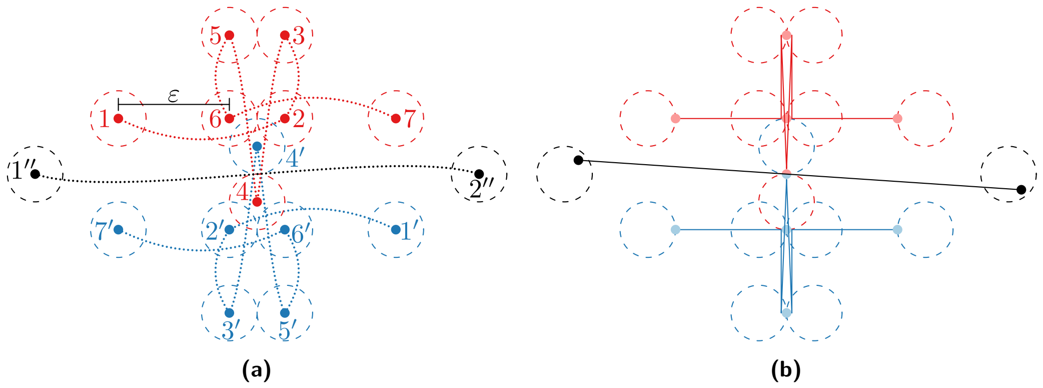


Figure 2 (a) The pivot gadget. (b) A weakly simple realisation, showing a simple perturbation. Note that this perturbation moves most vertices outside of their assigned disks

center $(0,0)$ of the pivot. The coordinates of the top component will be:

$$1,\ldots,7 \mapsto (-1-\varepsilon,2),(1,2),(1,5),(0,-1),(-1,5),(-1,2),(1+\varepsilon,2)$$

for some value $\varepsilon > 0$, and the bottom component is a rotated copy of the top component:

$$1',\ldots,7' \mapsto (1+\varepsilon,-2),(-1,-2),(-1,-5),(0,1),(1,-5),(1,-2),(-1-\varepsilon,-2).$$

There is some freedom in choosing the placement of regions $1''$ and $2''$, but to make sure that the realisation passes through the pivot we require that the x-coordinate of region $1''$ is less than -1, that of region $2''$ is greater than 1, and the tangent lines to the pair of circles $1''$ and $2''$ all intersect the segment between $(0,-5)$ and $(0,5)$. We can accommodate any angle of the edge between regions $1''$ and $2''$ by choosing $\varepsilon > 0$ sufficiently small. Namely, such that the point $(-\varepsilon,2)$ where we will realise region 1 lies above the tangent lines from point $(0,0)$ to region $1''$, and such that symmetric properties hold for regions 7, $1'$ and $7'$.

Figure 2(b) depicts a weakly simple realisation of the pivot gadget. Other weakly simple realisations may differ in placement for points in regions 1, 7, $1'$, $7'$, $1''$ and $2''$ only; the segment connecting $1''$ and $2''$ must always pass through $(0,0)$.

▶ **Lemma 1.** *For any weakly simple realisation, the segment of the pivot gadget connecting regions $1''$ and $2''$ must pass through $(0,0)$.*

▶ **Lemma 2.** *Any choice of points for regions $1''$ and $2''$ such that the segment connecting them passes through $(0,0)$ has a corresponding weakly simple realisation of the pivot gadget.*

Variable gadget. We construct a variable gadget that can take on two distinct states, one for true and one for false, see Figure 3(a). The exact coordinates of the regions are as follows,

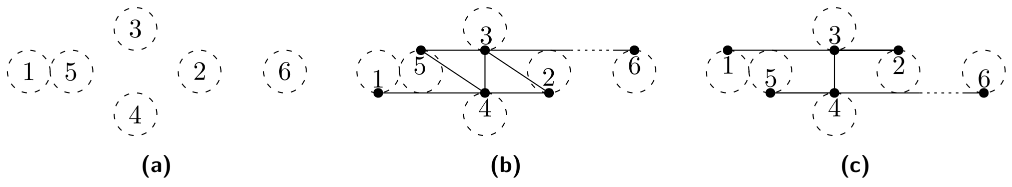


Figure 3 (a) The variable gadget. (b) The false state. (c) The true state.

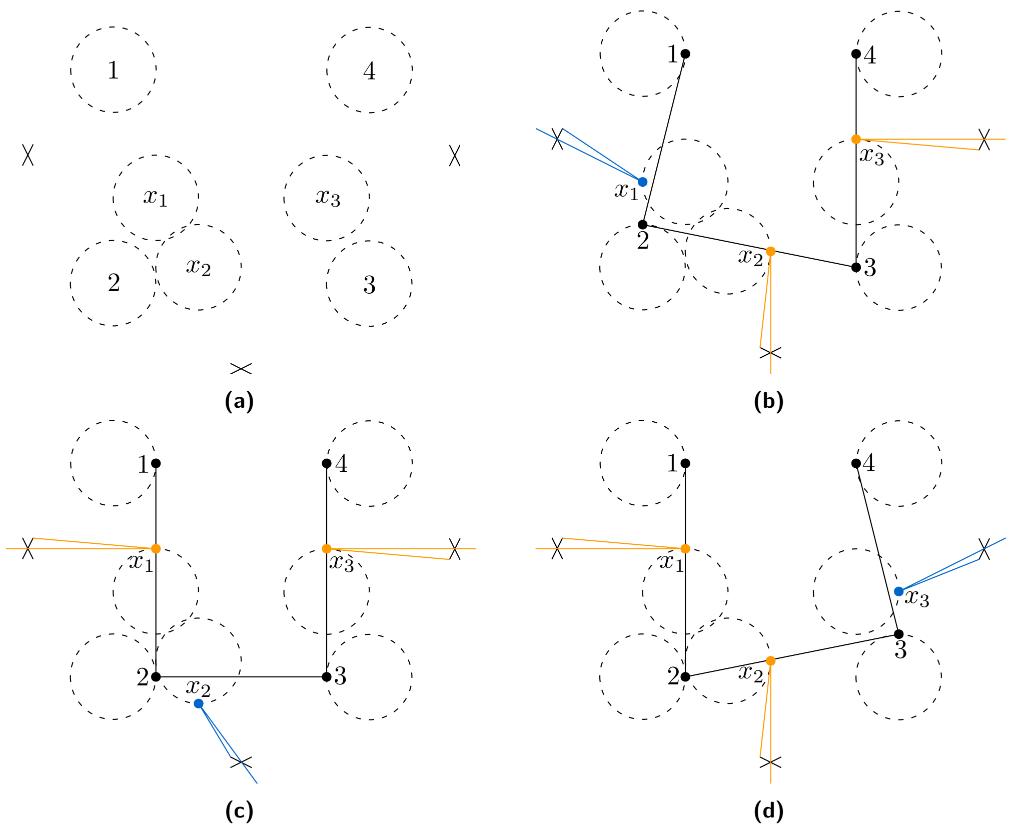


Figure 4 (a) The clause gadget, with realisations where (b) x_1, (c) x_2, or (d) x_3 is true.

assuming the left most region is centered at $(0, 0)$ and $l > 8$ is arbitrary:

$$1, \ldots, 6 \mapsto (0, 0), (8, 0), (5, 2), (5, -2), (2, 0), (l, 0).$$

▶ **Lemma 3.** *The variable gadget admits exactly two weakly simple realisations.*

If the final horizontal line from region 5 to 6 is above the blocking vertical line between regions 3 and 4, then the gadget is in its false state (Figure 3(b)). We consider the situation where it is below as the true state (Figure 3(c)).

Clause gadget. The clause gadget is used to represent formulas of the form $(x_1 \vee x_2 \vee x_3)$, where the variables can also all be negated. The initial placement of the regions for this gadget can be seen in Figure 4(a). We use an X-shape to schematically represent the pivot gadgets. The regions x_1, x_2 and x_3 correspond to the literals of the clause and the regions 1, 2, 3 and 4 are used to put restrictions on these literals. The exact positions of the regions of this gadget are as follows, centering the top-left region at $(0, 0)$:

$$1, 2, 3, 4 \mapsto (0, 0), (0, -5), (6, -5), (6, 0).$$

We place the point for x_1 at $(5, -3)$, x_2 at $(2, -4.6)$ and x_3 at $(1, -3)$. The construction used to connect the regions of these literals to variable gadgets and the exact placement of the pivot gadgets is explained later. For now, it is only important to know that when a variable is false, then the part of the polyline going to the literal point must either go completely vertical through the pivot gadget at the bottom, or completely horizontal through the pivot gadgets at the sides. The regions of x_1, x_2, and x_3 are placed precisely so that in this situation there is exactly one realisation that does not intersect the pivot gadget. We shall refer to the position of a variable in this realisation as its *false position* and say that it is *covered* in a solution if it lies in the interior of the polyline defined by the corner points of the clause gadget. If the variable is true then the line can be placed at an angle through the pivot gadget, allowing the literal point to be realised in multiple ways. We refer to the left-, bottom- and rightmost positions of x_1, x_2, and x_3 as their respective *true position.*

▶ **Lemma 4.** *For every choice of two of the three literals there exists a realisation that uncovers their false positions, and any realisation uncovers at most two false positions.*

Connecting variables and clauses. For the reduction we must connect the variable gadgets to the clause gadgets. For this we follow the layout given by the planar monotone 3SAT instance (see Figure 1). Clauses always connect to variables in a downward or upward facing E-shape. To preserve the planarity of this graph, our reduction should follow this shape to a close enough degree. So the clause gadget is to be placed directly above one of the variable gadgets and the other two variable gadgets connect to it with a bend from the side.

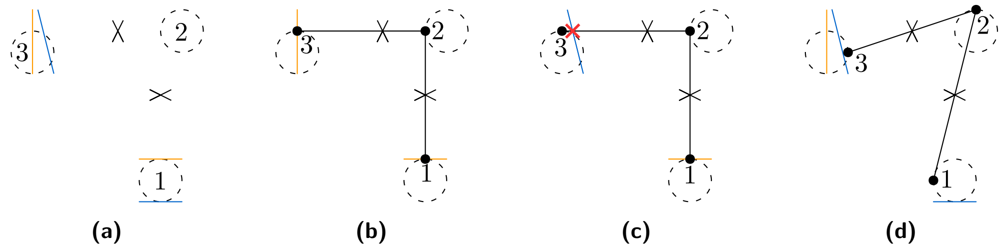


Figure 5 (a) The right wire gadget to connect a variable to a clause. Realisations where (b) both are in the false state, (c) the variable is in its false state and the clause in its true state and (d) the variable gadget in its true state.

Consider the right connection from a variable to a clause. The wire gadget in this case involves three regions and two pivot gadgets (see Figure 5(a)). Region 1 is horizontally aligned with the variable gadget of the corresponding to the right literal of the clause. As in Figure 3, we place these variable regions between region 2 and 6, resulting in a realisation with a horizontal line at the top if the state is false and at the bottom if the state is true. We denote the position of region 1 as $(0, 0)$. Region 3 is the right literal region of the clause gadget. The position of this region is determined by the clause gadget. Let region 3 be positioned at $(-a, b)$. For our wire gadget, we place region 2 at position $(1, b + 1)$ and place two pivot gadgets at $(\frac{-a}{2} + 1, b + 1)$ and $(0, \frac{b}{2} + 1)$, oriented appropriately. We stretch the variable-clause graph without changing its validity, so that a and b are large enough that the components of the wire gadget do not overlap.

▶ **Lemma 5.** *If a variable gadget for x_i is in its false state, any weakly simple realisation places x_i in its false position. Otherwise x_i can be placed in either its true or false position.*

The wire gadget for connecting the left literal of a clause to its variable gadget is the above gadget but mirrored. For connecting the middle variable to the clause we do the same, except here we need only a single pivot gadget and no region 2, as this connection does not need to bend. All three versions of the wire gadget can be seen in effect in Figure 6.

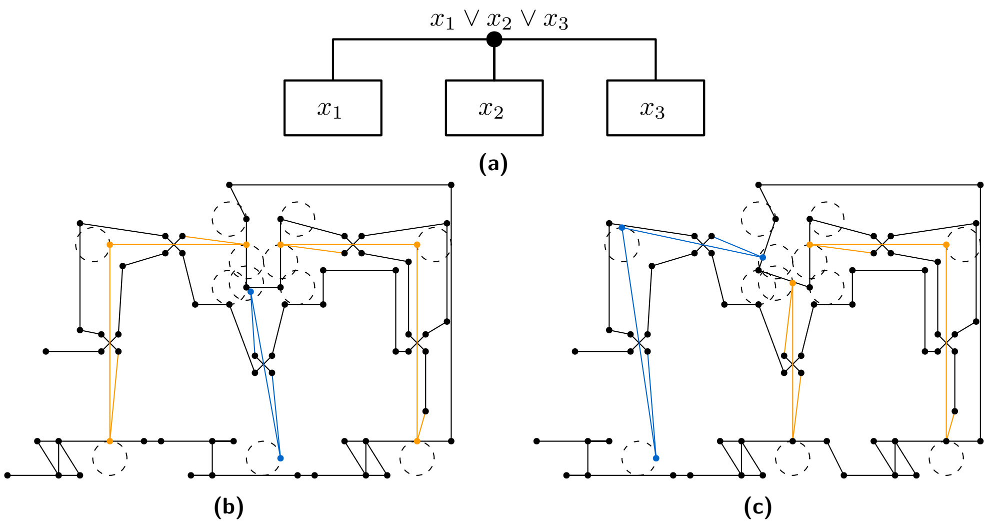


Figure 6 Example of the construction for variable-clause graph (a). The weakly simple realisation in (b) sets x_2 to true and the realisation in (c) sets x_1 to true.

The complete reduction. The general construction works as follows. First we place all gadgets as prescribed by the layout obtained from the planar monotone 3SAT instance. Next, we connect the gadgets in a planar manner. For this, we connect the variables from left to right as in Figure 6. We connect the gadgets in the top half of the construction by processing the clauses on the outer face (that lie above the x-axis) as follows, and connecting the results from left to right. Connect the clause to its right wire (and its pivot gadgets), then recursively process and connect the clauses in the region enclosed by its right and middle wire, then connect the middle wire, then recursively process and connect the clauses in the region enclosed by the middle and left wire, and finally connect the left wire. The gadgets in the bottom half are connected in a symmetric manner. We can put everything together by connecting the variables, top half, and bottom half (their loose endpoints all lie on the "outer face").

Correctness of the reduction follows from the correctness of the individual gadgets. As each gadget and connection uses a constant number of regions, whose positions can be determined based on the variable-clause graph, the reduction is polynomial in size and time.

▶ **Theorem 6.** *Deciding whether there exists a weakly simple realisation for a sequence of unit disks is NP-hard.*

▶ **Observation 7.** If our construction admits a weakly simple realisation, then there is one that uses only the top, right, bottom and leftmost points of its regions. The problem hence remains NP-hard when each region is a subset of a unit disk and contains these four points.

The problem we considered can also be looked at from a different point of view. One could consider the center of each region in the sequence to be a vertex of an input polyline and the regions themselves denoting how we are allowed to perturb these vertices. In that case the setting of unit disks that we discussed corresponds to perturbing vertices by unit distance under the L_2 norm. Under the L_1 norm the regions are diamond-shaped and contained in a unit disk. Thus by the above observation the problem remains NP-hard. After an affine transformation, it follows that the problem is also NP-hard under the L_∞ norm.

3 Discussion

We showed that deciding if an imprecise polygon has a weakly simple realisation is NP-hard if each region is a unit disk, unit square, or vertical unit segment. By growing each region slightly, the same follows for deciding whether a simple realisation exists. Whereas the case of vertical segments is NP-complete, it is unclear whether the problems of Section 2 lie in NP, as it is unclear whether polynomially many bits suffice to encode a solution. The settings of Section 2 lie in the complexity class $\exists\mathbb{R}$, and we wonder if these settings are $\exists\mathbb{R}$-hard.

References

1 Hugo A. Akitaya, Greg Aloupis, Jeff Erickson, and Csaba D. Tóth. Recognizing weakly simple polygons. *Discret. Comput. Geom.*, 58(4):785–821, 2017.

2 Patrizio Angelini, Giordano Da Lozzo, Marco Di Bartolomeo, Giuseppe Di Battista, Seok-Hee Hong, Maurizio Patrignani, and Vincenzo Roselli. Anchored drawings of planar graphs. In *GD*, volume 8871 of *Lecture Notes in Computer Science*, pages 404–415. Springer, 2014.

3 Hsien-Chih Chang, Jeff Erickson, and Chao Xu. Detecting weakly simple polygons. In *SODA*, pages 1655–1670. SIAM, 2015.

4 Michael Godau. On the difficulty of embedding planar graphs with inaccuracies. In *GD*, volume 894 of *Lecture Notes in Computer Science*, pages 254–261. Springer, 1994.

5 Maarten Löffler. *Data Imprecision in Computational Geometry*. PhD thesis, Utrecht University, 2009.

6 Maarten Löffler. Existence and computation of tours through imprecise points. *Int. J. Comput. Geom. Appl.*, 21(1):1–24, 2011.

7 Takao Nishizeki and Md. Saidur Rahman. *Planar Graph Drawing*, volume 12 of *Lecture Notes Series on Computing*. World Scientific, 2004.

8 Marcel Roeloffzen. Finding structures on imprecise points. Master's thesis, TU Eindhoven, 2009.

9 Rodrigo I. Silveira, Bettina Speckmann, and Kevin Verbeek. Non-crossing paths with geographic constraints. *Discret. Math. Theor. Comput. Sci.*, 21(3), 2019.

Oriented Spanners

Kevin Buchin[1], Joachim Gudmundsson[2], Antonia Kalb[1], Aleksandr Popov[*3], Carolin Rehs[1], André van Renssen[2], and Sampson Wong[2]

1 **Technical University of Dortmund, Germany**
`kevin.buchin, antonia.kalb, carolin.rehs@tu-dortmund.de`

2 **University of Sydney, Australia**
`joachim.gudmundsson, andre.vanrenssen@sydney.edu.au, sampson.wong123@gmail.com`

3 **TU Eindhoven, The Netherlands** `a.popov@tue.nl`

Abstract

Given a point set P and a parameter t, a (directed) geometric t-spanner $G = (P, E)$ is a subgraph of the complete (bi-directed) graph such that for every pair of points the shortest path in G is at most a factor t longer than the corresponding Euclidean distance.

In this paper, we introduce a new model: the oriented (geometric) spanner. This is a directed graph where every edge is one-way, i.e. $(p, p') \in E \implies (p', p) \notin E$. We investigate bounds for the minimal dilation. For 2-dimensional point sets, we prove NP-hardness of computing a minimal oriented dilation graph. For 1-dimensional point sets, 1-spanners are trivial to obtain. Therefore, we restrict the graphs: We present a dynamic program to compute a 1-page planar minimal oriented dilation graph in $O(n^8)$ time and a greedy algorithm to compute a constant dilation spanner in $O(n^3)$ time.

1 Introduction

1.1 Problem Setup and Related Work

Spanners are a well studied problem [2, 11, 13]: Given a set P of n points and a parameter t, we want to compute a subgraph G of the complete (bi-directed) graph such that for every $p, p' \in P$ the shortest path from p to p' in G is at most a factor t longer than the shortest path in the complete (bi-directed) graph, that is the (directed) edge (p, p').

We introduce a new model, *oriented spanners*. This is a directed spanner $G = (P, E)$ with $E \subset P \times P$ where $(p, p') \in E$ implies $(p', p) \notin E$. As in an oriented graph the length of the shortest path from p to p' and back is not twice their distance, the term of dilation changes. This new measurement for oriented graphs is similar to the dilation of round trip spanners [5, 6].

Like (directed) spanners, oriented spanners can be motivated by infrastructure. For example, the new constraint to one-way roads or tracks becomes interesting in applications with space limitations. Another use case for oriented spanners are communication networks. E.g. to exclude a hack, we could require that two neighbouring devices must not exchange data by the same direct connection.

The spanner-problem has many variations, e.g. spanners may be required to be planar [2] or have bounded degree [1, 9]. It is NP-hard to compute a minimal dilation graph, if the number of edges is bounded [4, 7] or planarity is required [3, 10]. Such restrictions are also of interested for oriented spanners.

[*] Supported by the Dutch Research Council (NWO) under the project number 612.001.801.

In this paper, we firstly define oriented spanners. As introduction to this new model we analyse bounds of the oriented dilation (see Section 2) and the hardness of computing a minimal oriented dilation graph (see Section 3). We show that oriented spanners for 1-dimensional points sets can be computed efficiently.

1.2 Basic Definitions

A spanner is a tuple $G = (P, E)$ of a given *point set* P and *edges* E. It is a subgraph of the complete graph $K(P) = (P, P \times P \setminus \{(p, p) \mid p \in P\})$ (*loops* are not allowed). In this paper, all edges are *directed*, i.e. the edge (p, p') goes from p to p'. By $\Delta p_1p_2p_3$ we denote the directed triangle of these points, that is the edge set $\{(p_1, p_2), (p_2, p_3), (p_3, p_1)\}$.

In this paper, we only consider geometric spanners. For geometric spanners, the *length of an edge* (p, p') is the Euclidean distance $|p - p'|$. The *length of a path or cycle* is the sum of the length of its edges. In general, for a (un-)directed t-spanner G, the dilation $t = \max\left\{\frac{|d_G(p,p')|}{|p-p'|} \mid p, p' \in P\right\}$ represents the relation of the shortest path $d_G(p, p')$ from p to p' in G to the shortest path in $K(P)$, that is the edge (p, p').

▶ **Definition 1.1** (minimal dilation graph). For a point set P, a t-spanner is called a *minimal dilation graph* for P, if there is no t'-spanner for P with $t' < t$.

An *oriented graph* satisfies $(p, p') \in E \implies (p', p) \notin E$. By this restriction, if the relation of $d_G(p, p')$ to $|p-p'|$ is minimized by adding the edge (p, p'), the dilation in the other direction is never 1. Therefore, considering the dilation as $t = \max\left\{\frac{|d_G(p,p')|}{|p-p'|} \mid p, p' \in P\right\}$, would not tell much about the quality of an oriented spanner. This means to obtain meaningful results, we need to define the term of *oriented dilation*.

By $C_G(p, p')$ we denote the *shortest oriented cycle* containing the points p and p' in an oriented graph G. The *optimal oriented cycle* $\Delta(p, p')$ for two points $p, p' \in P$ is the shortest oriented cycle containing p and p' in $K(P)$. Notice, $\Delta(p, p')$ is the triangle $\Delta pp'p''$ with $p'' = \underset{p^* \in P}{\arg\min} |p - p^*| + |p^* - p'|$.

▶ **Definition 1.2** (oriented dilation). Given a point set P and an oriented graph G for P, the *oriented dilation of two points* $p, p' \in P$ is defined as

$$\mathrm{odil}(p, p') = \frac{|C_G(p, p')|}{|\Delta_G(p, p')|}.$$

The *dilation* t *of an oriented graph* is defined as $t = \max\{\mathrm{odil}(p, p') \mid \forall p, p' \in P\}$.

The minimal oriented dilation graph is defined analogously to Definition 1.1. Since it is NP-hard to compute this graph for a 2-dimensional point set (Section 3), we investigate 1-dimensional point sets. In this paper, we number the points in a 1-dimensional point set P in increasing order, i.e. p_1 is the left most point in P.

▶ **Definition 1.3** (1-page planar). Let P be a 1-dimensional point set. Let $\mathbb{H}$ be the closed half plane defined by the dimension axis of P. A graph $G = (P, E)$ is called 1*-page planar*, if G can be embedded in $\mathbb{R}^2$ such that each edge in E lies in $\mathbb{H}$ and no edges intersect. An edge may be bend, this does not change its length. (Compare to book embedding of graphs [8].)

A 1-page planar graph $G = (P, E)$ is called *maximal* 1*-page planar*, if for every edge $e \notin E$ the graph $G' = (P, E \cup \{e\})$ is not 1-page planar. We call the edge set $\{(p_i, p_{i+1}) \mid \forall p_i \in P, i \leq |P| - 1\}$ the *baseline*. A directed edge (p_j, p_i) with $i < j$ is called a *back edge*.

In the following, we call a 1-page planar oriented spanner that includes a baseline and all other edges are back edges, a 1-*PPB* spanner. Lemma 1.4 shows that a 1-PPB spanner has a smaller dilation than a spanner with the same edge set but a different orientation. Without loss of generality, we consider only 1-PPB spanners.

▶ **Lemma*** **1.4** (oriented dilation of 1-PPB)**.** *Let $G = (P, E)$ be a 1-page planar oriented t-spanner for a 1-dimensional point set P. Let $G' = (P, E')$ be a 1-PPB t'-spanner for P, we achieve by flipping the orientation of some edges of E and adding missing baseline edges. It holds $t' \le t$.*

2 Bounds for Minimal Oriented Dilation

In this section, we bound the minimal dilation of oriented spanners for 1- and 2-dimensional point sets, to motivate the problem of computing a minimal dilation graph (Section 3).

▶ **Theorem 2.1.** *There are 2-dimensional point sets for which no oriented 1-spanner exists.*

Proof. We show that there is no 1-spanner for the point set P where p_1, p_2 and p_4 form an equilateral triangle and p_3 is its centre (compare to Figure 1). The optimal oriented cycle $\Delta(p_i, p_j)$ with $i, j \in \{1, 2, 4\}$ is $\Delta p_i p_3 p_j$ or $\Delta p_j p_3 p_i$. Each of these triangles is optimal for $\Delta(p_3, p_i)$. If there is an oriented 1-spanner $G = (P, E)$, these triangles have to be contained in E. Ignoring the orientation, the union of these triangles is the complete graph K_4. Therefore, E contains no further edges. It remains to orientate the edges. But, there is no possibility to orientate the edges of K_4 such that for each $i, j \in \{1, 2, 4\}$ the optimal triangle $\Delta p_i p_3 p_j$ or $\Delta p_j p_3 p_i$ is contained. ◀

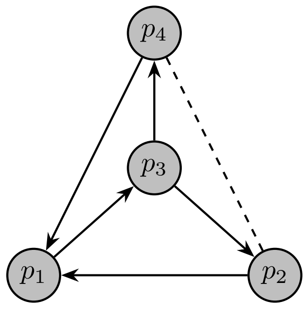


Figure 1 No possibility to orientate the faces of K_4 consistently

▶ **Lemma*** **2.2** (bound for oriented dilation in 1 dimension)**.** *Let P be a 1-dimensional point set. The oriented dilation of any t-spanner for P is bounded by*

$$t \le \max\{\mathrm{odil}(p_i, p_{i+2}), \mathrm{odil}(p_i, p_{i+3}) \mid \forall p_i \in P\}.$$

Lemma 2.2 gives a non-planar oriented 1-spanner for every 1-dimensional point set. Note, that this graph is even sparse, i.e. $|E| \in O(|P|)$.

▶ **Corollary 2.3.** *For every 1-dimensional point set P, the non-planar oriented graph $G = (P, E)$ with $E = \{(p_i, p_{i+1}), (p_{i+2}, p_i), (p_{i+3}, p_i) \mid \forall p_i \in P\}$ is a 1-spanner for P.*

The proofs of results marked by * are given in the full version.

▶ **Lemma*** **2.4** (bound for oriented dilation of 1-PPB). *Let P be a 1-dimensional point set. The oriented dilation of a 1-PPB t-spanner for P is bounded by*

$$t \leq \max\{\mathrm{odil}(p_i, p_{i+2}) \mid \forall p_i \in P\}.$$

If a spanner $G = (P, E)$ with $|P| > 3$ is 1-page planar, there is a tuple $p_i, p_{i+2} \in P$ where $C_G(p_i, p_{i+2}) \neq \Delta(p_i, p_{i+2})$. Combining this with Lemma 2.4, we follow

▶ **Corollary 2.5.** *There is no 1-page planar 1-spanner for any point set P with $|P| > 3$.*

Let $G = (P, E)$ be a graph with $|P| = 5$, $E = \{(p_i, p_{i+1}) \mid \forall p_i \in P\} \cup \{(p_3, p_1), (p_5, p_3)\}$ and the distances $|p_1 - p_2| = |p_4 - p_5| = \epsilon$ and $|p_2 - p_3| = |p_3 - p_4| = 1$. For an arbitrary small ϵ, G is an *almost* 1-spanner. Further, holds

▶ **Theorem 2.6.** *There are 1-dimensional point sets where no 1-page planar oriented t-spanner exists for $t < 2$.*

Proof. We show that $t \geq 2$ holds for every t-spanner of the point set P with $|P| = 5$ where each consecutive pair of points has distance 1. Since a 1-PPB spanner has a smaller dilation than a spanner with the same edge set but another orientation (Lemma 1.4) and maximality decreases dilation, it suffices to look at all maximal 1-PPB graphs for P. These are listed in Figure 2. Since each consecutive pair of points has distance 1, it holds $|\Delta(p_i, p_{i+2})| = 4$. For every maximal 1-PPB graph G for P, there is a tuple $p_i, p_{i+2} \in P$ where $C_G(p_i, p_{i+2})$ visits every point, i.e. $C_G(p_i, p_{i+2}) = (p_1, p_2), \ldots, (p_4, p_5), (p_5, p_1)$. In Figure 2a) that is p_2, p_4, in b,c) p_3, p_5 and in d,e) p_1, p_3. Combining these observations with Lemma 2.4, gives

$$t \geq \max\left\{\frac{|C_G(p_i, p_{i+2})|}{|\Delta(p_i, p_{i+2})|} \mid \forall p_i \in P\right\} = \frac{\max\limits_{p_i \in P} |C_G(p_i, p_{i+2})|}{4} = \frac{|p_1 - p_5| \cdot 2}{4} = \frac{8}{4} = 2$$

for the dilation t of every 1-page planar oriented spanner for P. ◀

Algorithm 1 returns a constant dilation spanner for any given set of n 1-dimensional points in $O(n^3)$ time. Theorem 2.6 is the biggest tight lower bound we have proven jet. We do not know the maximal dilation of an optimal 1-page planar oriented spanner and if Algorithm 1 is worst-case optimal.

▶ **Corollary 2.7.** *There is a 1-PPB $O(1)$-spanner for every 1-dimensional point set.*

Algorithm 1 Greedy $O(1)$-spanner

```
Require: 1-dimensional point set $P = \{p_1, \ldots, p_n\}$ (numbered from left to right)
Ensure: 1-page planar oriented $O(1)$-spanner for $P$
  Sort edges in $E' = \{(p_j, p_i) \mid \forall\ i+2 \leq j \leq n\}$ ascending by length // possible back edges
  for each $(p_j, p_i) \in E'$ do
    if $\nexists (p_r, p_l) \in E$ with $(l < i < r < j)$ or $(i < l < j < r)$ then   // preserve planarity
      $E \leftarrow E \cup \{(p_j, p_i)\}$                                   // add back edge $(p_j, p_i)$
    end if
  end for
  return $G = (P, E' \cup \{(p_i, p_{i+1}) \mid \forall\ 1 \leq i \leq n-1\})$                 // add baseline
```

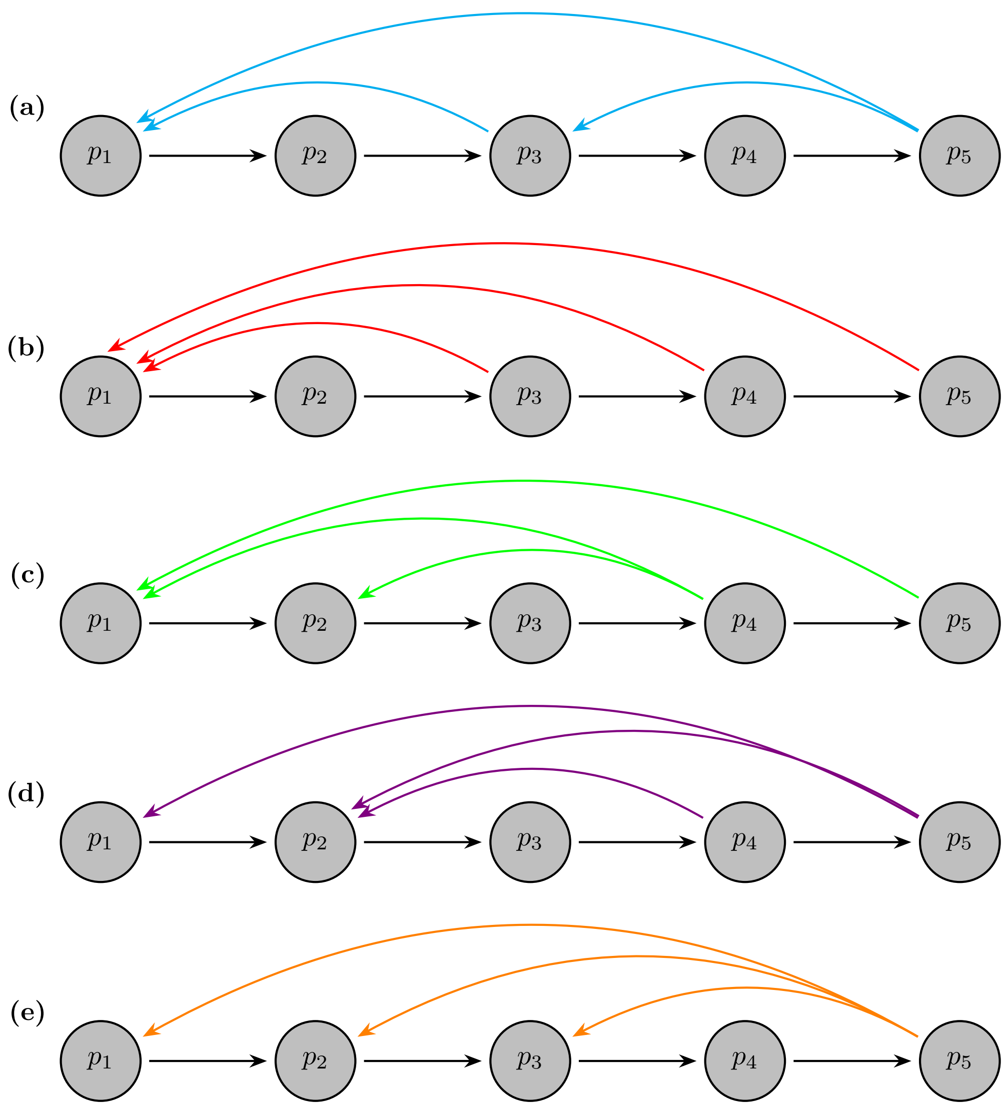


Figure 2 All maximal 1-PPB graphs for a 1-dimensional set of 5 points

3 Hardness

Since there is no oriented 1-spanner for every 2-dimensional point set (Theorem 2.1), we are interested in computing a minimal oriented dilation graph.

▶ **Theorem 3.1.** *Let P be a 2-dimensional set of n points. Given a parameter $m' \geq n$, it is NP-hard to compute a minimal oriented dilation graph $G = (P, E)$ with $|E| \leq m'$.*

Proof. We prove the problem is NP-hard even for $m' = n$. We reduce the problem of constructing an oriented minimal dilation cycle ($m' = n$) to the Euclidean Travelling Salesman Problem (EUCLIDEAN-TSP is NP-hard [12]). The denominator in the definition of (oriented) dilation is fixed by the point set. So, the oriented dilation is minimized by minimizing the largest shortest oriented cycle over all pairs of points. Since a minimal dilation cycle is searched ($m' = n$), the shortest oriented cycle is the same for each pair of points. Therefore, to compute a minimal oriented dilation graph with n edges, is the same as searching for the shortest Hamilton cycle. That is EUCLIDEAN-TSP. ◀

For a 1-dimensional point set, unlike non-planar oriented spanners (Theorem 2.3), the minimal dilation of a 1-page planar spanner is not always 1 (Theorem 2.5). For a given set of n points, Algorithm 2 computes in $O(n^8)$ time a 1-PPB minimal dilation graph. The dynamic program bases on the following idea (compare to Figure 3): We want to minimize the dilation t of a graph for $\{p_l, \dots, p_r\}$. Due to planarity, if $(p_r, p_l) \in E$, it holds $(p_j, p_i) \notin E$ for $l < i < r < j$ and $i < l < j < r$. We test all candidates for a split point p_k. By Lemma 2.4, it holds $t = \max\{t', t'', \mathrm{odil}(p_{k-1}, p_{k+1})\}$ where t' is the minimal dilation for $\{p_l, \dots, p_k\}$ and t'' is the minimal dilation for $\{p_k, \dots, p_r\}$. Point p_{k-1} and p_{k+1} is the only tuple of distance 2 (compare to Lemma 2.4) which crosses the split point p_k. Maintaining the parameters l' and r' such that $(p_{l'}, p_l) \in C_G(p_l, p_{l+1})$ and $(p_r, p_{r'}) \in C_G(p_{r-1}, p_r)$, we can realise $C_G(p_{k-1}, p_{k+1}) = C_G(p_{k-1}, p_k) \cup C_G(p_k, p_{k+1})$ to separate the problem at p_k (compare to Figure 3a)). Each recursion terminates with a forbidden combination of l,l',r' and r or with a set of 3 points.

▶ **Corollary 3.2.** *Given a 1-dimensional point set P, computing a 1-page planar minimal oriented dilation graph for P needs polynomial time.*

4 Open Problems

This paper is a first introduction to this new concept of the oriented spanners. Since it is NP-hard to compute a minimal oriented dilation graph for a 2-dimensional point set, we focused on 1-dimensional point sets. The 1-dimensional results might be expanded to an approximation algorithm for 2-dimensional point sets, which we plan to present in the future.

We prove that there are 1-dimensional point sets for which every 1-page planar oriented spanner has a dilation bigger or equal than 2 and we present a greedy algorithm that computes a 1-page planar oriented $O(1)$-spanner for every set of points. It remains to close this gap by proving tighter bounds.

References

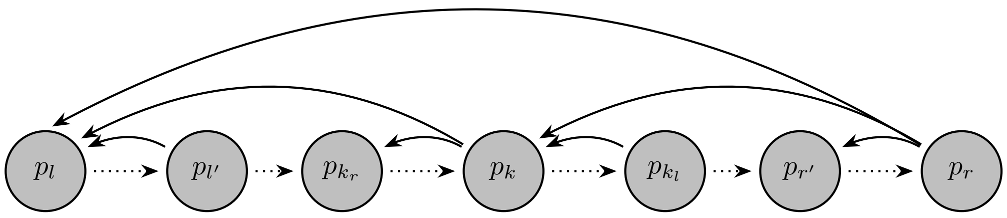
1 Ahmad Biniaz, Prosenjit Bose, Jean-Lou De Carufel, Cyril Gavoille, Anil Maheshwari, and Michiel Smid. Towards plane spanners of degree 3. *Journal of Computational Geometry*, 8(1):11–31, 2017.

Algorithm 2 Minimal dilation spanner

```
Require: 1-dimensional point set $P = \{p_1, \dots, p_n\}$ (numbered from left to right)
Ensure: 1-page planar minimal oriented dilation spanner for $P$
  Initialise table $\textsc{oE} = [1, \dots, n] \times [1, \dots, n] \times [1, \dots, n] \times [1, \dots, n]$
  // $\textsc{odil}(E')$ returns the oriented dilation of 1-PPB spanner with $E'$ as back edges
  // $\textsc{oE}(l, l', r', r)$= back edges set $E'$ of a minimal dilation spanner $G$ for $\{p_l, \dots, p_r\}$ with
  $(p_{l'}, p_l) \in C_G(p_l, p_{l+1})$ and $(p_r, p_{r'}) \in C_G(p_{r-1}, p_r)$
  Fill table by dynamic program based on the recursion formula:
```

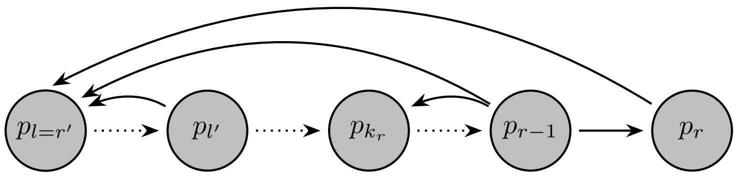
$$\textsc{oE}(l,l',r',r) = \begin{cases} \text{“invalid”, if } l' < l+2,\ r' > r-2 \text{ or } l > r \text{ (contradicts definition)} & (1) \\ \text{“invalid”, if } r < l+2 \text{ (no oriented spanners for } |P| < 3) & (2) \\ (p_r, p_l), \text{ if } l' = r \text{ and } r' = l \text{ (spanner for } |P| = 3 \text{ is a cycle)} & (3) \\ \text{“invalid”, if } l' < r,\ r' > l \text{ and } l' > r' \text{ (contradicts planarity)} & (4) \\ E' = \textsc{oE}(l, l', k_r, r-1) \cup \{(p_r, p_l)\}, \text{ if } l' < r \text{ and } r' = l, \text{ choose } l \leq k_r \leq r-3 \text{ s.t. } \textsc{odil}(E') \text{ is minimal} & (5) \\ E' = \textsc{oE}(l+1, k_l, r', r) \cup \{(p_r, p_l)\}, \text{ if } l' = r \text{ and } r' > l, \text{ choose } l+3 \leq k_l \leq r \text{ s.t. } \textsc{odil}(E') \text{ is minimal} & (6) \\ E' = \textsc{oE}(l, l', k_r, k) \cup \textsc{oE}(k, k_l, r', r) \cup \{(p_r, p_l)\}, \text{ if } l' < r,\ r' > l \text{ and } l' \leq r', \text{ choose } l \leq k_r \leq k-2,\ l+2 \leq k \leq r-2 \text{ and } k+2 \leq k_l \leq r \text{ s.t. } \textsc{odil}(E') \text{ is minimal} & (7) \end{cases}$$

```
  $E' \leftarrow \textsc{oE}(1, l', r', n)$, choose $l'$ and $r'$ s.t. $\textsc{odil}(E')$ is minimal
  return $G = (P, E' \cup \{(p_i, p_{i+1}) \mid \forall\ 1 \leq i \leq n-1\})$        // add baseline
```



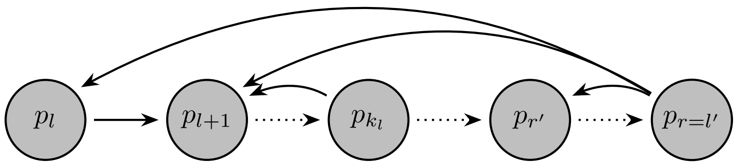
(a)

$\mathrm{OE}(l, l', r', r) = \mathrm{OE}(l, l', k_r, k) \cup \mathrm{OE}(k, k_l, r', r) \cup \{(p_r, p_l)\}$



(b)

$\mathrm{OE}(l, l', l, r) = \mathrm{OE}(l, l', k_r, r-1) \cup \{(p_r, p_l)\}$



(c)

$\mathrm{OE}(l, r, r', r) = \mathrm{OE}(l+1, k_l, r', r) \cup \{(p_r, p_l)\}$

Figure 3 Visualisation of the recursive cases of the recursion formula in Algorithm 2

2 Prosenjit Bose and Michiel Smid. On plane geometric spanners: A survey and open problems. *Journal of Computational Geometry*, 46(7):818–830, 2013.

3 Ulrik Brandes and Dagmar Handke. NP-completness results for minimum planar spanners. In *Graph-Theoretic Concepts in Computer Science*, pages 85–99. Springer, 1997.

4 Leizhen Cai. NP-completeness of minimum spanner problems. *Discrete Applied Mathematics*, 48(2):187–194, 1994.

5 Lenore J. Cowen and Christopher G. Wagner. Compact roundtrip routing for digraphs. In *Proceedings of the Tenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 885–886. Society for Industrial and Applied Mathematics, 1999.

6 Lenore J. Cowen and Christopher G. Wagner. Compact roundtrip routing in directed networks. *Journal of Algorithms*, 50(1):79–95, 2004.

7 Panos Giannopoulos, Rolf Klein, Christian Knauer, Martin Kutz, and Dániel Marx. Computing geometric minimum-dilation graphs is NP-hard. *Journal of Computational Geometry*, 20(2):147–173, 2010.

8 Xiaxia Guan, Chuxiong Wu, Weihua Yang, and Jixiang Meng. A survey on book-embedding of planar graphs. *Frontiers of Mathematics in China*, 17(2):255–273, 2022.

9 Iyad Kanj, Ljubomir Perkovic, and Duru Turkoglu. Degree four plane spanners: Simpler and better. *Journal of Computational Geometry*, 8(2):3–31, 2017.

10 Yusuke Kobayashi. NP-hardness and fixed-parameter tractability of the minimum spanner problem. *Theoretical Computer Science*, 746:88–97, 2018.

11 Giri Narasimhan and Michiel Smid. *Geometric Spanner Networks.* Cambridge University Press, 2007.

12 Christos Papadimitriou. The euclidean traveling salesman problem is NP-complete. *Theoretical Computer Science*, 4(3):237–244, 1977.

13 David Peleg and Alejandro Schäffer. Graph spanners. *Journal of Graph Theory*, 13(1):99–116, 1989.

Insertion-Only Dynamic Connectivity in General Disk Graphs

Haim Kaplan[1], Katharina Klost[2], Kristin Knorr[*2], Wolfgang Mulzer[†2], and Liam Roditty[3]

1 **School of Computer Science, Tel Aviv University**
`haimk@tau.ac.il`

2 **Institut für Informatik, Freie Universität Berlin**
`{kathklost, knorrkri, mulzer}@inf.fu-berlin.de`

3 **Department of Computer Science, Bar Ilan University**
`liamr@macs.biu.ac.il`

Abstract

Let $S \subseteq \mathbb{R}^2$ be a set of n *sites* in the plane, so that every site $s \in S$ has an *associated radius* $r_s > 0$. Let $\mathcal{D}(S)$ be the *disk intersection graph* defined by S, i.e., the graph with vertex set S and an edge between two distinct sites $s, t \in S$ if and only if the disks with centers s, t and radii r_s, r_t intersect. Our goal is to design data structures that maintain the connectivity structure of $\mathcal{D}(S)$ as S changes dynamically over time.

We consider the incremental case, where new sites can be inserted into S. While previous work focuses on data structures whose running time depends on the ratio between the smallest and the largest site in S, we present a data structure with $O(\alpha(n))$ amortized query time and $O(\log^5 n)$ expected amortized insertion time.

1 Introduction

The question if two vertices in a given graph are connected is crucial for many applications. If multiple such *connectivity queries* need to be answered, it makes sense to preprocess the graph into a suitable data structure. In the static case, where the graph does not change, we get an optimal answer by using a graph search to determine the connected components and by labeling the vertices with their respective components. In the dynamic case, where the graph can change over time, things get more interesting, and many variants of the problem have been studied.

We construct an insertion-only dynamic connectivity data structure for disk graphs. Given a set $S \subseteq \mathbb{R}^2$ of n sites in the plane with associated radii r_s for each site s, the *disk graph $\mathcal{D}(S)$ for S* is the intersection graph of the disks D_s induced by the sites and their radii. While $\mathcal{D}(S)$ is represented by $O(n)$ numbers describing the disks, it might have $\Theta(n^2)$ edges. Thus, when we start with an empty disk graph and successively insert sites, up to $\Omega(n^2)$ edges may be created. We describe a data structure whose overall running time for any sequence of n site insertions is $o(n^2)$, while allowing for efficient connectivity queries. For unit disk graphs (i.e., all associated radii $r_s = 1$), a fully dynamic data structure with a similar performance guarantee is already given by Kaplan et al. [2]. In the same paper, Kaplan et al. present an incremental data structure whose running time depends on the ratio

* Supported by the German Science Foundation within the research training group 'Facets of Complexity' (GRK 2434).

† Supported in part by ERC StG 757609.

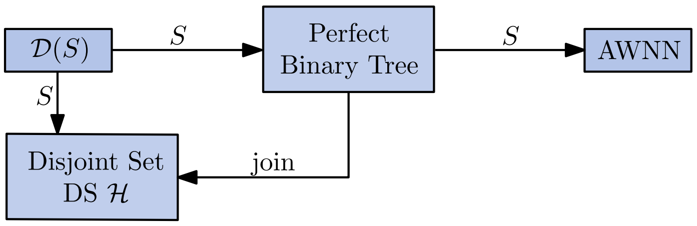


Figure 1 Overview of the data structure (Theorem 2.5)

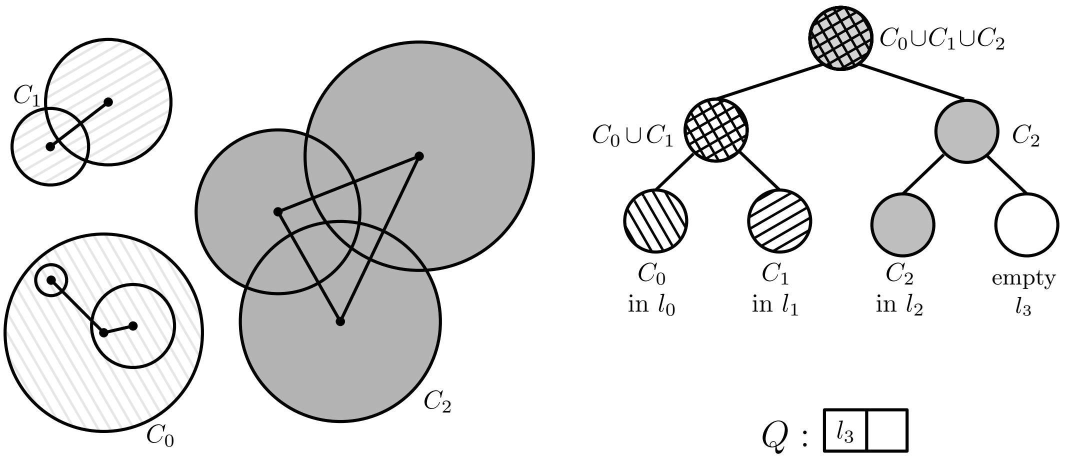


Figure 2 Disk graph with associated component tree and queue Q (tiling of a component corresponds to the tiling of nodes storing its disks).

Ψ of the smallest and the largest radius in S. In this setting, they achieve $O(\alpha(n))$ amortized query time and $O(\log(\Psi)\log^4 n)$ expected amortized insertion time.

We focus on general disk graphs, with no assumption on the radius ratio. Our approach has two main ingredients. First, we simply represent the connected components in a data structure for the disjoint set union problem [1]. This allows for fast queries, in $O(\alpha(n))$ amortized time, also mentioned by Reif [5]. The second ingredient is an efficient data structure to find all components in $\mathcal{D}(S)$ that are intersected by any given disk (and hence tells us which components need to be merged after an insertion of a new site). A schematic overview of the data structure is given in Figure 1.

2 Insertion-Only Data Structure for Unbounded Radii

As in the incremental data structure for disk graphs with bounded radius ratio by Kaplan et al. [2], we use a disjoint set union data structure to represent the connected components of $\mathcal{D}(S)$ and to perform the connectivity queries. To insert a new site s into S, we first find the set $\mathcal{C}_s$ of components in $\mathcal{D}(S)$ that are intersected by D_s. Then, once $\mathcal{C}_s$ is known, we can simply update the disjoint set union structure to support further queries.

In order to identify $\mathcal{C}_s$ efficiently, we use a *component tree* $T_{\mathcal{C}}$. This is a binary tree whose leaves store the connected components of $\mathcal{D}(S)$. The idea is illustrated in Figure 2 showing a disk graph and its associated component tree. We require that $T_{\mathcal{C}}$ is a complete binary tree, and some of its leaves may not have a connected component assigned to them, like l_3 in Figure 2. Those leaves are *empty*. Typically, we will not distinguish the leaf

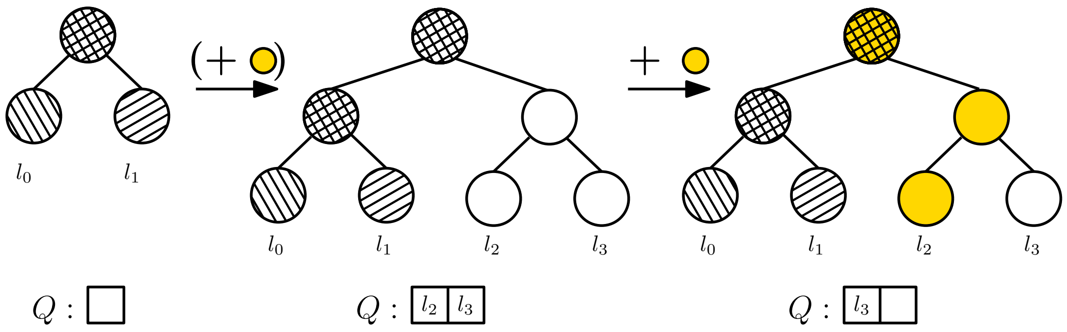


Figure 3 If the tree has no empty leaves before the insertion of an isolated component, a new root and an empty subtree are added (second tree is an intermediate state before actual insertion).

storing a connected component and the component itself. Also when suitable, we will treat a connected component as a set of sites. Every node of $T_{\mathcal{C}}$ stores a fully dynamic additively weighted nearest neighbor data structure (AWNN). An AWNN stores a set P of n points, each associated with a weight w_p. On a nearest neighbor query with a point $q \in \mathbb{R}^2$ it returns the point $p \in P$ that minimizes $\|pq\| + w_p$. For this data structure, we use the following result by Kaplan et al. [3] with an improvement by Liu [4].

▶ **Lemma 2.1** (Kaplan et al. [3, Theorem 8.3, Section 9], Liu [4, Corollary 4.3]). *There is a fully dynamic AWNN data structure that allows insertions in $O(\log^2 n)$ amortized expected time and deletions in $O(\log^4 n)$ amortized expected time. Furthermore, a nearest neighbor query takes $O(\log^2 n)$ worst case time. The data structure requires $O(n \log n)$ space.*

The component tree maintains the following invariants. Invariant 2 allows us to use a query to the AWNN to find the disk whose boundary is closed to a query point.

Invariant 1: Every connected component of $\mathcal{D}(S)$ is stored in exactly one leaf of $T_{\mathcal{C}}$, and

Invariant 2: The AWNN of a node $u \in T_{\mathcal{C}}$ contains the sites of all connected components that lie in the subtree rooted at u, where a site $s \in S$ has assigned weight $-r_s$.

In addition to $T_{\mathcal{C}}$, we store a queue $Q_{\mathcal{C}}$ that contains exactly the empty leaves in $T_{\mathcal{C}}$.

We now describe how to update $T_{\mathcal{C}}$ when a new site s is inserted. We maintain $T_{\mathcal{C}}$ in such a way that the structure of $T_{\mathcal{C}}$ changes only when s creates a new isolated component in $\mathcal{D}(S)$. In the following lemma, we consider the slightly more general case of inserting a new connected component C that does not intersect any connected component already stored in $T_{\mathcal{C}}$.

▶ **Lemma 2.2.** *Let $T_{\mathcal{C}}$ be a component tree of height h with n sites and let C be an isolated connected component. We can insert C into $T_{\mathcal{C}}$ in amortized time $O(h \cdot |C| \cdot \log^2 n)$.*

Proof. The insertion performs two basic steps: first, we find or create an empty leaf l_i into which C can be inserted. Second, the AWNN structures along the path from l_i to the root of $T_{\mathcal{C}}$ are updated.

For the first step, we check if $Q_{\mathcal{C}}$ is non-empty. If so, we extract the first element from $Q_{\mathcal{C}}$ to obtain our empty leaf l_i. If $Q_{\mathcal{C}}$ is empty, there are no empty leaves, and we have to expand the component tree. For this, we create a new root for $T_{\mathcal{C}}$, and we attach the old tree as one child. The other child is an empty complete tree of the same size as the old tree. This creates a complete binary tree, see intemediate state in Figure 3. We copy the AWNN of the former root to the new root, we add all new empty leaves to $Q_{\mathcal{C}}$, and we extract l_i from $Q_{\mathcal{C}}$.

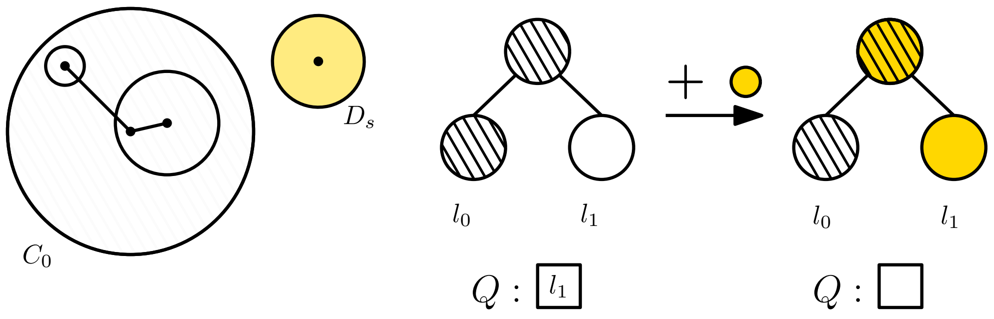


Figure 4 Inserting a component (potentially isolated disk D_s) into an empty leaf (C_i stored in l_i): The isolated component C, yellow disk D_s, is inserted in the empty leaf l_1 and all its ancestors (indicated by coloring).

For the second step, we insert C into l_i, and we store an AWNN structure with the sites from C in l_i. Then, the AWNN structures on all ancestors of l_i are updated by inserting the sites of C, see Figure 4 and Figure 3 in case of tree extension respectively.

This procedure maintains both invariants: since an isolated component does not affect the remaining connected components of $\mathcal{D}(S)$, it has to be inserted into a new leaf, maintaining the first invariant. The second invariant is taken care of in the second step, by construction. Afterwards, the queue has the correct state, since we extract the leaf used in the insertion.

The running time for finding or creating an empty leaf is amortized $O(1)$. This is immediate if $Q_\mathcal{C} \neq \emptyset$, and otherwise, we can charge the cost of building the empty tree, inserting the empty leaves into $Q_\mathcal{C}$, and producing an AWNN structure for the new root to the previous insertions. The most expensive step consists in updating the AWNN structures for the new component. In each of the h AWNN structures of the ancestors of l_i, we must insert $|C|$ disks. By Lemma 2.1, this results into an expected amortized time of $O(h \cdot |C| \cdot \log^2 n)$. ◀

Next, we describe how to find the set $\mathcal{C}_s$ of connected components that are intersected by a disk D_s.

▶ **Lemma 2.3.** *Let $T_\mathcal{C}$ be a component tree of height h that stores n disks. We can find $\mathcal{C}_s$ in worst case time $O(\max\{|\mathcal{C}_s| \cdot h, 1\} \cdot \log^2 n\})$.*

Proof. First, observe that if the site returned by a query to an AWNN structure with s does not intersect D_s, then D_s does not intersect any disk for the sites stored in this AWNN. Thus, the case where $\mathcal{C}_s = \emptyset$ can be identified by a query to the AWNN structure in the root of $T_\mathcal{C}$, in $O(\log^2 n)$ time.

In any other case, we perform a top down traversal of $T_\mathcal{C}$. Let u be the current node. We query the AWNN structures of both children of u with s, and we recurse only into the children where the nearest neighbor intersects D_s. The set $\mathcal{C}_s$ then contains exactly the connected components of all leaves where D_s intersects its weighted nearest neighbor. Since every leaf found corresponds to one connected component intersecting D_s and we recurse into all subtrees whose union of sites have a non-empty intersection with s, we do not miss any connected components.

For every connected component, there are at most h queries to AWNN structures along the path from the root to the components. A query takes $O(\log^2 n)$ amortized time by Lemma 2.1, giving an amortized time of $O(|\mathcal{C}_s| \cdot h \cdot \log^2 n)$, if s is not isolated. The overall time follows from taking the maximum of both cases. ◀

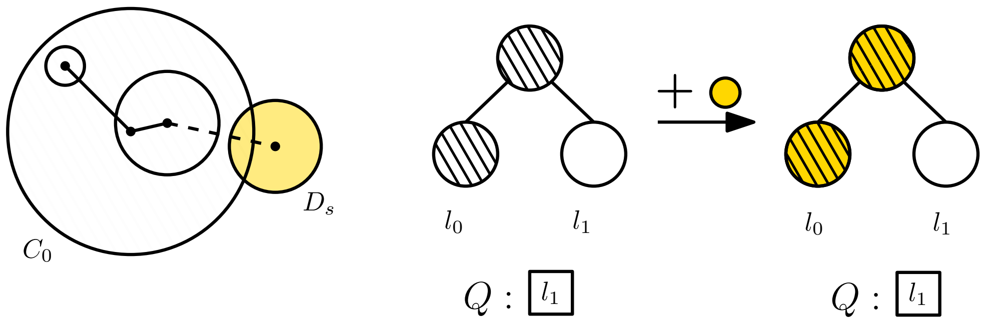


Figure 5 Inserting a site if $|\mathcal{C}_s| = 1$ (C_i stored in l_i): The yellow disk D_s intersects C_0 (dashed edge). Site s is added to the AWNN of the associated leaf l_0 and its ancestors.

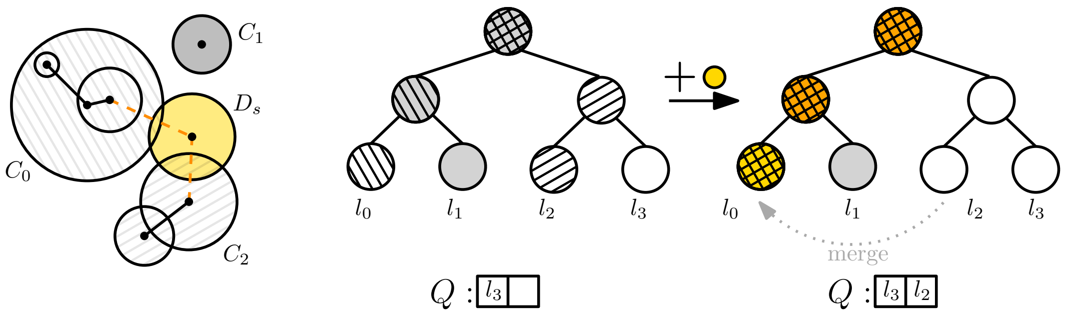


Figure 6 Inserting a site if $|\mathcal{C}_s| > 1$ (C_i stored in l_i): D_s intesects C_0 and C_2. Since $|C_0| = 3$ and $|C_2| = 2$, the largest component C_L is C_0. Thus, D_s and C_2 are merged into C_0 and C_2 is removed from l_2 up to the lca, the root. The empty leave l_2 is enqueued.

Using Lemma 2.2 and Lemma 2.3, we can now describe how to insert a single disk into a component tree.

▶ **Lemma 2.4.** *Let $T_\mathcal{C}$ be a component tree that stores n sites. A new site s can be inserted into $T_\mathcal{C}$ in $O(\log^5 n)$ amortized expected time.*

Proof. First, we use the algorithm from Lemma 2.3. to find $\mathcal{C}_s$. If $|\mathcal{C}_s| = 0$, we use Lemma 2.2 to insert s as a singleton isolated connected component.

Otherwise, if $|\mathcal{C}_s| \geq 1$, let $C_L = \arg\max_{C \in \mathcal{C}_s} |C|$ be a largest connected component in $\mathcal{C}_s$. We insert s into C_L and into the AWNN structures of all ancestors of C_L. Then, if $|\mathcal{C}_s| = 1$, we are done, see Figure 5. If $|\mathcal{C}_s| > 1$, all components in $\mathcal{C}_s$ now form a new, larger, component in $\mathcal{D}(S)$. We perform the following *clean-up* step in $T_\mathcal{C}$.

For each component $C_i \in \mathcal{C}_s \setminus \{C_L\}$, all sites from C_i are inserted into C_L. Let lca be the lowest common ancestor of the leaves for C_i and C_L in $T_\mathcal{C}$. Then all sites from C_i are deleted from the AWNN structures along the path from C_i to lca, and reinserted along the path from C_L to lca. Finally, all newly empty leaves are inserted into $Q_\mathcal{C}$. For an illustration of the insertion of s and the clean-up step, see Figure 6.

To show correctness, we again argue that the invariants are maintained. If $|\mathcal{C}_s| = 0$ this follows by Lemma 2.2. In the other case, we directly insert s into a connected component intersected by s and update all AWNN structures along the way. As Lemma 2.3 correctly finds all relevant connected components, and we explicitly move the sites in these components to C_L during clean-up, Invariant 1 is fulfilled. In a similar vein, we update all AWNN structures of sites that move to a new connected component, satisfying Invariant 2. Moreover, we keep

$Q_{\mathcal{C}}$ updated by inserting or removing empty leaves when needed during the algorithm.

To complete the proof, it remains to analyze the running time. In the worst case, where all components are singletons, a component tree that stores n sites has height $O(\log n)$. If $|\mathcal{C}_s| = 0$ the running time for finding $\mathcal{C}_s$ is $O(\log^2 n)$ by Lemma 2.3. The insertion and restructuring is done with Lemma 2.2, yielding an expected amortized time of $O(\log^3 n)$. In the case $|\mathcal{C}_s| = 1$, with $h = O(\log n)$ a running time of $O(\log^3 n)$ for finding $\mathcal{C}_s$ follows by Lemma 2.3. Following similar arguments to the case $|\mathcal{C}_s| = 0$, the time needed for the insertion and restructuring is expected amortized $O(\log^3 n)$.

Finally, we consider the case $|\mathcal{C}_s| > 1$. By Lemma 2.3, finding $\mathcal{C}_s$ takes worst case time $O(|\mathcal{C}_s| \cdot \log^3 n)$. Then the insertion of s can be done in expected amortized time $O(\log^3 n)$, as in the cases above. It remains to analyze the running time of the clean-up step. We know that the first common ancestor might be the root of $T_{\mathcal{C}}$. Hence, in the worst case, we have to perform $\sum_{C_i \in (\mathcal{C}_s \setminus \{C_L\})} |C_i| \cdot O(\log n)$ insertions and deletions for a single clean-up step. As the time for the deletions in the AWNN structures dominates, this is expected amortized $\sum_{C_i \in \mathcal{C}_s \setminus \{C_L\}} O(|C_i| \cdot \log^4 n)$ worst case time. Note that since $|C_i| \geq 1$ and we have to insert s, the running time of Lemma 2.3 is dominated by the clean-up step.

The overall time spent on all clean-up steps over all insertions is then upper bounded by $\sum_{s \in S} \sum_{C_i \in (\mathcal{C}_s \setminus \{C_L\})} O(|C_i| \cdot \log^4 n)$. Observe, that during the lifetime of the component tree, each disk can only be merged into $O(\log n)$ connected components. Thus, we have

$$\sum_{s \in S} \sum_{C_i \in \mathcal{C}_s \setminus \{C_L\}} |C_i| = O(n \log n),$$

and the overall expected time spent on clean-up steps is $O(n \log^5 n)$. As the case $|\mathcal{C}_s| > 1$ turned out to be the most complex case, the overall running time follows. ◀

▶ **Theorem 2.5.** *There is an incremental data structure for connectivity queries in disk graphs with $O(\alpha(n))$ amortized query time and $O(\log^5 n)$ expected amortized update time.*

Proof. We use a component tree as described above to maintain the connected components. Additionally, we maintain a disjoint set data structure $\mathcal{H}$, where each connected component forms a set, see Figure 1. Queries are performed directly in $\mathcal{H}$ in $O(\alpha(n))$ amortized time.

When inserting an isolated component during the update, this component is added to $\mathcal{H}$. When merging several connected components in the clean-up step, this change is reflected in $\mathcal{H}$ by suitable union operations. The time for updates in the component tree dominates the updates in $\mathcal{H}$, leading to an expected amortized update time of $O(\log^5 n)$. ◀

3 Conclusion

We introduced a data structure that solves the incremental connectivity problem in general disk graphs with $O(\alpha(n))$ amortized query and $O(\log^5 n)$ amortized expected update time. The question of finding efficient fully-dynamic data structures for both the general and the bounded case remains open.

References

1 Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms, Third Edition.* The MIT Press, 3rd edition, 2009.

2 Haim Kaplan, Alexander Kauer, Katharina Klost, Kristin Knorr, Wolfgang Mulzer, Liam Roditty, and Paul Seiferth. Dynamic connectivity in disk graphs. In *38th International*

Symposium on Computational Geometry (SoCG 2022), pages 49:1–49:17, 2022. doi:10.4230/LIPIcs.SoCG.2022.49.

3 Haim Kaplan, Wolfgang Mulzer, Liam Roditty, Paul Seiferth, and Micha Sharir. Dynamic planar Voronoi diagrams for general distance functions and their algorithmic applications. *Discrete Comput. Geom.*, 64(3):838–904, 2020. doi:10.1007/s00454-020-00243-7.

4 Chih-Hung Liu. Nearly optimal planar k nearest neighbors queries under general distance functions. *SIAM Journal on Computing*, 51(3):723–765, 2022. doi:10.1137/20m1388371.

5 John H. Reif. A topological approach to dynamic graph connectivity. *Information Processing Letters*, 25(1):65–70, 1987. doi:10.1016/0020-0190(87)90095-0.

An s-t Jordan curve crossing boundaries of a set of disk-homeomorphic objects in the plane

Aida Abiad[1], **Julian Golak**[2], **Alexander Grigoriev**[3], **and Freija van Lent**[4]

1 **Department of Mathematics and Computer Science, Eindhoven University of Technology**
Department of Mathematics: Analysis, Logic and Discrete Mathematics, Ghent University
Department of Mathematics and Data Science, Vrije Universiteit Brussel
a.abiad.monge@tue.nl

2 **Department of Data Analytics and Digitalisation, Maastricht University**
julian@golak.de

3 **Department of Data Analytics and Digitalisation, Maastricht University**
a.grigoriev@maastrichtuniversity.nl

4 **Department of Data Analytics and Digitalisation, Maastricht University**
f.vanlent@maastrichtuniversity.nl

Abstract

Let $\mathcal{S}_n$ be a set of n objects in the plane homeomorphic to disks, further referred to as shapes. Let G be a geometric intersection multigraph for $\mathcal{S}_n$, i.e., each shape in $\mathcal{S}_n$ is represented by a vertex in G, and two vertices of G are connected by k edges, where k is the number of intersection components between the two corresponding shapes. For the case of two shapes having Δ intersection components, we prove that for any two points in the plane there is a Jordan curve connecting these two points and crossing the boundaries of the shapes at most $2\Delta + 2$ times. For the general case of n objects, we prove that any two points in the plane can be connected by a Jordan curve that crosses the boundaries of the shapes at most $2M \min\{n, \Delta + 1\}$, where M is the number of inclusion-maximal intersection components and Δ is the maximum degree of G.

1 Introduction

Take any drawing of a finite number of shapes in the plane and choose two arbitrary, interior points in this embedding. An interior point is defined as a point that is not on a boundary of any shape. To connect these two points, one can draw a continuous line that crosses the boundaries of the shapes the least number of times. It is trivial to find an upper bound for the number of crossings of a line between two points in a given embedding of shapes: such line can be found using trial and error. However, we show that there exists an upper bound for the number of crossings of a minimum crossing Jordan curve between any two points in *any* embedding of n disk-homeomorphic objects in the plane that solely depends on the number of objects, n, and the maximum degree of the corresponding intersection multigraph, Δ.

A similar, but not identical, problem to bounding any minimum crossing curve between a point s and a point t has been widely studied in the area of sensor networks. Intrusion detection and border surveillance constitute a major application category for wireless sensor networks. One application is to detect intruders as they cross a border or enter a protected area. This type of coverage is referred to as *barrier coverage*, where the sensors form a barrier for the intruders. In [2], Kumar et al. specify the key notion of k-barrier coverage, which

39th European Workshop on Computational Geometry, Barcelona, Spain, March 29–31, 2023.
This is an extended abstract of a presentation given at EuroCG'23. 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.

holds if every path joining a point in S to a point in T must intersect at least k distinct sensor regions. The notion of barrier coverage closest to bounding a minimum crossing $\{s,t\}$-curve is discussed by Bereg and Kirkpatrick, see [1]. They define the thickness of a barrier as the minimum number of sensor region intersections of a path between any two regions.

In this work, we begin by introducing some preliminary definitions and lemmas, and providing a formal problem definition. In Section 2, we present an upper bound for any minimum crossing curve for embeddings of 2 shapes. Then, we extend this result for embeddings of n shapes in Section 3. Finally, we summarize our findings and discuss some open questions.

1.1 Preliminary Definitions and Lemmas

This section introduces several preliminary definitions that allow us to delve into the problem.

▶ **Definition 1.1.** A *{s,t}-simple curve* is a curve between an interior point s and an interior point t that does not intersect itself.

▶ **Definition 1.2.** A set of points in the plane is a 2-dimensional geometric shape S if it is *homeomorphic to the closed disk* $\{(x,y)|x^2+y^2 \leq 1\}$.

▶ Remark. $\mathcal{S}_n = \{S_1, \ldots, S_n\}$ is a collection of n shapes.

Important to note is that a shape S can be non-convex but cannot intersect itself.

▶ **Definition 1.3.** $\Gamma(S)$ is the closed Jordan curve that determines the *boundary of a shape* S.

▶ **Definition 1.4.** A *minimum crossing* $\{s,t\}$*-curve* is a {s,t}-simple curve that intersects boundaries of shapes the least number of times.

▶ Remark. We assume that, in any intersection of two shapes, one can at least fit an open ball of sufficiently small diameter. (**Assumption 1**)

This assumption is made to prevent shapes from solely touching instead of intersecting or having uncountably many boundary points in common.

In the following definition, a geometric intersection multigraph is introduced. Note that geometric intersection multigraphs are an extension of intersection graphs. Classical intersection graphs are simple graphs that include at most one edge between any two shapes [3]. Intersection multigraphs include a connecting edge for every intersection that any two shapes have in a given embedding, making it is a multigraph.

▶ **Definition 1.5.** An *intersection multigraph* $G^{\mathcal{A}}_{\mathcal{S}_n}$ is a multigraph where each shape of the collection $\mathcal{S}_n$ in an embedding $\mathcal{A}$ is represented by a vertex, and two vertices of $G^{\mathcal{A}}_{\mathcal{S}_n}$ are connected by e edges, with e being the number of connected components of the intersection between the corresponding shapes in embedding $\mathcal{A}$. Define the maximum degree of the intersection multigraph as Δ

▶ **Definition 1.6.** Given a collection of shapes $\mathcal{S}_n$, a *face* f is an open disc resulting from considering the plane without the boundaries of the shapes, i.e. $\mathbb{R}^2 \backslash \Gamma(\mathcal{S}_n)$. Define the *boundary of a face* f as $\Gamma(f)$, and $\mathcal{F}$ as the set of all faces in an embedding.

▶ **Definition 1.7.** The *dual graph* G^* of an embedding is the graph where each vertex $v(f)$ corresponds to a face f in the plane. Any two vertices $v(f)$ and $v(g)$ are connected if their corresponding faces f and g are adjacent to each other.

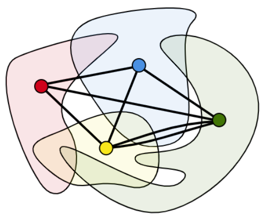


Figure 1 An example of an intersection multigraph, as defined in Definition 1.5.

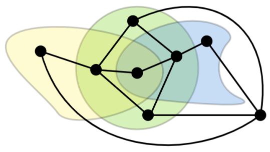


Figure 2 An example of a dual graph of an embedding of shapes, as defined in Definition 1.7.

Examples of an intersection multigraph and a dual graph can be found in Figures 1 and 2, respectively.

▶ **Lemma 1.8.** *For any path of length k in the dual graph, there is a corresponding minimum crossing curve in the embedding crossing k boundaries.*

▶ **Definition 1.9.** The *height* h of a face f is the number of shapes present in this face. We define the maximum height over all faces as h_{max}.

▶ **Lemma 1.10.** *A face f of height h can only be adjacent to faces of height $h-1$ or $h+1$.*

1.2 Problem definition and summary of results

Let $\mathcal{S}_n$ be a set of n shapes in an embedding $\mathcal{A}$ in the plane. Let $G^{\mathcal{A}}_{\mathcal{S}_n}$ be the geometric intersection multigraph for this embedding $\mathcal{A}$ of $\mathcal{S}_n$ with maximum degree Δ. This paper shows an upper bound for the number of crossings of a minimum crossing $\{s,t\}$-curve that solely depends on n and Δ in embeddings of 2 shapes and embeddings of n shapes.

2 Embeddings of 2 shapes

In this section, we present a linear upper bound for any minimum crossing $\{s,t\}$-curve in an embedding of two shapes. The main result is shown in Theorem 2.1.

▶ **Theorem 2.1.** *Given an embedding of two shapes S_1 and S_2 in $\mathbb{R}^2$, there exists a simple $\{s,t\}$-curve C crossing $\Gamma(S_1) \cup \Gamma(S_2)$ at most $2\Delta + 2$ times for any two points $s, t \in \mathbb{R}^2 \backslash (\Gamma(S_1) \cup \Gamma(S_2))$.*

To show that the result of Theorem 2.1 holds, we introduce some additional definitions.

▶ **Definition 2.2.** In an embedding of two shapes S_1 and S_2, a face f is an *intersection face* if $f \subseteq S_1 \cap S_2$.

▶ **Definition 2.3.** For a shape S and a face $f \subset S$, f is *chordal in S* if $S \backslash f$ consists of more than one component. Consequently, a face f is *non-chordal in S* if $S \backslash f$ is one component.

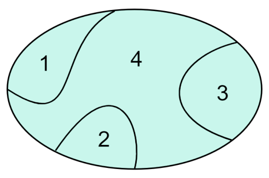


Figure 3 In this shape, face 4 is chordal and 1, 2 and 3 are non-chordal.

The proof of Theorem 2.1 requires Lemmas 2.5, 2.6 and 2.7. These lemmas are based upon the partition of the faces of shapes into three mutually-exclusive sets: intersection faces, chordal faces and non-chordal faces. The sets are defined in Definition 2.4 and a visual example of chordal and non-chordal faces is shown in Figure 3. Lemma 2.5 shows that for any interior points s, t in S_1, there exists a minimum crossing $\{s,t\}$-curve within S_1 that crosses the boundary of S_2 at most 2Δ times. This result is extended in Lemmas 2.6, 2.7 and 2.8 to eventually show in Theorem 2.1 that for any interior points $s, t \in \mathbb{R}^2$, the number of boundary crossings is at most $2\Delta + 2$.

▶ **Definition 2.4.** Given an embedding $\mathcal{A}$ of two shapes S_1 and S_2, every face $f \subseteq S_1$ can be assigned into one of the following three sets of faces:
1. $F_I = \{f | f \subseteq S_1 \cap S_2\}$, f is an intersection face; or
2. $F_C = \{f | f \subseteq S_1 \backslash S_2$ and f is chordal in $S_1\}$; or
3. $F_{NC} = \{f | f \subseteq S_1 \backslash S_2$ and f is not chordal in $S_1\}$.

By symmetry, a similar assignment into three sets of faces holds for every face $f \subseteq S_2$.

▶ Remark. Given an embedding of shapes S_1 and S_2, by Definition 2.4 and Lemma 1.10, faces assigned to the same set cannot be adjacent to each other.

▶ Remark. Given an embedding of shapes S_1 and S_2, by Definition 2.4 and Lemma 1.10, faces $f \in F_C$ and $g \in F_{NC}$ cannot be adjacent.

Using Definition 2.4 and the above remarks, we show the following lemmas.

▶ **Lemma 2.5.** *In an embedding of shapes S_1 and S_2 with interior points $s, t \in S_1$, there exists a minimum crossing $\{s,t\}$-curve $C_{s,t} \subset S_1$ crossing $\Gamma(S_2)$ at most 2Δ times.*

Proof. The intuition of the proof goes as follows. Since $C_{s,t} \subset S_1$, we only utilize faces $f \subset S_1$ and consider the subgraph $G^{\mathcal{A}_{S_1}}$ of the dual graph. Using Definition 1.7 and Lemma 1.8, partition the vertices of $G^{\mathcal{A}_{S_1}}$ into chordal, non-chordal and intersection vertices. This is similar to the three sets of faces in Definition 2.4. Clearly, $G^{\mathcal{A}_{S_1}}$ is a tripartite graph.

i. Case $\boldsymbol{\Delta} = 1$. Denote the unique intersection face by f_I. Any vertices $v(f), v(g)$ of faces $f, g \subset S_1 \backslash S_2$ are connected to $v(f_I)$ and thus there exists a path $P = v(f)v(f_I)v(g)$ of length $2\Delta = 2$. By Lemma 1.8, there also exists a simple $\{s,t\}$-curve in $\mathcal{A}$, for interior points $s, t \in S_1$, that crosses $\Gamma(S_1) \cup \Gamma(S_2)$ at most 2 times.

ii. Case $\boldsymbol{\Delta} \geq 2$. We bound the diameter of tripartite $G^{\mathcal{A}_{S_1}}$, as this will provide an upper bound on a shortest path between any two vertices $v(f)$ and $v(g)$, $f, g \subset S_1$. By Definition 2.3 and Lemma 1.10, a face of S_1 that is adjacent to two intersection faces is a chordal face of $S_1 \backslash S_2$. Notice that there exists a path $P_{I_1,I_d} = v(f_{I,1})v(f_{C,1})v(f_{I,2}) \ldots v(f_{I,d-1})v(f_{C,d-1})v(f_{I,d})$ between any pair of intersection faces $f_{I,1}$ and $f_{I,d}$, where $v(f_{I,i})$, with $i = 1, \ldots, d$, are the vertices of distinct intersection faces, and $v(f_{C,j})$, with $j = 1, \ldots, d-1$, correspond to k distinct chordal faces. This path P has a length of at most $2(\Delta - 1)$. Now notice that every face $f \subset S_1$ is either an intersection face or adjacent to an intersection face. Therefore, there exists a path like $P' = v(f)Pv(g)$ in the dual subgraph

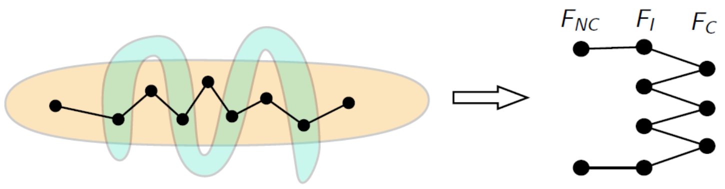


Figure 4 An example illustrating Definition 2.4 and Lemma 2.5.

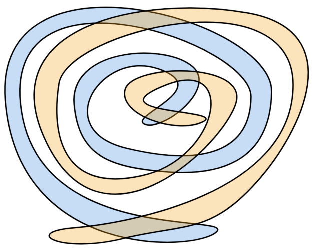


Figure 5 An almost tight example for Theorem 2.1.

$G^{\mathcal{A}_{S_1}}$ between any two vertices $v(f)$ and $v(g)$, $f, g \in S_1$, having a length of at most 2Δ. By Lemma 1.8, there also exists a simple $\{s,t\}$-curve in $\mathcal{A}$, for interior points $s, t \in S_1$, that crosses $\Gamma(S_1) \cup \Gamma(S_2)$ at most 2Δ times. ◀

▶ **Lemma 2.6.** *In an embedding of shapes S_1 and S_2 with interior points $s, t \in S_1 \cup S_2$, there exists a minimum crossing $\{s,t\}$-curve $C_{s,t}$ crossing $\Gamma(S_1) \cup \Gamma(S_2)$ at most 2Δ times.*

▶ **Lemma 2.7.** *In an embedding of shapes S_1 and S_2 with interior points $s, t \in \mathbb{R}^2 \backslash (S_1 \cup S_2)$, there exists a minimum crossing $\{s,t\}$-curve $C_{s,t}$ crossing $\Gamma(S_1) \cup \Gamma(S_2)$ at most $2\Delta + 2$ times.*

▶ **Lemma 2.8.** *In an embedding $\mathcal{A}$ of shapes S_1 and S_2 with interior points $s \in \mathbb{R}^2 \backslash (S_1 \cup S_2)$, $t \in S_1$, there exists a simple $\{s,t\}$-curve $C_{s,t}$ crossing $\Gamma(S_1) \cup \Gamma(S_2)$ at most $2\Delta + 1$ times.*

Proof of Theorem 2.1

Proof. Given two shapes in an embedding $\mathcal{A}$, arbitrarily label them S_1 and S_2, and denote $\Gamma(S_1) \cup \Gamma(S_2)$ by Γ. Consider a minimum crossing $\{s,t\}$-curve $C_{s,t}$. By Lemmas 2.5 and 2.6, for any $s, t \in S_1 \cup S_2$, $C_{s,t}$ crosses Γ at most 2Δ times. In Lemma 2.7, we show that $C_{s,t}$ crosses Γ at most $2\Delta + 2$ times for any $s, t \in \mathbb{R}^2 \backslash (S_1 \cup S_2)$. By Lemma 2.8, for any $s \in \mathbb{R}^2 \backslash (S_1 \cup S_2)$ and $t \in S_1$, $C_{s,t}$ crosses Γ at most $2\Delta + 1$ times. Note that this result also holds for a minimum crossing curve $C_{s',t'}$ with $s' = t$ and $t' = s$. Hence, for any interior points $s, t \in \mathbb{R}^2$ there exists a minimum crossing curve $C_{s,t}$ crossing $\Gamma(S_1) \cup \Gamma(S_2)$ at most $2\Delta + 2$ times. ◀

Figure 5 displays an almost tight example for Theorem 2.1 with a bound of $2\Delta - 2$.

3 Embeddings of n shapes

In this section, we show that there exists an upper bound for any minimum crossing $\{s,t\}$-curve in an embedding of n shapes. We first provide the definition of a maximal inclusion face and then show the main result in Theorem 3.2.

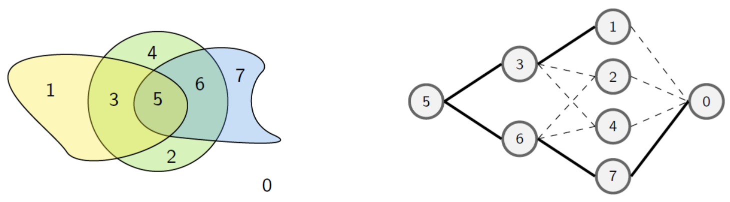


Figure 6 An example of a dual graph partitioned by height.

▶ **Definition 3.1.** A face f is a *maximal inclusion face* (in short, *MIF*) if every face adjacent to f has lower height. Additionally, define the complete set of $MIFs$ as $\mathcal{M}$.

▶ **Theorem 3.2.** *Given an embedding of a finite collection of shapes $\mathcal{S}_n$ in $\mathbb{R}^2$ and the set of Maximal Inclusion Faces $\mathcal{M}$, there exists a simple $\{s,t\}$-curve $C_{s,t}$ crossing $\Gamma(\mathcal{S}_n)$ at most $2|\mathcal{M}|\min\{n,\Delta+1\}$ times for any two interior points $s,t \in \mathbb{R}^2$.*

We use Lemmas 3.3 and 3.4 to show that Theorem 3.2 holds. In Lemma 3.3, we bound the diameter of the dual graph of an embedding of n shapes using $h_{\max}$ and $|\mathcal{M}|$. Lemma 3.4 then bounds $h_{\max}$.

▶ **Lemma 3.3.** *Given an embedding $\mathcal{A}$ of a set of shapes $\mathcal{S}_n$ and its dual graph $G^{\mathcal{A}}$, the diameter of $G^{\mathcal{A}}$ is upper bounded by $2h_{\max}|\mathcal{M}|$.*

Lemma 3.3 is a result of partitioning the vertices of the dual graph of an embedding of shapes by the height of the corresponding faces. An example of this type of partitioning is given in Figure 6.

▶ **Lemma 3.4.** *Given an embedding of shapes $S_1,..,S_n$, the maximum height of a face $h_{\max}$ is upper bounded by $\min\{n,\Delta+1\}$.*

Sketch of the proof of Theorem 3.2

Proof. As shown in Lemma 3.3, the diameter of the dual graph $G^{\mathcal{A}}$ of an embedding $\mathcal{A}$ of a set of shapes $\mathcal{S}_n$ is upper bounded by $2h_{max}|\mathcal{M}|$. By Lemma 3.4, h_{max} is upper bounded by $\min\{n,\Delta+1\}$. It follows that the diameter of the dual graph is bounded above by $2|\mathcal{M}|\min\{n,\Delta+1\}$, and therefore the length of any shortest path in $G^{\mathcal{A}}$ is bounded by the same number. If there exists a path between any two vertices in $G^{\mathcal{A}}$ with maximum length $2|\mathcal{M}|\min\{n,\Delta+1\}$, then by Lemma 1.8 there exists a minimum crossing $\{s,t\}$-curve $C_{s,t}$ in $\mathcal{A}$ crossing $\Gamma(\mathcal{S}_n)$ at most $2|\mathcal{M}|\min\{n,\Delta+1\}$ times. ◀

4 Conclusion and open questions

In this paper, we have shown that there exists a linear upper bound of $2\Delta+2$ for a minimum crossing $\{s,t\}$- curve in any embedding of 2 shapes in the plane. For embeddings of n shapes, we establish an upper bound of $2|\mathcal{M}|\min\{n,\Delta+1\}$.

Several questions remain. Firstly, what is the theoretic upper bound on the number of maximal inclusion faces ($|\mathcal{M}|$) in terms of n and Δ? The best example we have constructed so far encounters $\frac{1}{2}n\Delta$ maximal inclusion faces, as demonstrated by an example of nested shapes in Figure 7. We conjecture that this number is the right theoretic upper bound for $|\mathcal{M}|$. Consequently, what is the true hypothesis on the number of crossings of any minimum crossing $\{s,t\}$-curve in terms of n and Δ only?

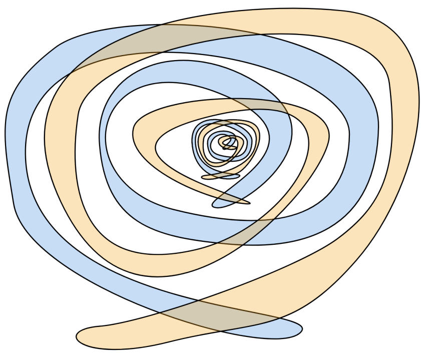


Figure 7 An almost tight example for Theorem 3.2.

References

1 Sergey Bereg and David Kirkpatrick. Approximating barrier resilience in wireless sensor networks. In *International Symposium on Algorithms and Experiments for Sensor Systems, Wireless Networks and Distributed Robotics*, pages 29–40. Springer, 2009.

2 Santosh Kumar, Ten H Lai, and Anish Arora. Barrier coverage with wireless sensors. In *Proceedings of the 11th annual international conference on Mobile computing and networking*, pages 284–298, 2005.

3 Terry A McKee and Fred R McMorris. *Topics in intersection graph theory*. SIAM, 1999.

Simultaneous Drawing of Layered Trees

Julia Katheder[1], Stephen G. Kobourov[2], Axel Kuckuk[1], Maximilian Pfister[1], and Johannes Zink[3]

1 **Wilhelm-Schickard-Institut für Informatik, Universität Tübingen, Tübingen, Germany**
`firstname.lastname@uni-tuebingen.de`

2 **Department of Computer Science, University of Arizona, Tucson, USA**
`lastname@cs.arizona.edu`

3 **Institut für Informatik, Universität Würzburg, Würzburg, Germany**
`lastname@informatik.uni-wuerzburg.de`

Abstract

We study the crossing minimization problem in a layered graph drawing of rooted trees whose leaves have a given fixed order on the first layer. The task is to permute the vertices on the other layers to minimize the number of crossings. While this problem is known to be NP-hard for multiple trees even on just two layers, we give a polynomial-time algorithm for the restricted case of two trees. On the other hand, when restricting the number of layers to three, we describe an XP-algorithm in the number of trees.

1 Introduction

Visualizing hierarchical structures as directed trees is essential for many applications, from software engineering [2] to medical ontologies [1] and phylogenetics in biology [12]. Phylogenetic trees in particular can serve as an example to illustrate the challenges of working with hierarchical structures, as they are inferred from large amounts of data using various computational methods [19] and need to be analyzed and checked for plausibility using domain knowledge [9]. From a human perspective, visual representations are needed for this purpose. Most available techniques focus on the visualization of a single tree [6]. However, certain tasks may require working with multiple, possible interrelated trees, such as the comparison of trees [9, 11] or analyzing ambiguous lineages [13]. Graham and Kennedy [6] provide a survey for drawing multipe trees in this context.

While there are many different visualization styles for trees (see an overview by Schulz [15]), directed node-link diagrams are the standard. The most common approach to visualize a directed graph as a node-link diagram is the layered drawing approach due to Sugiyama et al. [17]. After assigning vertices to layers, the next step is to permute the vertices on each layer such that the number of crossings is minimized, as crossings negatively affect the readability of a graph drawing [14, 18]. However, this problem turns out to be hard even when restricting the number of layers or the type of graphs. For example, if the number of layers is restricted to two, crossing minimization remains NP-hard for general graphs [5], even if the permutation on one layer is fixed [4], known as the *one sided crossing minimization* (OSCM) problem.

For the special case of trees on two layers, OSCM can be solved in polynomial time [7] and even if both layers are variable, the problem can be reduced to the minimum linear arrangement problem [16], which is polynomial-time solvable [3]. For arbitrarily many layers, the problem is still NP-hard even for trees [7], however, the obtained trees in the reduction [7] are not rooted (and we do not see an obvious way to adjust their construction).

39th European Workshop on Computational Geometry, Barcelona, Spain, March 29–31, 2023.
This is an extended abstract of a presentation given at EuroCG'23. 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 Contributions. We consider the crossing minimization problem for an n-vertex forest of k trees whose vertices are assigned to l layers such that all leaves are on the first layer and their order is fixed. We first describe a linear-time algorithm for the special case that all but one tree are paths (see Sec. 3). The general case where $k \in \mathcal{O}(n)$ is known to be NP-hard [10] even for $l = 2$ and trees of maximum degree 4. However, we show that the case $k = 2$ is polynomial-time solvable for arbitrary l using a dynamic program (see Sec. 4). Moreover, we describe an XP-algorithm in k modeling the solution space by a k-dimensional grid graph for $l = 3$ (see Sec. 5). We conclude with the elusive open case of $k \geq 3$ and $l \geq 4$ (see Sec. 6).

2 Preliminaries

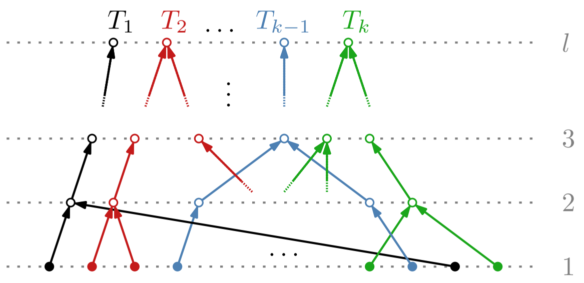


Figure 1 Upward drawing of k disjoint directed rooted trees $T_1, \dots T_k$ on l layers. As indicated by the filled vertices, the total order $\prec_1$ of the first layer is given, while $\prec_2, \dots, \prec_l$ need to be determined. In the following figures, we drop the arrow heads and assume an upward direction.

Let $\mathcal{F}$ be a forest of k disjoint rooted trees $T_1, \dots, T_k$ directed towards the roots such that all vertices except for the roots have outdegree 1. Let an assignment of vertices to l layers be given, such that each tree T_i is *drawn upward*, i.e., for any directed edge $(u, v) \in T_i$, we have that the layer of u, denoted by $L(u)$, is strictly less than $L(v)$. This implies that if $L(u) = 1$, u is a leaf of T_i. The other way around, we also require that for any leaf v, $L(v) = 1$ and, for any root r, $L(r) = l$. We further require that the total order $\prec_1$ of the first layer (i.e., the order of all leaves) is given as part of the input and defines a crossing-free embedding $\mathcal{E}_i$ with respect to each individual tree T_i, that is, there exists an ordering of the (non-leaf) vertices of T_i such that no two edges of T_i cross, see Fig. 1 for an illustration. We let $V_j(T_i)$ denote the (ordered) set of vertices of T_i on the j-th layer. If we refer specifically to the order of $V_j(T_i)$, we also use $\prec_j^i$ and, with respect to layer j, we also refer to $\prec_j^i$ as a *partial order*. Furthermore, we define the (unordered) set of all vertices on layer j as $V_j(\mathcal{F}) = V_j(T_1) \cup \dots \cup V_j(T_k)$.

The task is to find a total order $\prec_j$ of $V_j(\mathcal{F})$ for each $j \in \{2, \dots, l\}$ such that the total number of pairwise edge crossings implied by a corresponding straight-line realization of $\mathcal{F}$ is minimized. We restrict the notion of an upward drawing even further since we require that for any directed edge $(u, v) \in T_i$, we have that $L(u) + 1 = L(v)$. If our input does not fulfill this requirement, this can be achieved by subdividing edges which span several layers (as commonly done in the Sugiyama framework). Henceforth, we assume that n is the number of vertices after subdivision and $n_1, \dots, n_k$ is the number of vertices of $T_1, \dots, T_k$, respectively.

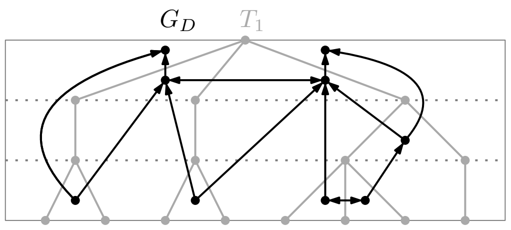
The following observation guarantees that in an optimal solution, we have that the partial order $\prec_j^i$ for a layer j restricted to the vertices of T_i is consistent with the embedding $\mathcal{E}_i$.

For statements marked with a $(\star)$, the proofs are available in the long version on arXiv [8].

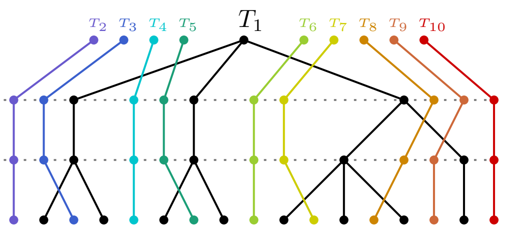
▶ Claim 2.1 $(\star)$. In a crossing-minimal drawing of $\mathcal{F}$, no edges of the same tree cross.

Thus, in an optimal solution, the total order of each layer respects the partial orders of the embeddings of the trees and we only need to combine these partial orders appropriately.

3 One Tree and Multiple Paths on Arbitrarily Many Layers



(a) The drawing of T_1 along with a bounding box induces the faces corresponding to the vertices of the upward dual graph G_D. Note that faces with an equal maximum layer of their respective vertices are connected by a bidirectional edge in G_D.



(b) Drawing with the minimum number of crossings of a forest $\mathcal{F} = \{T_1, \ldots, T_{10}\}$, while each T_i with $i \in \{2, \ldots, 10\}$ is a path. Note that, for T_5 and T_9, two paths in G_D have minimum value, yet the leftmost path is chosen as described in Sec. 3.

Figure 2 Drawing one tree and multiple paths with minimum crossings on arbitrary many layers.

In this section, we sketch out a simple linear-time algorithm to solve the problem for arbitrary k and l if the given forest is heavily restricted. For more details, see the extended description of the algorithm in the long version [8].

Let $\mathcal{F} = \{T_1, \ldots, T_k\}$ be a given forest such that T_i is a path for $i \in \{2, \ldots, k\}$. Recall that the total order $\prec_1$ allows for a crossing-free drawing D of the tree T_1. Consider the *weak dual graph* of the union of D and the axis-parallel bounding box of D, that is, the graph having a node for every inner face of this union and an edge if two faces of D are adjacent. From this dual graph, we obtain a directed auxiliary graph G_D where we add edge directions towards the faces terminating at a higher layer; see Fig. 2(a). Observe that G_D has two sinks. When assigning unit costs to each edge of G_D, the total costs of a directed path P in G_D correspond to the number of crossings a path T_i drawn upward along the faces of P induces.

By the given order of the vertices on the first layer, we know the node f of G_D corresponding to the face where a path T_i originates, that is, the face where the leaf of T_i lies. The optimal way to draw T_i upward is then derived by computing a shortest path P in G_D that starts at f and terminates at one of the two sinks of G_D (we consider both sinks as potential endpoints and take the one admitting the shorter path). In order to ensure that, for $i, j \in \{2, \ldots, k\}$, no two paths T_i and T_j cross, we always choose the leftmost minimum cost path as shown in Fig. 2(b).

▶ **Theorem 3.1** ($\star$). *Given an n-vertex forest $\mathcal{F} = \{T_1, \ldots, T_k\}$ on $l \in \mathcal{O}(n)$ layers whose leaves have a fixed order on the first layer such that T_1 is a directed rooted tree and, for each $i \in \{2, \ldots, k\}$, T_i is a directed path, we can compute a drawing of $\mathcal{F}$ with the minimum number of crossings in $\mathcal{O}(n)$ time.*

We remark that we can use the same algorithm in the case when T_1 is a planar graph and the paths $T_2, \ldots, T_k$ start and end on any layer (although the runtime grows to quadratic).

4 Two Trees on Arbitrarily Many Layers

In this section, we assume that we are given a forest $\mathcal{F} = \{T_1, T_2\}$ with embeddings $\mathcal{E}_1$ and $\mathcal{E}_2$. We fix the drawing of T_1 according to $\mathcal{E}_1$ and the only remaining task is to add the non-leaf vertices of T_2 in the order prescribed by $\mathcal{E}_2$ such that the number of crossings is minimized.

To this end, we describe a dynamic programming approach. In a complete drawing, we specify, for a vertex v of T_2, its position on its layer with respect to the closest neighbor b of T_1 to the right of v. We use the symbol $\varnothing$ if there is no such neighbor. Let P_v be the set of predecessors of v in T_2. If v lies on a layer $i \geq 3$, we define the number of crossings in an optimal partial solution for the drawing of the subtree of T_2 rooted at v and placed after b as

$$o[v,b] = \sum_{w \in P_v} \min_{b_w \in V_{i-1}(T_1) \cup \{\varnothing\}} \left(o[w, b_w] + c(w, b_w, v, b) \right) ,$$

where $c(w, b_w, v, b)$ is the number of crossings of the edge (w, v) with edges of T_1 if b_w is the right neighbor of w and b is the right neighbor of v. This value is solely dependent on the placement of the vertices v and w within the partial orders $\prec_i^1$ and $\prec_{i-1}^1$, respectively. As $\prec_1$ is fixed, the number of crossings in an optimal partial solution for the second layer is

$$o[v,b] = \sum_{w \in P_v} c(w, b_w, v, b) ,$$

where b_w is given by $\prec_1$. The minimum number of crossings is then $o^* = \min\{o[r_2, r_1], o[r_2, \emptyset]\}$, where r_1 and r_2 are the roots of T_1 and T_2, respectively. We return the drawing corresponding to o^*, i.e., we specify for each vertex v of T_2 its right neighbor b of T_1 when computing o^*. If we have equally good right neighbors for v, we take the leftmost one. Finally, for vertices of T_2 having the same right neighbor, we arrange them in the order given by $\mathcal{E}_2$.

Next, we prove the correctness of our dynamic program by the following claims.

▶ Claim 4.1 ($\star$). The computed drawing of T_2 is planar.

Proof sketch. Consider the bottommost layer where the vertices in the computed drawing of T_2 are not arranged as in the given planar embedding. With moving a vertex on this layer, we could decrease the number of crossings and the values of the dynamic program. ◀

Following from Claim 4.1, it only remains to consider crossings between T_1 and T_2.

▶ Claim 4.2 ($\star$). The number of crossings in the computed drawing is minimum.

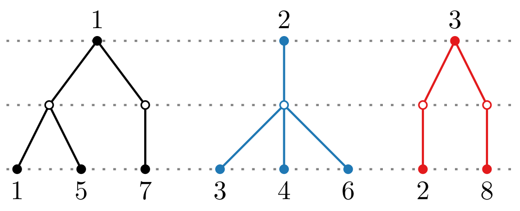
Proof sketch. In our dynamic program, all possible arrangements of the endpoints of an edge $e \in T_2$ with respect to T_1 are considered, which determine the number of crossings of e. ◀

▶ Claim 4.3 ($\star$). The running time of our algorithm is in $\mathcal{O}(n_1^2 \cdot n_2) \subseteq \mathcal{O}(n^3)$.

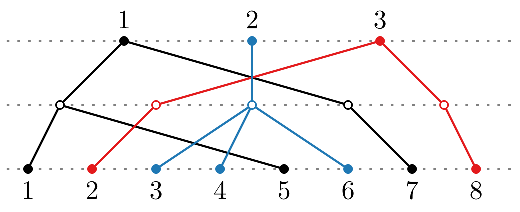
Proof sketch. For each edge of T_2, we pre-compute the number of crossings depending on its $\mathcal{O}(n_1^2)$ possible arrangements in $\mathcal{O}(n_1^2 n_2)$ time. We use them then to compute the $\mathcal{O}(n_1 n_2)$ entries $o[v, b]$, where every entry depends on $\mathcal{O}(n_1)$ other entries, in total $\mathcal{O}(n_1^2 n_2)$ time. ◀

▶ **Theorem 4.4.** *Let $\mathcal{F} = \{T_1, T_2\}$ be a n-vertex forest of two rooted trees, each of them given with a layered upward-planar embedding where all leaves are assigned to layer 1 and have a fixed order and both roots are assigned to layer k. We can compute a drawing of $\mathcal{F}$ with the minimum number of crossings in $\mathcal{O}(n^3)$ time.*

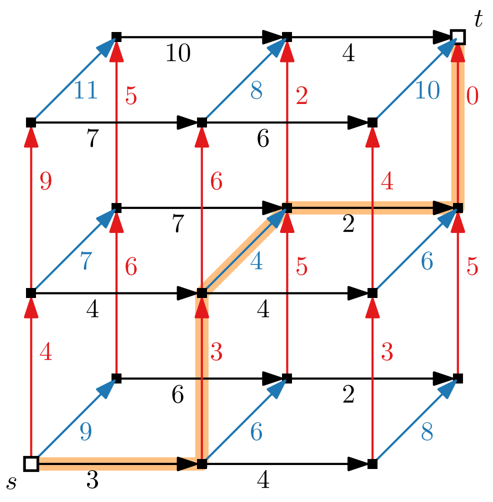
We remark that we can generalize this approach to a graph G_1 with a given layered upward-planar embedding and a tree T_2 because we can do the crossing calculation in the same way for a graph as for a tree. Though the root layer may contain an arbitrary number of vertices of G_1, the optimal position for the root of T_2 can be determined in linear time.



(a) Here, we are given three trees with a total order of the vertices on the first and the third layer.



(b) Drawing with the minimum number of crossings (six pairwise crossings). The total order of the vertices on the middle layer corresponds to the shortest st-path highlighted in orange on the right side.



(c) Grid graph H whose st-paths represent precisely the (allowed) total orders of the vertices on the second layer. The st-path highlighted in orange is the shortest path with weight 12.

Figure 3 Reducing the problem of finding a layered drawing of k trees on three layers with the minimum number of crossings to a shortest-path problem in a weighted k-dimensional grid graph.

5 Multiple Trees on Three Layers

We consider the case that we are given a forest $\mathcal{F} = \{T_1, \ldots, T_k\}$ of k trees spanning three layers each. Note that on the third layer there are only the k roots with $k!$ ways to arrange them. We consider each permutation of the roots individually. Consequently, we henceforth assume that both the total order $\prec_1$ of the leaves and the total order $\prec_3$ of the roots are fixed, and the only remaining task is to combine the k given partial orders $\langle \prec_2^1, \ldots, \prec_2^k \rangle$ of the vertices on the second layer to a total order $\prec_2$.

Let σ be a permutation of $V_2(\mathcal{F})$ respecting these partial orders. For $i \in \{1, \ldots, k\}$ and some vertex $v \in V_2(\mathcal{F})$, we denote the position (starting at 0) of v within the subsequence of σ consisting of the vertices $V_2(T_i) \cup \{v\}$ by $p^i_{v,\sigma}$. Note that, in a drawing using σ as $\prec_2$, we can charge every crossing to precisely two vertices of $V_2(\mathcal{F})$ as any crossing occurs between two edges that have two distinct endpoints on the second layer. Now we claim that for a vertex $v \in V_2(T_j)$, where $j \in \{1, \ldots, k\}$, the number of crossings charged to v with respect to σ depends only on $p^i_{v,\sigma}$ for each $i \in \{1, \ldots, k\} \setminus \{j\}$. Consider the (crossing-free) embedding of T_i together with all leaves and roots alone and then insert v and its incident edges onto the second layer. We denote the resulting number of crossings by $c^i_{v,p}$ where $p = p^i_{v,\sigma}$. The number $c_{v,\sigma}$ of crossings charged to v is then $c_{v,\sigma} = \sum_{i \in \{1,\ldots,k\} \setminus \{j\}} c^i_{v,p}$ and the total number of crossings $c_\sigma = \sum_{v \in V_2(\mathcal{F})} c_{v,\sigma}/2$.

▶ **Lemma 5.1** ($\star$). *For all combinations of $i \in \{1, \ldots, k\}$, $v \in V_2(\mathcal{F}) \setminus V_2(T_i)$, and $p \in \{0, \ldots, |V_2(T_i)|\}$, we can compute every value $c^i_{v,p}$ in a total of $\mathcal{O}(n^2)$ time.*

We now construct a weighted directed acyclic st-graph H (see Fig. 3) whose st-paths represent precisely all total orders of $V_2(\mathcal{F})$ that respect the partial orders $\langle \prec_2^1, \ldots, \prec_2^k \rangle$. Moreover, for an st-path π representing a total order σ of $V_2(\mathcal{F})$, the weight of π is twice the number of crossings induced by σ. We let H be the k-dimensional grid graph of side lengths $|V_2(T_1)| \times \cdots \times |V_2(T_k)|$ directed from one corner to an opposite corner, that is,

H has the node set $\{(x_1, \dots, x_k) \mid x_1 \in \{0, \dots, |V_2(T_1)|\}, \dots, x_k \in \{0, \dots, |V_2(T_k)|\}\}$ and there is a directed edge from $(x_1, \dots, x_k)$ to $(y_1, \dots, y_k)$ if $x_j + 1 = y_j$ for exactly one $j \in \{0, \dots, k\}$ and $x_i = y_i$ otherwise. Observe that, within H, $(0, \dots, 0)$ is the unique source and $(|V_2(T_1)|, \dots, |V_2(T_k)|)$ is the unique sink. We denote these nodes by s and t, respectively. We let the edge from $(x_1, \dots, x_k)$ to $(y_1, \dots, y_k)$ where $x_j + 1 = y_j$ represent (i) taking the y_j-th vertex of $V_2(T_j)$, which we denote as v, (ii) after having taken x_i vertices of $V_2(T_i)$ for each $i \in \{1, \dots, k\}$. Thus, we let the weight of this edge be the number of crossings charged to v in this situation, that is, the weight is $\sum_{i \in \{1,\dots,k\} \setminus \{j\}} c^i_{v,x_i}$.

Clearly, any st-path π in H has length n_2. If we traverse π, we can think of the second layer being empty when we start at s, and then, for each edge of π, we take the corresponding vertex of V_2 and add it to the second layer. Since the edge weights equal the number of crossings the corresponding vertex would induce in this situation, finding a shortest st-path in H means finding a crossing-minimal total order $\prec_2$. By construing H (using Lemma 5.1 to compute the edge weights) and searching for a shortest st-path, we obtain an XP-algorithm in k; see Theorem 5.2, which we formally prove in the full version [8].

▶ **Theorem 5.2** ($\star$). *Given an n-vertex forest $\mathcal{F}$ of k directed rooted trees on three layers whose roots are on the third layer and whose leaves have a fixed order on the first layer, we can compute a drawing of $\mathcal{F}$ with the minimum number of crossings in $\mathcal{O}(k^2 n^k)$ time.*

We remark that our result also holds for k planar graphs provided that on the third layer there are $\mathcal{O}(k)$ vertices. This includes the case that there are no vertices on the third layer and we have just two layers. For planar graphs and an arbitrary number of vertices on the third layer, our result holds if the total order of vertices on the third layer is fixed as well.

6 Open Problems

Can we solve the case $l = 3$ layers in FPT-time in the number k of trees? Is the case $k = 3$ and $l = 4$ polynomial-time solvable, and if so, for which k and l does it become hard?

References

1 Olivier Bodenreider. The unified medical language system (UMLS): Integrating biomedical terminology. *Nucleic Acids Research*, 32(suppl_1):267–270, 2004. doi:10.1093/nar/gkh061.

2 Sudarshan S. Chawathe, Anand Rajaraman, Hector Garcia-Molina, and Jennifer Widom. Change detection in hierarchically structured information. *ACM SIGMOD Record*, 25(2):493–504, 1996. doi:10.1145/235968.233366.

3 Fan R. K. Chung. On optimal linear arrangements of trees. *Computers and Mathematics with Applications*, 10(1):43–60, 1984. doi:10.1016/0898-1221(84)90085-3.

4 Peter Eades and Nicholas C. Wormald. Edge crossings in drawings of bipartite graphs. *Algorithmica*, 11(4):379–403, 1994. doi:10.1007/bf01187020.

5 Michael R. Garey and David S. Johnson. Crossing number is NP-complete. *SIAM Journal on Algebraic Discrete Methods*, 4(3):312–316, 1983. doi:10.1137/0604033.

6 Martin Graham and Jessie Kennedy. A survey of multiple tree visualisation. *Information Visualization*, 9(4):235–252, 2010. doi:10.1057/ivs.2009.29.

7 Martin Harrigan and Patrick Healy. k-level crossing minimization is NP-hard for trees. In Naoki Katoh and Amit Kumar, editors, *Proc. 5th International Workshop on Algorithms and Computation (WALCOM'11)*, volume 6552 of *Lecture Notes in Computer Science*, pages 70–76. Springer, 2011. doi:10.1007/978-3-642-19094-0_9.

8 Julia Katheder, Stephen G. Kobourov, Axel Kuckuk, Maximilian Pfister, and Johannes Zink. Simultaneous drawing of layered trees. arXiv preprint, 2023. `arXiv:2302.11952`.

9 Zipeng Liu, Shing Hei Zhan, and Tamara Munzner. Aggregated dendrograms for visual comparison between many phylogenetic trees. *IEEE transactions on visualization and computer graphics*, 26(9):2732–2747, 2019. `doi:10.1109/tvcg.2019.2898186`.

10 Xavier Muñoz, Walter Unger, and Imrich Vrto. One sided crossing minimization is NP-hard for sparse graphs. In Petra Mutzel, Michael Jünger, and Sebastian Leipert, editors, *Proc. 9th International Symposium on Graph Drawing (GD'01)*, volume 2265 of *Lecture Notes in Computer Science*, pages 115–123. Springer, 2001. `doi:10.1007/3-540-45848-4_10`.

11 Tamara Munzner, François Guimbretiere, Serdar Tasiran, Li Zhang, and Yunhong Zhou. Treejuxtaposer: scalable tree comparison using Focus+Context with guaranteed visibility. *ACM Transactions on Graphics*, 22(3):453–462, 2003. `doi:10.1145/1201775.882291`.

12 Georgios A Pavlopoulos, Theodoros G Soldatos, Adriano Barbosa-Silva, and Reinhard Schneider. A reference guide for tree analysis and visualization. *BioData mining*, 3(1):1–24, 2010. `doi:10.1186/1756-0381-3-1`.

13 Pere Puigbò, Yuri I Wolf, and Eugene V Koonin. Search for a 'tree of life' in the thicket of the phylogenetic forest. *Journal of Biology*, 8(6):1–17, 2009. `doi:10.1186/jbiol159`.

14 Helen C. Purchase, David A. Carrington, and Jo-Anne Allder. Empirical evaluation of aesthetics-based graph layout. *Empirical Software Engineering*, 7(3):233–255, 2002. `doi:10.1023/A:1016344215610`.

15 Hans-Jörg Schulz. Treevis.net: A tree visualization reference. *IEEE Computer Graphics and Applications*, 31(6):11–15, 2011. `doi:10.1109/mcg.2011.103`.

16 Farhad Shahrokhi, Ondrej Sýkora, László A. Székely, and Imrich Vrto. On bipartite drawings and the linear arrangement problem. *SIAM Journal on Computing*, 30(6):1773–1789, 2000. `doi:10.1137/S0097539797331671`.

17 Kozo Sugiyama, Shojiro Tagawa, and Mitsuhiko Toda. Methods for visual understanding of hierarchical system structures. *IEEE Transactions on Systems, Man, and Cybernetics*, 11(2):109–125, 1981. `doi:10.1109/TSMC.1981.4308636`.

18 Colin Ware, Helen C. Purchase, Linda Colpoys, and Matthew McGill. Cognitive measurements of graph aesthetics. *Information Visualization*, 1(2):103–110, 2002. `doi:10.1057/palgrave.ivs.9500013`.

19 Ziheng Yang and Bruce Rannala. Molecular phylogenetics: principles and practice. *Nature reviews genetics*, 13(5):303–314, 2012. `doi:10.1038/nrg3186`.

Classification of 2D bichromatic points with outliers[*]

Erwin Glazenburg, Marc van Kreveld, and Frank Staals

Utrecht University
`[e.p.glazenburg, m.j.vankreveld, f.staals]@uu.nl`

Abstract

Let $R \cup B$ be a set of n points in $\mathbb{R}^2$, and let k be an integer. We present an algorithm to compute a linear separator $\hat{s}$ that separates the "red" points R from the "blue" points B with at most k outliers while minimizing the distance to the farthest outlier in $O(nk + n \log n)$ time.

1 Introduction

Classification is a well known and well studied problem: given a "training" set of n data items with known classes, decide which class to assign to a new query item. A support vector machine (SVM) [3] is a popular classification method, in particular for binary classification problems in which there are only two classes: "red" and "blue". An SVM maps the input data items to points in some high dimensional $\mathbb{R}^d$, and constructs a hyperplane s that separates the "red" points R from the "blue" points B "as well as possible". Intuitively, it tries to minimize the distance from s to the set $X_s \subseteq R \cup B$ of points *misclassified* by s while maximizing the distance to the closest correctly classified points. A red point counts as misclassified if it lies strictly on the blue side of s, and vice versa.

Unfortunately, an SVM typically does not provide guarantees on the number of misclassified points, nor the distances between the points and the separating plane. Our main goal is to develop a separator that *does* provide such guarantees, both in terms of the number of misclassified points, and the distances to the points. We focus on the case where R and B are sets of points in the plane, as in Figure 1.

Aronov et al. [1] also considered computing optimal separators. They consider minimizing the following four error measures:

- $M_{\text{mis}}(s) = |X_s|$, the number of misclassified points.
- $M(s) = M_{\max}(s) = \max_{p \in X_s} \mathit{dist}(s, p)$, the maximum distance to a misclassified point.
- $M_{\text{sum}}(s) = \sum_{p \in X_s} \mathit{dist}(s, p)$, the sum of distances to misclassified points.
- $M_{\text{sum2}}(s) = \sum_{p \in X_s} \mathit{dist}(s, p)^2$, the sum of squared distances to misclassified points.

For n points in $\mathbb{R}^2$, the running times for computing an optimal separator vary from $O(n \log n)$ for the $M_{\max}$ measure to $O(n^2)$ for the M_{mis} and M_{sum2} measures. Har-Peled and Koltun [7] achieved similar results. Chan [2] showed that minimizing M_{mis} can even be done in an output sensitive $O((n + k^2) \log^2 n)$ time, where k is the smallest number of misclassifications possible.

[*] EG is supported by the Netherlands Organisation for Scientific Research (NWO) under project OCENW.M20.135.

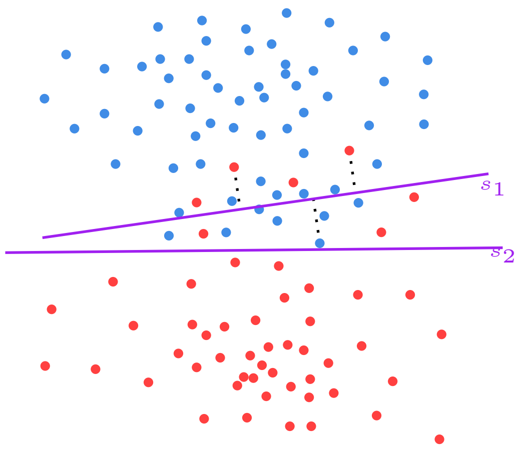


Figure 1 Some red and blue points with two separators s_1 and s_2. The distances from s_1 to its furthest misclassified points are shown as dashed lines.

Problem Definition. We propose a novel, natural measure that includes both the number of misclassified points and the distance of the misclassified points to the separating line. Given a number k we define a separator s to be *valid* if it misclassifies at most k points, i.e. $M_{\text{mis}}(s) \leq k$. We then wish to find a valid separator $\hat{s}_{\text{opt}}$ that minimizes the distance to the farthest misclassified point, that is, $\hat{s}_{\text{opt}} = \text{argmin}_{s \in V} M_{\max}(s)$, where V is the set of all valid separators. We refer to this problem as the *k-mis MinMax* problem.

Results and Organization. As a warmup, we first consider minimizing the maximum distance to misclassified points without any constraint on the number of misclassified points. In Section 2, we prove some useful properties of solutions to this *MinMax* problem. In Section 3 we build on these insights and characterize solutions to the k-mis MinMax problem. In Section 4 we present an $O(nk + n \log n)$ time algorithm to find an optimal separator that misclassifies at most k points. Some proofs are omitted due to a lack of space; these will appear in the full version of this article.

Notation and definitions. We use the standard point-line duality that maps any point $p = (p_x, p_y)$ in the primal plane to the line $p^* : y = p_x x - p_y$ in the dual plane, and any line $\ell : y = mx + c$ in the primal plane into the point $(m, -c)$ in the dual plane.

We describe an algorithm for finding an optimal separator $\hat{s}$ that classifies points above as blue, and points below as red; in the dual this means lines above the separating point $\hat{s}^*$ are classified as red and lines below as blue. We can then repeat the algorithm to find the best separator that classifies the other way around, and finally output the best of the two.

For an arrangement of lines $\mathcal{A}$, the $\leq k$*-level* of $\mathcal{A}$ is the set of points for which there are at most k lines below it, and the k*-level* is the boundary of this set. Note that a k-level lies exactly on existing lines in $\mathcal{A}$. The $\geq k$-level is defined similarly. Although these terms refer to a region in the plane, with a slight abuse of notation we will use them to refer to the part of the arrangement that lies in, or on the boundary of, this region. The 0-level of $\mathcal{A}$ is also called the *lower envelope* $\mathcal{L}(\mathcal{A})$, and the n-level is also called the *upper envelope* $\mathcal{U}(\mathcal{A})$.

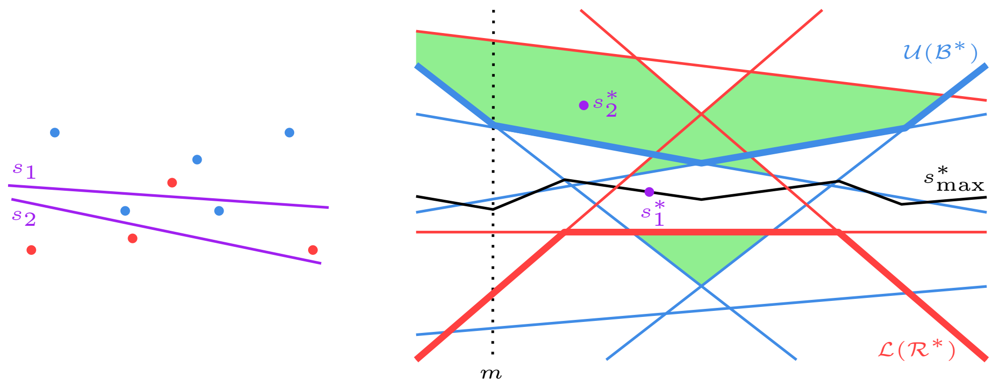


Figure 2 Some primal points and separators (left) with their duals (right). Valid cells for $k = 2$ are green.

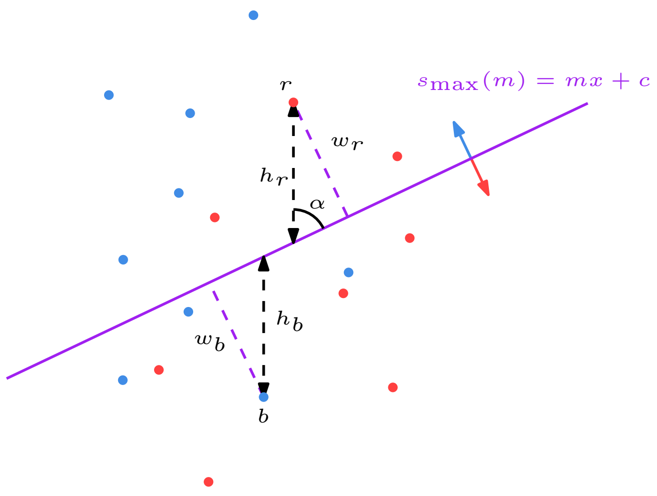


Figure 3 The optimal separating line $s_{\max}(m)$ for a given slope m, with extremal points r and b.

2 Properties of the MinMax problem

For the MinMax problem we wish to compute $\hat{s}_{\max} = \operatorname{argmin}_s M(s)$, a separator with minimum distance to the farthest misclassified point.

2.1 Fixed slope

First we consider the problem for lines with a fixed slope m; note that the dual points of these lines all lie on the vertical line $x = m$ in the dual plane. For slope m there is some optimal intercept c such that the resulting separating line $s_{\max}(m) = mx + c$ minimizes the error $M(s_{\max}(m))$. The error for any separator with slope m is defined by a red point r and/or a blue point b, the extremal points in that direction, as in Figure 3. Changing the intercept will not change which points are extremal, only the distances to these extremal points. Separator $s_{\max}(m)$ minimises distances to r and b, so it is in their middle with equal distance to both.

▶ **Lemma 2.1.** *In the dual plane, for a fixed slope m the optimal MinMax separator $s^*_{\max}(m)$ is vertically in the middle of $\mathcal{L}(R^*)$ and $\mathcal{U}(B^*)$ at $x = m$.*

2.2 Any slope

Let $s^*_{\max} = \{s^*_{\max}(m) \mid m \in \mathbb{R}\}$ be the curve in the dual plane representing the optimal separator for the MinMax problem for any given slope, shown in black in Figure 2. By Lemma 2.1 $s^*_{\max}$ is in the middle of the red and blue envelopes $\mathcal{U}(B^*)$ and $\mathcal{L}(R^*)$ for every slope. Since the lower or upper envelope of a set of n lines is a polygonal chain with $O(n)$ vertices, $s^*_{\max}$ is also a polygonal chain with $O(n)$ vertices.

▶ **Lemma 2.2.** *The global optimum $\hat{s}^*_{\max}$ for the MinMax problem lies on a vertex of $s^*_{\max}$.*

3 Properties of the k-mis MinMax problem

Recall we are given an integer $k \leq n$, and a separator s is valid if $M_{\text{mis}}(s) \leq k$. We want to compute an optimal separator $\hat{s}_{\text{opt}}$, which is a valid separator with minimum $M(\hat{s}_{\text{opt}})$.

The MinMax problem is a special case of this problem with $k = n$. Note that if an optimal solution to the MinMax problem $\hat{s}_{\max}$ is valid for a given k, it is also an optimal solution for the k-mis MinMax problem. However $\hat{s}_{\max}$ is not always valid, for example in Figure 2 where the whole curve $s^*_{\max}$ is invalid.

Any perfect separator $s_{perfect}$ with $M_{\text{mis}}(s_{perfect}) = 0$ will have error $M(s_{perfect}) = 0$ as well, and therefor be optimal. We can check if any such separators exist in $O(n \log n)$ time by calculating the envelopes $\mathcal{U}(B^*)$ and $\mathcal{L}(R^*)$ and intersecting them: all points above $\mathcal{U}(B^*)$ and below $\mathcal{L}(R^*)$ are perfect separators. From now on we will assume there are no perfect separators.

Let $\mathcal{A}$ be the arrangement of the dual lines in $R^* \cup B^*$. For any two separators s^*_1 and s^*_2 in the same face of $\mathcal{A}$, $M_{\text{mis}}(s^*_1) = M_{\text{mis}}(s^*_2)$. Let faces containing valid separators be *valid* faces. Note that points on the boundary of a valid face are also valid, since in the primal plane points on the separator are counted as correctly classified.

Let $\mathcal{R}^*_k$ be the $\leq k$-level of R^*, let $\mathcal{B}^*_k$ be the $\geq (n-k)$-level of B^*. Then all valid points lie inside $\mathcal{R}^*_k \cap \mathcal{B}^*_k$, since any point s^* above $\mathcal{R}^*_k$ has more than k red lines below it, so $M_{\text{mis}}(s^*) > k$ as well, making it invalid (similarly for $\mathcal{B}^*_k$).

This means we are not interested in all of $\mathcal{A}$, but only in the overlay $\mathcal{I}$ of $\mathcal{R}^*_k$ and $\mathcal{B}^*_k$. Both $\mathcal{R}^*_k$ and $\mathcal{B}^*_k$ have complexity $O(nk)$ [5], but we can not yet draw conclusions about the complexity of $\mathcal{I}$ since intersections between $\mathcal{R}^*_k$ and $\mathcal{B}^*_k$ add additional complexity. Following an idea of Chan [2] we can form $O(k)$ concave chains such that every edge in $\mathcal{R}^*_k$ is covered by an edge of at least one chain, the *concave chain decomposition* of $\mathcal{R}^*_k$. Similarly we can form $O(k)$ convex chains that cover $\mathcal{B}^*_k$. Since a concave chain and a convex chain can intersect at only two points, there are $O(k^2)$ intersections between the concave chains and the convex chains, and therefore $O(k^2)$ intersections between $\mathcal{R}^*_k$ and $\mathcal{B}^*_k$. Therefore, the overlay $\mathcal{I}$ has complexity $O(nk + k^2) = O(nk)$.

In our algorithm we will want to divide $\mathcal{I}$ into vertical slabs divided by vertical lines at each vertex of $s^*_{\max}$, as shown in Figure 4. Let $\mathcal{S}$ be the resulting arrangement. At any given slope there are k red and k blue lines in $\mathcal{I}$, by definition of $\leq k$-levels, so each vertical line has only $O(k)$ intersections in $\mathcal{I}$. Since $s^*_{\max}$ has $O(n)$ vertices we add $O(n)$ vertical lines, adding a total of $O(nk)$ vertices to $\mathcal{S}$. This means $\mathcal{S}$ still has complexity $O(nk)$.

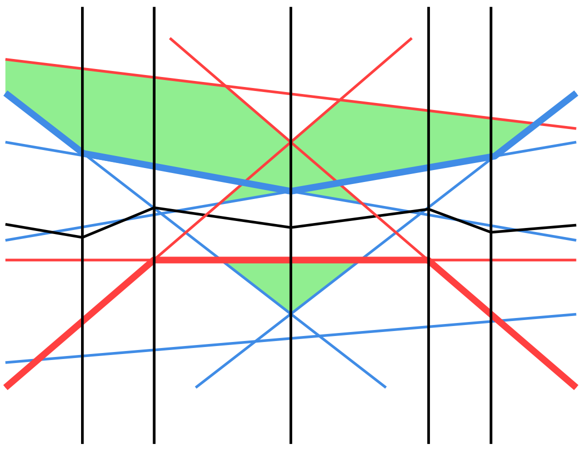


Figure 4 An arrangement with vertical slabs $\mathcal{S}$. Every segment of $s^*_{\max}$ defines one slab.

3.1 Properties of the optimum

▶ **Lemma 3.1.** *For a fixed slope m, an optimal separator $s_{\text{opt}}(m)$ with that slope is a valid point $s^*_{\text{opt}}(m)$ in the dual plane with the smallest vertical distance to $s^*_{\max}(m)$.*

If there are no valid points in the dual plane at $x = m$, then all separators with slope m misclassify more than k points, so $s^*_{\text{opt}}(m)$ does not exist.

Let $s^*_{\text{opt}} = \{s^*_{\text{opt}}(m) \mid m \in \mathbb{R}\}$ be the set of points in the dual plane representing an optimal separator for the k-mis MinMax problem for any slope, shown in purple in Figure 5. s^*_{opt} forms a set of polylines, is non-continuous and need not be defined for every slope.

A global optimum $\hat{s}^*_{\text{opt}}$ clearly lies on s^*_{opt}. We can further characterize this optimum:

▶ **Lemma 3.2.** *An optimal dual point $\hat{s}^*_{\text{opt}}$ is one of the following:*

a. *A vertex of a valid cell, vertically closest to $s^*_{\max}$.*
b. *A (valid) vertex of $s^*_{\max}$.*
c. *The valid point vertically closest to a vertex of $s^*_{\max}$.*
d. *The intersection of a valid edge with $s^*_{\max}$.*

Proof. Intuition: in the primal plane, the optimal separating line has to be bounded by at least three points. These can either be extremal points that we want the separator to be as close to as possible, or points that the separator is not allowed to cross because that would make it invalid. The four cases are shown in Figure 5.

Proof by contradiction. Let s'^* be an optimal dual point that is not any of the four cases **a, b, c, d**. Then it has to be one of the following five cases, for each of which we show that we can change s'^* slightly to decrease $M(s'^*)$, meaning s'^* was not optimal, a contradiction.

1. s'^* is not inside a valid face; clearly this is not possible.
2. s'^* is in the interior of a valid face and not on $s^*_{\max}$. Then moving vertically towards $s^*_{\max}$ will decrease $M(s'^*)$ by Lemma 3.1.
3. s'^* lies in the interior of a valid face and in the interior of an edge of $s^*_{\max}$. Then moving to one of the two adjacent vertices will decrease $M(s'^*)$, by the same proof of Lemma 2.2.
4. s'^* lies in the interior of a valid edge e^*, not on $s^*_{\max}$ and not above/below a $s^*_{\max}$ vertex. This means primal line s' goes through point e, as in Figure 6. It has a single extremal point (if multiple points of the same color would be equally far away we would be below

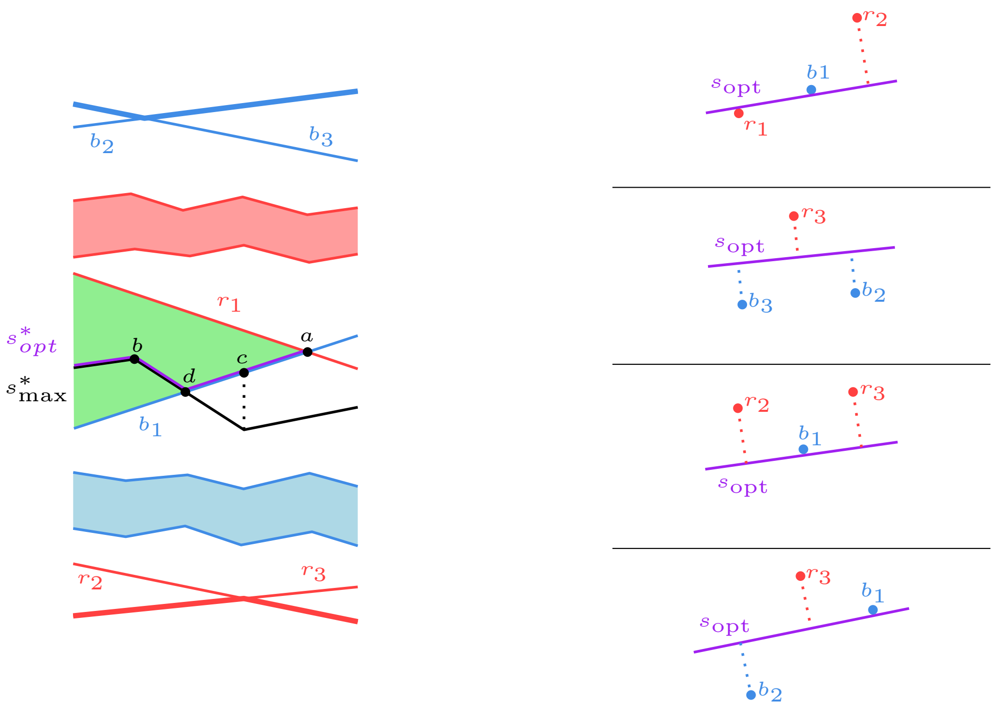


Figure 5 Left: the cases **a, b, c, d** for s^*_{opt} in the dual plane; the red/blue shaded regions represent some correctly classified lines. Right: from top to bottom, the cases **a, b, c, d** for s_{opt} in the primal plane.

a s^*_{max} vertex, and if a blue and red point would be equally far away we would be on s^*_{max}). W.l.o.g. let the extremal point be a blue point b, and let c_b be a circle centered at b tangent to s'. We can rotate s' either clockwise or counterclockwise around e (clockwise in Figure 6) to make it intersect c_b, decreasing the distance to b and decreasing $M(s'^*)$.

5. s'^* lies in the interior of a valid edge e^* above/below a s^*_{max} vertex, or on a valid vertex, but there is another point s_2 with the same slope that is closer to s^*_{opt}. Then s_2 is better by Lemma 3.1.

Now the only remaining cases are cases **a, b, c, d**, proving the lemma. ◀

▶ **Lemma 3.3.** *There are $O(nk)$ points of type* **a** *and* **d**, *and $O(n)$ points of type* **b** *and* **c**.

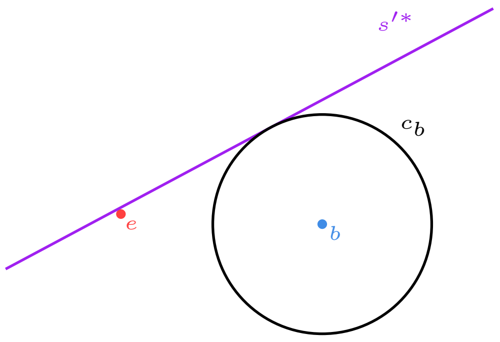


Figure 6 Separator s'^* going through point e with extremal point b.

4 Solving the k-mis MinMax problem

We now have all the ingredients to solve the k-mis MinMax problem in $O(nk + n\log n)$ time.

1. Construct $\mathcal{L}(R^*)$ and $\mathcal{U}(B^*)$. This takes $O(n\log n)$ time with a convex hull algorithm [4].
2. Construct $s^*_{\max}$ from $\mathcal{L}(R^*)$ and $\mathcal{U}(B^*)$. This takes $O(n)$ time by simultaneously scanning $\mathcal{L}(R^*)$ and $\mathcal{U}(B^*)$ from left to right.
3. Construct $\mathcal{R}^*_k$ and $\mathcal{B}^*_k$. This can be done in $O(nk + n\log n)$ time by Everett et al. [5].
4. Overlay $\mathcal{R}^*_k$ and $\mathcal{B}^*_k$ to get $\mathcal{I}$. Almost all faces in $\mathcal{R}^*_k$ are convex, since it is an arrangement of lines, except for the upper unbounded face which has the k-level as boundary. This face can be trivially triangulated, making the arrangement convex. The same holds for $\mathcal{B}^*_k$. We can then use Finke and Hinrichs' algorithm [6] to overlay two convex arrangements in time linear in both input and output size, which is $O(nk)$ in our case.
5. Walk through the faces of arrangement $\mathcal{I}$ in depth-first search order, maintaining the number of misclassifications per face, and storing it for each face. For two neighboring faces F_1 and F_2, $|M_{\text{mis}}(F_1) - M_{\text{mis}}(F_2)| = 1$, since we cross only a single line. This means we can maintain the number of misclassifications in constant time per step, so going through the whole arrangement takes $O(nk)$ time.
6. Insert vertical lines into $\mathcal{I}$ at each vertex of $s^*_{\max}$ to get $\mathcal{S}$. This can for example be done in $O(nk)$ time using Finke and Hinrichs algorithm [6].
7. Insert curve $s^*_{\max}$ into $\mathcal{S}$ in $O(nk)$ time, again using Finke and Hinrichs algorithm [6]. Let $\mathcal{S}'$ be the resulting arrangement.
8. Iterate through all vertices of $\mathcal{S}'$ in $O(nk)$ time. For vertex s^*, if $M_{\text{mis}}(s^*) \le k$, calculate its error $M(s^*)$. Maintain the vertex with the smallest error.

Step 8 in the above algorithm will iterate through all type **a** points because they are all vertices of $\mathcal{I}$, all type **b** points because these vertices were inserted when inserting $s^*_{\max}$ into $\mathcal{S}$ to get $\mathcal{S}'$, all type **c** vertices because they were introduced when inserting the vertical lines into $\mathcal{I}$ to get $\mathcal{S}$, and all type **d** vertices because they were introduced when inserting $s^*_{\max}$ into $\mathcal{S}$ to get $\mathcal{S}'$. By Lemma 3.2, we have found the optimal separator.

▶ **Theorem 4.1.** *Let R, B be two sets of red and blue points respectively, with $|R \cup B| = n$, and let k be an integer. We can compute an optimal separating line that misclassifies at most k points in $O(nk + n\log n)$ time.*

References

1 Boris Aronov, Delia Garijo, Yurai Núñez Rodríguez, David Rappaport, Carlos Seara, and Jorge Urrutia. Minimizing the error of linear separators on linearly inseparable data. *Discret. Appl. Math.*, 160(10-11):1441–1452, 2012. doi:10.1016/j.dam.2012.03.009.

2 Timothy M. Chan. Low-dimensional linear programming with violations. *SIAM J. Comput.*, 34(4):879–893, 2005. doi:10.1137/S0097539703439404.

3 Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Mach. Learn.*, 20(3):273–297, 1995. doi:10.1007/BF00994018.

4 Mark de Berg, Otfried Cheong, Marc J. van Kreveld, and Mark H. Overmars. *Computational geometry: algorithms and applications, 3rd Edition.* Springer, 2008. URL: https://www.worldcat.org/oclc/227584184.

5 Hazel Everett, Jean-Marc Robert, and Marc J. van Kreveld. An optimal algorithm for the ($\leq$ k)-levels, with applications to separation and transversal problems. *Int. J. Comput. Geom. Appl.*, 6(3):247–261, 1996.

6 Ulrich Finke and Klaus H. Hinrichs. Overlaying simply connected planar subdivisions in linear time. In Jack Snoeyink, editor, *Proceedings of the Eleventh Annual Symposium on Computational Geometry*, pages 119–126. ACM, 1995. doi:10.1145/220279.220292.

7 Sariel Har-Peled and Vladlen Koltun. Separability with outliers. In *Algorithms and Computation, 16th International Symposium, ISAAC 2005, Sanya, Hainan, China, December 19-21, 2005, Proceedings*, volume 3827 of *Lecture Notes in Computer Science*, pages 28–39. Springer, 2005. doi:10.1007/11602613_5.

Gradual Simplification of Polylines

Stefan Funke[1] and Sabine Storandt[2]

1 **University of Stuttgart**
`funke@fmi.uni-stuttgart.de`

2 **University of Konstanz**
`storandt@inf.uni-konstanz.de`

Abstract

We propose the gradual polyline simplification problem, where a fine-grained succession of consistent simplifications of a given input polyline must be computed. We discuss different problem variants and present exact and approximate algorithms to solve them efficiently.

1 Introduction

Polyline simplification is the process of reducing the complexity of linear structures while ensuring that the output still resembles the main features of the input. There is a wide range of applications for polyline simplification, including data compression, noise reduction in trajectories, or visualization of linear structures on maps.

Formally, a polyline $L = p_1, p_2, \ldots, p_n$ is defined as a sequence of points and the induced straight line segments between consecutive points. In the polyline simplification problem, the input is a polyline, a distance measure d_X, and a threshold $\varepsilon > 0$. A line segment s_{ij} (also called shortcut) between polyline points p_i and $p_{j>i}$ is called *valid* if $d_X(s_{ij}, L[i,j]) \leq \varepsilon$, where $L[i,j]$ refers to the subpolyline of L from p_i to p_j. We also refer to $d_X(s_{ij}, L[i,j])$ as the shortcut error of s_{ij}. The goal of polyline simplification is to compute a minimum-sized path from p_1 to p_n that only uses valid shortcuts. The endpoints of these shortcuts define the simplified polyline $L' \subseteq L$. Typical similarity measures for polylines are the Hausdorff distance d_H and the Fréchet distance d_F. The polyline simplification problem can be solved optimally in $\mathcal{O}(n^2)$ for d_H [5] and in $\mathcal{O}(n^2 \log n)$ for d_F [6].

In [3], the problem of progressive simplification was introduced, motivated by the application of visualizing polylines on different levels of granularity e.g. in zoomable digital maps. Here, given a sequence of distance threshold values $\varepsilon_1 < \varepsilon_2 < \cdots < \varepsilon_k$, one needs to find a valid polyline simplification for ε_1, followed by a valid polyline simplification for ε_2, and so on, with the constraint that the simplification for ε_i contains of a subset of the points chosen for the simplification for ε_{i-1} for all $i > 1$ (also referred to as consistency). The optimization goal is to find a sequence of simplifications with the smallest number of shortcuts accumulated over all simplification levels. It was shown in [3] that the problem can be solved to optimality in $\mathcal{O}(n^3 k)$ for both d_H and d_F. Furthermore, a continuous version was discussed which can be solved in $\mathcal{O}(n^5)$.

Specifying a reasonable sequence of distance thresholds for progressive simplification is a non-trivial task. For example, it could easily happen that several consecutive ε values in the sequence induce the same simplification; thus only adding computational complexity but no further visual benefits. Furthermore, adding one ε to the sequence might impact all simplifications, as it might be beneficial to retain certain points in earlier simplifications to get smaller simplifications in later stages. The continuous version gets rid of the issue of needing to specify $\varepsilon_1 < \varepsilon_2 < \cdots < \varepsilon_k$ but at the cost of a significant increase in running time.

39th European Workshop on Computational Geometry, Barcelona, Spain, March 29–31, 2023.
This is an extended abstract of a presentation given at EuroCG'23. 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.

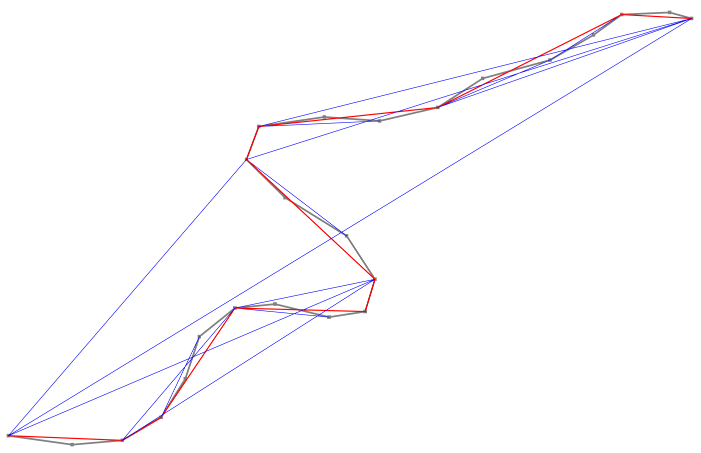


Figure 1 Example polyline (grey) and gradual simplification (blue and red shortcuts) with the optimal ordering for sum-error. The red polyline corresponds to the respective simplification of the input polyline after half the points have been shortcutted.

In this paper, we will introduce and discuss the notion of gradual line simplification as an alternative to progressive simplification.

2 The Gradual Line Simplification (GLS) Problem

While in progressive line simplification (PLS) the sequence of distance thresholds is provided as input and the goal is to find the smallest consistent simplification sequence that adheres to these thresholds, GLS takes an orthogonal view.

▶ **Definition 2.1** (Gradual Line Simplification)**.** Given a polyline $L = p_1, p_2, \ldots, p_n$, a gradual simplification corresponds to an ordering of the inner points $p_2, \ldots, p_{n-1}$ which are then shortcutted in the respective order. The goal is to find a contraction order that minimizes the maximum shortcut error (**max-error**) or the total sum of shortcut errors (**sum-error**).

Figure 1 illustrates the outcome of gradual line simplification on a small example instance. Consistency is not an issue for GLS as every contraction order yields a consistent simplification sequence. And compared to PLS, the produced simplification sequence is more fine-grained. However, GLS deviates from the wide-spread approach for polyline simplification where the distance threshold ε is provided. In this scenario, the validity of a shortcut is determined by deciding whether its error is smaller or larger than ε. In GLS, we need to compute the actual shortcut errors which is computationally more demanding. Nevertheless, we will show that contraction orders with high quality outputs can be computed efficiently.

3 Algorithms for GLS

We will propose exact and approximate algorithms to find good contraction orders for max-error GLS and sum-error GLS. The algorithms heavily rely on access to shortcut errors $d_X(s_{ij}, L[i,j])$. Under d_H, such errors can be computed in $\mathcal{O}(j-i) \subseteq O(n)$. Under d_F, using the algorithms by Alt and Godau [2], the error can be computed in $\mathcal{O}(n^2)$, or $\mathcal{O}(n \log n)$ if parametric search is applied. Recently, a data structure (FDS) proposed by Buchin et al. [4] allows to preprocess a given polyline in time and space $\mathcal{O}(n \cdot k^{3+\delta} + n^2)$ for $\delta > 0$ and then answer shortcut error queries in time $\mathcal{O}(\frac{n}{k} \log^2 n + \log^4 n)$ for some choice of $k \in [1, n]$.

3.1 Exact Algorithms

We first show that both the max-error and the sum-error problem can be solved to optimality in polytime. In particular, we design efficient dynamic programs (DP) that require $\mathcal{O}(n^3)$ time and $\mathcal{O}(n^2)$ space.

Let $\varepsilon(s_{ij})$ denote the shortcut error of s_{ij} with respect to $L[i,j]$. Further, let $\varepsilon_{\max}(s_{ij})$ denote the max-error of an optimal solution of the gradual simplification problem restricted to $L[i,j]$ (which automatically implies that s_{ij} is part of the solution). Similarly, $\varepsilon_{sum}(s_{ij})$ denotes the optimal sum-error of gradual polyline simplification for $L[i,j]$. Let p_k be the last point shortcutted before p_i and p_j, i.e. the point whose shortcutting resulted in the insertion of s_{ij}. Then the shortcuts (or original segments) s_{ik} and s_{kj} are part of the solution as well, and we have $\varepsilon_{max}(s_{ij}) = \max\{\varepsilon_{\max}(s_{ik}), \varepsilon_{\max}(s_{kj}), \varepsilon(s_{ij})\}$ and $\varepsilon_{sum}(s_{ij}) = \varepsilon_{sum}(s_{ik}) + \varepsilon_{sum}(s_{kj}) + \varepsilon(s_{ij})$. Based on these formulas, we can construct the solution set recursively starting with s_{1n}, which always has to be contained in any solution. However, we don't know the value of k if we go top-down. But this can easily be overcome by iterating over all possible k with $i < k < j$ and picking the minimum resulting max- or -sum-error. We can store already computed solutions for each s_{ij} in a look-up table to avoid redundant computations. This results in the following dynamic program: We allocate an $n \times n$ table and initially set all entries to 0. In cell c_{ij} with $i < j$, we store $\varepsilon_{\max}(s_{ij})$ or $\varepsilon_{sum}(s_{ij})$, respectively. The table cells are filled by using the above formulas. As we always need access to all shortcuts of smaller hop length to get the correct results, we consider the cells sorted increasingly by $j - i$.

▶ **Theorem 3.1.** *The DP approach solves max-error GLS and sum-error GLS in time $\mathcal{O}(n^3)$ under d_H and d_F, respectively, using quadratic space.*

Proof. The created table clearly has a space consumption of $\Theta(n^2)$. The running time is determined by the time needed to fill those $\Theta(n^2)$ cells. To get the correct cell value for c_{ij}, we first need to compute $\varepsilon(s_{ij})$ and then iterate over all values k between i and j and check cells c_{ik} and c_{kj}. The latter part can be accomplished in constant time per considered cell and hence takes $\mathcal{O}(n)$ in total. The computation of $\varepsilon(s_{ij})$ depends on the distance measure. For d_H, it takes time $\mathcal{O}(n)$. For the Frèchet distance, as discussed above, this would take $\mathcal{O}(n \log n)$ when using the parametric search technique. However, based on the FDS by Buchin et al. with a choice of $k = \sqrt{n}$, preprocessing the polyline and determining all potential shortcut errors can be accomplished in $o(n^3)$. Thus, we spend on average $O(n)$ time per cell for both d_H and d_F, which amounts to an overall running time of $\mathcal{O}(n^3)$. ◀

3.2 Approximation Algorithms

The cubic running time and the quadratic space consumption of the exact algorithm might be prohibitive in practice, especially when dealing with large inputs. Therefore, we next investigate approximation algorithms with better performance.

We first observe that the max-error problem variant is very easy to approximate for d_F based on a well-known lemma by Agarwal [1], rephrased below in our terminology:

▶ **Lemma 3.2.** *Given a polyline L, consider a shortcut s_{ij} for the subpolyline $L[i,j]$ with error ε. Then for any shortcut s_{ab} with $i \leq a < b \leq j$, its error under d_F is bounded by $d_F(s_{ab}, L[a,b]) \leq 2\varepsilon$.*

According to the lemma, the error of shortcut s_{1n} – which has to be contained in all gradual simplifications – is at least half the error of any other possible shortcut s_{ab} with $1 \leq a < b \leq n$. Thus, we get the following corollary.

▶ **Corollary 3.3.** *Any ordering is a 2-approximation for max-error GLS under d_F.*

Now let us turn to the sum-error problem variant. Note that the sum-error variant coincides with minimizing the average shortcut error, as the total number of shortcuts is $n-2$ for any ordering. In the following we present and analyze a greedy algorithm for sum-error GLS. The idea is to simply always choose the point next whose shortcutting will result in the currently smallest shortcut error.

▶ **Theorem 3.4.** *The greedy ordering is a 4-approximation for sum-error GLS under d_F.*

Proof. Let L be a polyline of size n. Let $S_1, \dots, S_{n-2}$ be the shortcuts created by the greedy algorithm in their insertion order and let $\pi_1, \dots, \pi_{n-2}$ denote the the points in L according to their contraction order. Further, let $S_1^*, \dots, S_{n-2}^*$ be the shortcuts inserted based on an optimal point ordering. We use L_i or L_j^* to refer to the subpolylines shortcutted by S_i or S_j^*, respectively. Furthermore we use $\varepsilon(S)$ to denote the shortcut error of a shortcut S.

We construct an assignment of shortcuts inserted by the greedy algorithm to optimal shortcuts. In particular, we assign shortcut S_i to S_j^* if the following two conditions are met:

- At the moment before π_i is shortcutted by the greedy algorithm, there are at least three points in L_j^* that are not yet shortcutted, including π_i.
- Index j is the smallest index in the optimal ordering for which the above property holds.

Note that the assignment is well-defined, as we have $S_{n-2}^* = s_{1n}$ and the first condition is always true for S_{n-2}^* as we never shortcut its endpoints.

We now show that if S_i is assigned to S_j^* it holds $\varepsilon(S_i) \leq 2\varepsilon(S_j^*)$. This applies because if at least three points of L_j^* are not shortcutted before the shortcutting of point π_i, we know that shortcutting the middle of these three points would result in a shortcut for a subpolyline of L_j^*. According to Lemma 3.2, the error of any such shortcut is upper bounded by $2\varepsilon(S_j^*)$. As the greedy algorithm selects the next point to shortcut based on the minimum possible induced error at the current stage, we thus conclude that $\varepsilon(S_i) \leq 2\varepsilon(S_j^*)$ has to hold as well.

Let now c_j be the number of shortcuts S_i assigned to a particular shortcut S_j^* in the optimal solution. Then the greedy sum-error can be upper bound by $\sum_{i=1}^{n-2} \varepsilon(S_i) \leq \sum_{j=1}^{n-2} c_j \cdot 2\varepsilon(S_j^*)$.

To complete the proof, we will argue that $c_j \leq 2$ for all $j = 1, \dots, n-2$. Assume now for contradiction that there exists a shortcut S_j^* to which we assigned at least three greedy shortcuts. Let the respective points that resulted in these shortcut insertions in that order be q_1, q_2, q_3. Based on the assignment criterion, we have $q_1, q_2, q_3 \in L_j^*$. Furthermore, there need to be three points that are not shortcutted yet at the point greedy considers q_3 in order for the respective shortcut to get assigned to S_j^*. Hence we have at least also $q_4, q_5 \in L_j^*$ that are shortcutted after q_3. Now we consider the point p^* that was shortcutted by S_j^* in the optimal solution. At that moment, there were only the points p_l, p^*, p_r left in L_j^*, where p_l is the left endpoint of S_j^* and p_r the right one. Hence the shortcuts (or original line segments) $S_l = (p_l, p^*)$ and $S_r = (p^*, p_r)$ exist in the optimal solution. If those are shortcuts, their index needs to be smaller than j, as they were constructed before S_j^*. Now if we have q_1, q_2, q_3, q_4, q_5 in L_j^*, at least three of them have to be contained in either the subpolyline belonging to S_l or S_r. W.l.o.g. assume it is S_l. That automatically implies that S_l is indeed a shortcut and not an original segment. Now if we shortcut the q_i with smallest index i in S_l (which implies $i \leq 3$), we assign S_i to S_l, as S_l has a smaller index than S_j^*, and there are at least three not yet shortcutted points in the respective subpolyline including q_i. This contradicts our claim that S_i is assigned to S_j^*. We therefore conclude that $c_j \leq 2$ holds. Accordingly, we get $\sum_{j=1}^{n-2} c_j \cdot 2\varepsilon(S_j^*) \leq \sum_{j=1}^{n-2} 2 \cdot 2\varepsilon(S_j^*) = 4 \sum_{j=1}^{n-2} \varepsilon(S_j^*) = 4 \cdot OPT$. ◀

Thus, a greedy ordering provides simultaneously a 2-approximation for max-error and a 4-approximation for sum-error (or average-error).

Regarding the running time, we observe that the greedy algorithm only needs to compute linearly many shortcut errors instead of the quadratically many required by the exact algorithm: There are the initial errors of shortcuts s_{ii+1} for $i = 1, \dots, n-1$ which can be computed in constant time each. Then, after selecting the next point to contract, only the values for its two former neighbors need to be updated. As there are $n-2$ contractions overall, the total number of non-trivial error computations is $2n-4$. If we use the standard approach by Alt and Godau, then computing the errors would take $\mathcal{O}(n^3)$ in total and thus the running time would not be better than for the exact approach. If we use parameteric search, then the time decreases to $\mathcal{O}(n^2 \log n)$. For these two variants the space consumption is linear. However, we can again exploit the FDS by Buchin et al. to achieve further speed-up (while accepting an increased space consumption).

▶ **Lemma 3.5.** *The greedy algorithm can be implemented to run in $\mathcal{O}(n^2)$ time and space.*

Proof. Recall that FDS requires a preprocessing time of $\mathcal{O}(n \cdot k^{3+\delta} + n^2)$ and has a query time of $\mathcal{O}(\frac{n}{k}\log^2 n + \log^4 n)$ for some $k \in [1, n]$. As we need to issue $\mathcal{O}(n)$ shortcut error queries, the total time of the greedy algorithm based on the DS can be expressed as $\mathcal{O}(n \cdot k^{3+\delta} + n^2 + \frac{n^2}{k}\log^2 n + n\log^4 n)$. If we choose k to be slightly smaller than $n^{1/3}$, e.g. $k = n^{(3-\delta)/9}$, then all summands are in $\mathcal{O}(n^2)$. Selecting $n-2$ times the one with minimum error among the current $\mathcal{O}(n)$ shortcut candidates can also be accomplished in $\mathcal{O}(n^2)$. ◀

4 Future Work

For the Hausdorff distance, the greedy algorithm could be implemented potentially even faster when using suitable data structures. But it is unclear whether this would also provide a constant factor approximation for gradual line simplification. Another direction for future work would be the investigation of different objective functions, as e.g. the sum of squared shortcut errors, or the sum over the shortcut errors of each individual simplification.

References

1 Pankaj K Agarwal, Sariel Har-Peled, Nabil H Mustafa, and Yusu Wang. Near-linear time approximation algorithms for curve simplification. *Algorithmica*, 42(3), 2005.

2 Helmut Alt and Michael Godau. Computing the fréchet distance between two polygonal curves. *International Journal of Computational Geometry & Applications*, 1995.

3 Kevin Buchin, Maximilian Konzack, and Wim Reddingius. Progressive simplification of polygonal curves. *Computational Geometry*, 2020.

4 Maike Buchin, Ivor van der Hoog, Tim Ophelders, Lena Schlipf, Rodrigo I. Silveira, and Frank Staals. Efficient Fréchet Distance Queries for Segments. In *30th Annual European Symposium on Algorithms (ESA 2022)*, 2022. doi:10.4230/LIPIcs.ESA.2022.29.

5 Wing Shiu Chan and Francis Chin. Approximation of polygonal curves with minimum number of line segments or minimum error. *International Journal of Computational Geometry & Applications*, 1996.

6 Sabine Storandt and Johannes Zink. Polyline simplification under the local Fréchet distance has subcubic complexity in 2D. *arXiv preprint arXiv:2201.01344*, 2022.

The Fréchet mean in the space of segments*

Sergio Cabello[1] **and Panos Giannopoulos**[2]

1 **Faculty of Mathematics and Physics, University of Ljubljana, Slovenia**
Institute of Mathematics, Physics and Mechanics, Slovenia.
`sergio.cabello@fmf.uni-lj.si`

2 **City, University of London, UK**
`Panos.Giannopoulos@city.ac.uk`

Abstract

The Fréchet mean generalizes the classic notion of mean or average to any abstract metric space. We study the Fréchet mean in the space of straight-line segments in $\mathbb{R}^2$ under the Hausdorff distance. First, we give an example of two segments with infinitely many Fréchet means. Then, we give an algorithm that computes a $(1+\epsilon)$-approximation to k-means clustering in $O(n^{8k-2+\eta}+(n/\varepsilon)^{4k+1}\log^{4k+1}(n/\varepsilon))$ time, for any $\eta > 0$.

1 Introduction

The Fréchet mean [7] is a generalization of the classic notion of mean or average to any abstract metric space. For an n-point set $P = \{p_1, \dots, p_n\}$ in a metric space $(\mathcal{M}, d)$, a Fréchet mean is any element in the set

$$\arg\min_{p\in\mathcal{M}} \sum_{i=1}^{n} d^2(p, p_i).$$

For Euclidean spaces, the Fréchet mean is the usual arithmetic mean. Other usual means can be recovered as Fréchet means by considering other distances. Note that, in general spaces, the Fréchet mean may not be unique or may be the empty set, if the minimum is not attained. The Fréchet mean has become a well-studied concept in Statistics and in Riemannian spaces, where sometimes it is known as Karcher mean.

Our objective is to understand the concept and the computation of the Fréchet mean in the space of segments in the plane under the Hausdorff distance. More generally, we would like to consider clustering problems in the space of segments, where the cost of each cluster is given by the functional defining the Fréchet mean. This is the k-means problem. (The 1-means problem is precisely the problem of computing the Fréchet-mean.)

1.1 Formalization of the problem

For each point $p \in \mathbb{R}^2$, we use $x(p)$ and $y(p)$ for its two coordinates. Thus, $p = (x(p), y(p))$. For any two points $p, q \in \mathbb{R}^2$, we denote by pq the segment with endpoints p and q, and by $|pq|$ the Euclidean distance between them: $|pq|^2 = (x(p) - x(q))^2 + (y(p) - y(q))^2$.

Let $(\mathcal{S}, d_H)$ be the space of closed straight-line segments in $\mathbb{R}^2$, where d_H is the Hausdorff distance. Recall that the Hausdorff distance $d_H(A, B)$ between any two closed subsets $A, B \subset \mathbb{R}^2$ is defined by

$$d_H(A, B) = \max\Big\{\max_{a\in A}\min_{b\in B}|ab|,\ \max_{b\in B}\min_{a\in A}|ab|\Big\}.$$

* Research partially supported by the Slovenian Research Agency (P1-0297, J1-2452, N1-0218, N1-0285).

Define $\delta(p, s) = \min_{q \in s} |pq|$ for the distance from a point p to a segment or line s. It is well known and easy to see that for any two segments $s_1 = a_1b_1$ and $s_2 = a_2b_2$ in $\mathcal{S}$

$$d_H(s_1, s_2) = \max\{\delta(a_1, s_2), \delta(b_1, s_2), \delta(a_2, s_1), \delta(b_2, s_1)\}.$$

We are interested in computing a Fréchet mean in $(\mathcal{S}, d_H)$. That is, given a set S of n segments, find a segment $s_1^* \in \mathcal{S}$ such that the so-called *Fréchet variance*

$$\mathrm{cost}_S(s_1) := \sum_{s \in S} d_H^2(s_1, s)$$

is minimized at s_1^*. More generally and motivated by clustering problems, for any k segments $s_1, \ldots, s_k$ playing the role of "centers" of the cluster, we define the objective function

$$\mathrm{cost}_S(\{s_1, \ldots, s_k\}) \ := \ \sum_{s \in S} \min\{d_H^2(s_1, s), \ldots, d_H^2(s_k, s)\}$$

and define the k-means problem as the problem of finding a minimizer.

As we shall see, the problem is already non-trivial even for the Fréchet mean of two segments. We then present a $(1+\varepsilon)$-approximation for the k-means problem.

1.2 Related work

The k-means and k-medians clustering problem has been studied extensively in geometric and general metric spaces. While the mean minimizes the sum of the squares of the distances, the median minimizes the sum of the distances, and each of these concepts has its own advantages. For general metric spaces, both k-means and k-median clustering are APX-hard [3]. For Euclidean k-means, where the input is a set of points, and in fixed dimension, several PTASs are known [4, 5]. For curves, which are a generalization of segments, previous work has focused on k-medians [1, 11]. The Fréchet mean has been considered for persistence diagrams [10, 13], point sets on the unit circle [2], and in the space of graphs [6, 8, 9], to name a few spaces. For a general, comprehensive treatment of Fréchet means see [12].

2 An example of Fréchet mean in $(\mathcal{S}, d_H)$

Let $s_1 = a_1b_1$, $s_2 = a_2b_2$ be perpendicular segments, centered at the origin o, with $|s_1| = |s_2| = 2$; see Figure 1. We show that the set of Fréchet means in this case is 3-dimensional.

Let D be the disk centered at o with radius 1 and let $D_1, \ldots, D_4$ be the disks centered at $a_1/2, a_2/2, b_1/2, b_2/2$ with radius $\frac{1}{2}$. The region $D \setminus (D_1 \cup \cdots \cup D_4)$ has four connected regions. Let A_i be the closure of the connected region in the i-th quadrant; see Figure 1.

▶ **Theorem 2.1.** *A segment is a Fréchet mean of $\{s_1, s_2\}$ if and only if it goes through the origin and has its endpoints in different regions $A_1, \ldots, A_4$.*

Proof. Note that $d_H(s_1, s_2) = 1$. Therefore, for any solution (Fréchet mean) segment s we have $d_H^2(s, s_1) + d_H^2(s, s_2) \leq d_H^2(s_1, s_1) + d_H^2(s_1, s_2) = 0 + d_H^2(s_1, s_2) = 1$.

Observe that any solution segment s must pass through the origin. Otherwise, both of $d_H(s, s_1)$ and $d_H(s, s_2)$ can be simultaneously decreased by translating s such that its center coincides with the origin.

Next, consider a solution segment $s = ab$ that passes through the origin o with its endpoints a, b in the first and third quadrant, respectively. See Figure 1. Let a_c and a' be the points of intersection of the supporting line ℓ of s with the boundary of D and the

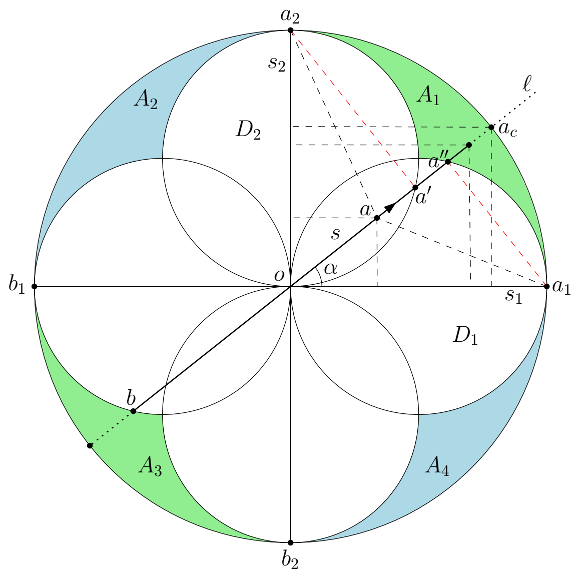


Figure 1 For the set of segments $S = \{s_1, s_2\}$, there is an infinite family of Fréchet means.

boundary of D_2 respectively. Note that $\angle oa'a_2$ is right. As a moves from o to a', we have $\delta(a_2, s) = |a_2 a| > \delta(a, s_2)$ and the unique minimum of $|a_2 a|$ is attained at $a = a'$. When a moves from a' to a_c, $\delta(a_2, s) = |a_2 a'|$ is constant. At $a = a_c$ we have that $\delta(a_2, s) = \delta(a_c, s_2)$, and beyond that $\delta(a_2, s) < \delta(a, s_2)$. Similarly, the minimum value of $\delta(a_1, s)$ is $|a_1 a''|$, where a'' is the projection of a_1 onto ℓ (the intersection of ℓ with the boundary of D_1). We also have that $\delta(a_1, s) = \delta(a_c, s_1)$ when a lies between a'' and a_c, and beyond a_c, $\delta(a_1, s) < \delta(a, s_1)$.

Assume that $a'' \in a'a_c$, as in Figure 1; the other case is symmetric. Using $\alpha := \angle aoa_1$, for every position of a in the segment $a''a_c$ we have

$$\delta^2(a, s_1)+\delta^2(a, s_2) \ \le\ \delta^2(a_1, s)+\delta^2(a_2, s) \ =\ |a_1a''|^2+|a_2a'|^2 \ =\ \sin^2\alpha+\sin^2(\pi/2-\alpha) \ =\ 1.$$

For every position of a in the segment oa'' we have $\delta^2(a_1, s)+\delta^2(a_2, s) \ >\ |a_1a''|^2+|a_2a'|^2 \ =\ 1$, while for every position of a past a_c we have that $\delta^2(a, s_1)+\delta^2(a, s_2) > \delta^2(a_1, s)+\delta^2(a_2, s) = 1$.

For the endpoint b of s in the third quadrant, the situation is symmetric. Using that the Haussdorf distance is attained at some endpoint, we get that each segment with an endpoint in A_1 and an endpoint in A_3 gives the minimum possible value of $d_H^2(s, s_1) + d_H^2(s, s_2) \ge 1$, and therefore is a Fréchet mean. The situation is symmetric for the other quadrants. ◀

3 A $(1 + \epsilon)$-approximation for k-means in $(\mathcal{S}, d_H)$

We use the following adaptation of the definition of Vigneron [14, Section 2.1]. Let $\mathcal{F} = \{f_i : \mathbb{R}^d \to \mathbb{R} \mid i \in I\}$ be a finite family of functions, where I is some index set. We say that $\mathcal{F}$ is *nice* if there exists a constant $\lambda > d > 0$ such that:

- each $f_i \in \mathcal{F}$ is nonnegative and bounded;
- for each $f_i \in \mathcal{F}$, there exists a semialgebraic set $\mathrm{supp}(f_i) \subseteq \mathbb{R}^d$ and an algebraic function g_i of degree at most λ with $f_i(x) = g_i(x)$ for $x \in \mathrm{supp}(f_i)$ and $f_i(x) = 0$ for $x \notin \mathrm{supp}(f_i)$;
- for each $f_i \in \mathcal{F}$, the semialgebraic set $\mathrm{supp}(f_i) \subseteq \mathbb{R}^d$ is a boolean combination of at most λ subsets of $\mathbb{R}^d$, each of them defined by an algebraic inequality of degree at most λ;
- for each $f_i \in \mathcal{F}$, the restriction of f_i to $\mathrm{supp}(f_i)$ is continuous.

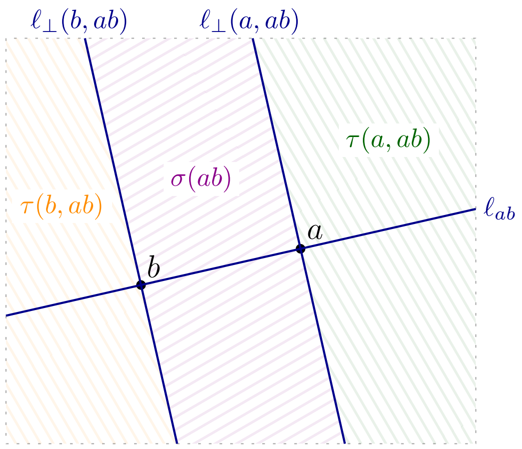


Figure 2 The regions $\sigma(ab)$, $\tau(a,ab)$ and $\tau(b,ab)$.

Our use of this concept will be through the following result for computing an approximation to the minimum of the function $\sum_i f_i$.

▶ **Theorem 3.1** (Adaptation of Theorem 3.4 in Vigneron [14]). *Assume that $\varepsilon \in (0,1)$. Let $\mathcal{F} = \{f_i : \mathbb{R}^d \to \mathbb{R} \mid i \in I\}$ be a nice family of m functions. Define $g = \sum_{i \in I} f_i$ and assume that $\min_{x \in \mathbb{R}^d} g(x)$ exists. Then we can compute a point $x'_\varepsilon \in \mathbb{R}^d$ such that $g(x'_\varepsilon) \leq (1+\varepsilon) \min_{x \in \mathbb{R}^d} g(x)$ in time $O(m^{2d-2+\eta} + (m/\varepsilon)^{d+1} \log^{d+1}(m/\varepsilon))$ for any $\eta > 0$. The constant hidden in the O-notation depends on η.*

Let us first consider the simpler case of two segments ab and $a'b'$ and how their Hausdorff distance is defined. We parameterize a segment $s = ab$ as the point $(x(a), y(a), x(b), y(b))$ in $\mathbb{R}^4$. Note that in this parameterization the segments ab and ba give different points in $\mathbb{R}^4$.

Let ℓ_{ab} be the line supporting the segment ab. For a point p, the distance $\delta(p, ab)$ is given by one of the three terms $|pa|$, $|pb|$, or $\delta(p, \ell_{ab})$. For each point $q \in \mathbb{R}^2$ and each segment ab, let $\ell_\perp(q, ab)$ be the line perpendicular to ℓ_{ab} through q. The lines $\ell_\perp(a, ab)$ and $\ell_\perp(b, ab)$ partition the plane into three 2-dimensional faces (Figure 2) with closures

$$\begin{aligned} \sigma(ab) &= \text{the closed slab between } \ell_\perp(a,ab) \text{ and } \ell_\perp(b,ab), \\ \tau(a,ab) &= \text{the closed halfspace defined by } \ell_\perp(a,ab) \text{ that does not contain } b, \\ \tau(b,ab) &= \text{the closed halfspace defined by } \ell_\perp(b,ab) \text{ that does not contain } a. \end{aligned}$$

We then have

$$\delta(p, ab) = \begin{cases} |pa| & \text{if } p \in \tau(a,ab), \\ |pb| & \text{if } p \in \tau(b,ab), \\ \delta(p, \ell_{ab}) & \text{if } p \in \sigma(ab). \end{cases}$$

For any two segments ab and $a'b'$, $d_H(ab, a'b')$ is given by one of the functions in the family

$$\mathcal{F}(ab, a'b') := \big\{|aa'|,\ |ab'|,\ |ba'|,\ |bb'|,\ \delta(a, \ell_{a'b'}),\ \delta(b, \ell_{a'b'}),\ \delta(a', \ell_{ab}),\ \delta(b', \ell_{ab})\big\}.$$

We next argue that all the expressions involved are algebraic. A point p lies on the line $\ell_\perp(a, ab)$ if and only the scalar product of the vectors $\vec{ap}$ and $\vec{ab}$ is zero. This is equivalent to $\big(x(p), y(p), x(a), y(a), x(b), y(b)\big)$ being a zero of the algebraic function

$$\psi(x, y, x_a, y_a, x_b, y_b) := (x - x_a)(x_b - x_a) + (y - y_a)(y_b - y_a).$$

The sign of this expression also tells on which side of $\ell_\perp(a,ab)$ the point p lies. Symmetrically, the sign of $\psi\big(x(p),y(p),x(b),y(b),x(a),y(a)\big)$ also tells on which side of $\ell_\perp(b,ab)$ point p is.

In the following, we will treat the segment $a'b'$ as variable, identified with $\mathbb{R}^4$, while the segment ab will be fixed. We next show that the space $\mathbb{R}^4$ can be decomposed into cells such that, within a cell, the distance is defined always by the same function from $\mathcal{F}(ab,a'b')$. Such a decomposition is given by the eight algebraic hypersurfaces describing the conditions

$$a' \in \ell_\perp(a,ab),\ b' \in \ell_\perp(a,ab),\ a' \in \ell_\perp(b,ab),\ b' \in \ell_\perp(b,ab),$$
$$a \in \ell_\perp(a',a'b'),\ b \in \ell_\perp(a',a'b'),\ a \in \ell_\perp(b',a'b'),\ b \in \ell_\perp(b',a'b').$$

Finally, we note that each function in $\mathcal{F}(ab,a'b')$ is algebraic.

We parameterize the space of (sequences of) k segments $a_1b_1,\dots,a_kb_k$ by the point

$$\big(x(a_1),y(a_1),x(b_1),y(b_1),\dots,x(a_k),y(a_k),x(b_k),y(b_k)\big) \in \mathbb{R}^{4k}.$$

Similarly, each $z \in \mathbb{R}^{4k}$ defines a k-tuple of segments with $s_1(z)=a_1(z)b_1(z),\dots,s_k(z)=a_k(z)b_k(z)$ by taking the inverse of the parameterization.

▶ **Theorem 3.2.** *Let k be a fixed, positive integer and let s be a segment in the plane. In $O(1)$ time we can construct a nice family $\mathcal{F}_s=\{f:\mathbb{R}^k\to\mathbb{R}\}$ of $O(1)$ functions such that*

$$\forall z \in \mathbb{R}^k: \quad \sum_{f\in\mathcal{F}_s} f(z) = \min_{i\in[k]} d_H(s,s_i(z))^2.$$

Proof. Let $s=ab$ be the fixed segment. For each index $i\in[k]$, we consider the set $\Sigma(i)$ of 8 hypersurfaces in $\mathbb{R}^{4k}$, each of them given by one of the following conditions

$$a_i \in \ell_\perp(a,ab),\ b_i \in \ell_\perp(a,ab),\ a_i \in \ell_\perp(b,ab),\ b_i \in \ell_\perp(b,ab),$$
$$a \in \ell_\perp(a_i,a_ib_i),\ b \in \ell_\perp(a_i,a_ib_i),\ a \in \ell_\perp(b_i,a_ib_i),\ b \in \ell_\perp(b_i,a_ib_i).$$

Note that here $x(a)$, $y(a)$, $x(b)$ and $y(b)$ are input data while $x(a_i)$, $y(a_i)$, $x(b_i)$ and $y(b_i)$ are variables defining coordinates in the parameter space $\mathbb{R}^{4k}$.

Let $\Sigma=\cup_{i\in[k]}\Sigma(i)$ and let $\mathcal{A}_\Sigma$ be the arrangement in $\mathbb{R}^{4k}$ induced by Σ. From the foregoing discussion, we have the following property: for each cell c of $\mathcal{A}_\Sigma$ and each index $i\in[k]$, the distance $d_H(s,s_i(z))$ is described by the same function from $\mathcal{F}(s,a_ib_i)$ for all $z\in c$. To clarify that only the coordinates of a_i,b_i are relevant in the functions in $\mathcal{F}(s,a_ib_i)$, we change the notation to $\mathcal{G}_i$ and take each function g_i of $\mathcal{G}_i$ to map from $\mathbb{R}^{4k}$ to $\mathbb{R}$. Formally, for each function $f_i\in\mathcal{F}(s,a_ib_i)$ we put into $\mathcal{G}_i$ the function $g_i(z):=f_i(ab,s_i(z))$.

We next define a set $\Lambda(s)$ of hypersurfaces in $\mathbb{R}^{4k}$ playing the role of "bisectors". For each $i,j\in[k]$ with $i<j$, we define $\Lambda(i,j)$ as the hypersurfaces given by setting any of the functions of $\mathcal{G}_i$ equal to any of the functions of $\mathcal{G}_j$. Since $\mathcal{G}_i$ has 8 functions for each $i\in[k]$, the set $\Lambda(i,j)$ has $8^2=64$ hypersurfaces.

We set $\Lambda=\cup_{i\in[k]}\cup_{j\in[k]\setminus[i]}\Lambda(i,j)$. Let $\mathcal{A}_\Lambda$ be the arrangement in $\mathbb{R}^{4k}$ induced by Λ. For each cell $c\in\mathcal{A}_\Lambda$ the sign of each function $g_i(z)-g_j(z)$ remains constant.

Finally, let $\mathcal{A}$ be the arrangement in $\mathbb{R}^{4k}$ induced by the hypersurfaces in $\Sigma\cup\Lambda$. Consider a cell $c\in\mathcal{A}$. Because c is contained in a cell of $\mathcal{A}_\Sigma$, for each $i\in[k]$ there is some function $g_{c,i}\in\mathcal{G}_i$ such that $d_H(s,s_i(z))=g_{c,i}(z)$ for all $z\in c$. Moreover, because c is contained in a cell of $\mathcal{A}_\Lambda$, for each distinct $i,j\in[k]$ the sign of

$$d_H(s,s_i(z))-d_H(s,s_j(z)) \;=\; g_{c,i}(z)-g_{c,j}(z)$$

is constant for all $z\in c$. This implies that, for each cell $c\in\mathcal{A}$, there exists some index $\iota(c)\in[k]$ with the following property: for each $z\in c$ it holds $\min_{i\in[k]} d_H(s,s_i(z))=g_{c,\iota(c)}(z)$.

For any set A, let 1_A be the function with $1_A(x) = 1$ if $x \in A$ and $1_A(x) = 0$ if $x \notin A$. For each cell $c \in \mathcal{A}$, define the function $h_c : \mathbb{R}^k \to \mathbb{R}$ by $h_c(z) = 1_c(z) \cdot g_{c,\iota(c)}(z)$. Finally, set $\mathcal{F} := \{h_c \mid c \in \mathcal{A}\}$. We can then express the function $z \in \mathbb{R}^{4k} \mapsto \min\{d_H(s, s_i(z) \mid i \in [k]\}$ as

$$\min_{i\in[k]} d_H(s, s_i(z)) \;=\; \sum_{c\in\mathcal{A}} 1_c(z)\, g_{c,\iota(c)}(z) \;=\; \sum_{c\in\mathcal{A}} h_c(z) \;=\; \sum_{h\in\mathcal{F}} h(z).$$

Since $\Sigma \cup \Lambda$ has $O(k^2) = O(1)$ hypersurfaces, the arrangement $\mathcal{A}$ has $O(1)$ cells, each of them described by $O(1)$ algebraic inequalities of constant description complexity and the family of functions $\mathcal{F}$ has the desired properties. ◀

▶ **Theorem 3.3.** *Let k a fixed, positive integer and let $\varepsilon \in (0, 1)$. Let S be a family of $n > k$ segments in the plane. We can compute k segments $s_{1,\varepsilon}, \dots, s_{k,\varepsilon}$ in $\mathbb{R}^2$ such that*

$$\mathrm{cost}_S(\{s_{1,\varepsilon}, \dots, s_{k,\varepsilon}\}) \;\le\; (1+\varepsilon) \min\Big\{\mathrm{cost}_S(\{s_1, \dots, s_k\}) \mid s_1, \dots, s_k \textit{ segments}\Big\}.$$

in time $O(n^{8k-2+\eta} + (n/\varepsilon)^{4k+1} \log^{4k+1}(n/\varepsilon))$, for any $\eta > 0$. The constant hidden in the O-notation depends on η and on k.

Proof. For each segment $s \in S$ we compute the family $\mathcal{F}_s$ of Theorem 3.2. Define $\mathcal{F} = \cup_{s\in S}\mathcal{F}_s$ and the function $g = \sum_{f\in\mathcal{F}} f$. Note that $\mathcal{F}$ is a family of $O(n)$ nice functions and

$$\forall z \in \mathbb{R}^{4k} : \quad g(z) \;=\; \sum_{s\in S}\sum_{f\in\mathcal{F}_s} f(z) \;=\; \sum_{s\in S} \min_{i\in[k]} d_H(s, s_i(z))^2 \;=\; \mathrm{cost}_S(\{s_1(z), \dots, s_k(z)\}).$$

We can then use Theorem 3.1 to find in time $O(|\mathcal{F}|^{2\cdot4k-2+\eta} + (|\mathcal{F}|/\varepsilon)^{4k+1} \log^{4k+1}(|\mathcal{F}|/\varepsilon))$, for any $\eta > 0$, a point $z'_\varepsilon \in \mathbb{R}^{4k}$ such that

$$g(z'_\varepsilon) \;\le\; (1+\varepsilon) \min_{z\in\mathbb{R}^{4k}} \mathrm{cost}_S(\{s_1(z), \dots, s_k(z)\}).$$

The point $z'_\varepsilon \in \mathbb{R}^{4k}$ defines the segments $s_{1,\varepsilon} := s_1(z'_\varepsilon), \dots, s_{k,\varepsilon} := s_k(z'_\varepsilon)$. Since $s_1(z), \dots, s_k(z)$ goes over all k tuples of segments when z iterates over all $\mathbb{R}^{4k}$, we have

$$\mathrm{cost}_S(\{s_{1,\varepsilon}, \dots, s_{k,\varepsilon}\} \;=\; g(z'_\varepsilon) \;\le\; (1+\varepsilon) \min_{s_1,\dots,s_k} \mathrm{cost}_S(\{s_1, \dots, s_k\}). \quad ◀$$

The next natural step in this line of work is to construct coresets to reduce the number of segments to consider, while keeping a good estimate of $\mathrm{cost}_S(\cdot)$, and apply Theorem 3.3 to the coreset.

References

1 Maike Buchin, Anne Driemel, and Dennis Rohde. Approximating (k, ℓ)-median clustering for polygonal curves. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms, SODA 2021*, pages 2697–2717, 2021.

2 Frédéric Cazals, Bernard Delmas, and Timothee O'Donnell. Fréchet mean and p-mean on the unit circle: Decidability, algorithm, and applications to clustering on the flat torus. In *19th International Symposium on Experimental Algorithms, SEA 2021*, volume 190 of *LIPIcs*, pages 15:1–15:16, 2021.

3 Vincent Cohen-Addad, Karthik C. S., and Euiwoong Lee. On approximability of clustering problems without candidate centers. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms, SODA 2021*, pages 2635–2648, 2021.

4 Vincent Cohen-Addad, Philip N. Klein, and Claire Mathieu. Local search yields approximation schemes for k-means and k-median in Euclidean and minor-free metrics. *SIAM Journal on Computing*, 48(2):644–667, 2019.

5 Dan Feldman, Melanie Schmidt, and Christian Sohler. Turning big data into tiny data: Constant-size coresets for k-means, PCA, and projective clustering. *SIAM Journal on Computing*, 49(3):601–657, 2020.

6 Daniel Ferguson and François G. Meyer. Computation of the sample Fréchet mean for sets of large graphs with applications to regression. In *Proceedings of the 2022 SIAM International Conference on Data Mining, SDM 2022*, pages 379–387, 2022.

7 M. Fréchet. Les éléments aléatoires de nature quelconque dans un espace distancié. *Annales de L'Institut Henri Poincaré*, 10(4):215–310, 1948.

8 Eric D. Kolaczyk, Lizhen Lin, Steven J. Rosenberg, Jie Xu, and Jackson Walters. Averages of unlabeled networks: Geometric characterization and asymptotic behavior. *The Annals of Statistics*, 48(1):514–538, 2020.

9 François G. Meyer. The Fréchet mean of inhomogeneous random graphs. In *Complex Networks & Their Applications X - Volume 1, Proceedings of the Tenth International Conference on Complex Networks and Their Applications, COMPLEX NETWORKS 2021*, volume 1015 of *Studies in Computational Intelligence*, pages 207–219, 2021.

10 Yuriy Mileyko, Sayan Mukherjee, and John Harer. Probability measures on the space of persistence diagrams. *Inverse Problems*, 27(12):124007, 2011.

11 Abhinandan Nath and Erin Taylor. k-median clustering under discrete Fréchet and Hausdorff distances. In *36th International Symposium on Computational Geometry, SoCG 2020*, volume 164 of *LIPIcs*, pages 58:1–58:15, 2020.

12 Christof Schötz. *The Fréchet Mean and Statistics in Non-Euclidean Spaces.* PhD thesis, Heidelberg University, The Faculty of Mathematics and Computer Science, 2021.

13 Katharine Turner, Yuriy Mileyko, Sayan Mukherjee, and John Harer. Fréchet means for distributions of persistence diagrams. *Discret. Comput. Geom.*, 52(1):44–70, 2014.

14 Antoine Vigneron. Geometric optimization and sums of algebraic functions. *ACM Trans. Algorithms*, 10(1):4:1–4:20, 2014.

On the Arrangement of Hyperplanes Determined by n Points

Michal Opler[1], **Pavel Valtr**[2], **and Tung Anh Vu**[3]

1 **Czech Technical University in Prague**
`michal.opler@fit.cvut.cz`
2 **Charles University**
`valtr@kam.mff.cuni.cz`
3 **Charles University**
`tung@kam.mff.cuni.cz`

Abstract

For any $d \leq 6$ and for any n, we determine the maximum number of cells in the arrangement of hyperplanes determined by n points in $\mathbb{R}^d$. It is shown that this number can be expressed as a polynomial in n of degree d^2 for any fixed d, and exact formulas for the first $d-1$ coefficients of this polynomial are given.

1 Introduction

Arrangements of lines in the plane and their higher-dimensional generalization, arrangements of hyperplanes in $\mathbb{R}^d$, are a basic geometric structure. If a finite set of hyperplanes is *in general position*, which means that the intersection of every k hyperplanes is $(d-k)$-dimensional, $k = 2, 3 \ldots, d+1$, the arrangement is called *simple*. If the hyperplanes of a hyperplane arrangement $\mathcal{A}$ are removed from $\mathbb{R}^d$, the remaining part of $\mathbb{R}^d$ consists of connected components called *cells of* $\mathcal{A}$. The following proposition implies that the number of cells of a simple arrangement of n hyperplanes in $\mathbb{R}^d$ is a function of n and d only, and is thus independent of the arrangement.

▶ Proposition 1. The number of cells in a simple arrangement of n hyperplanes in $\mathbb{R}^d$ is

$$\Phi_d(n) = \binom{n}{0} + \binom{n}{1} + \cdots + \binom{n}{d}.$$

For $d = 2$, Proposition 1 says that n lines in general position in the plane partition the plane into $\Phi_2(n) = \binom{n}{0} + \binom{n}{1} + \binom{n}{2} = \frac{n^2}{2} + \frac{n}{2} + 1$ cells. This fact is well known also outside of the discrete and computational geometry community due to the fact that it has several elementary proofs which nicely demonstrate the principle of mathematical induction. Proposition 1 for general d also has simple proofs using mathematical induction.

In this paper we consider arrangements of all hyperplanes in $\mathbb{R}^d$ determined by d-element subsets of a given set of n points in general position in $\mathbb{R}^d$. In particular, we are interested in the (maximum) number of cells in such arrangements.

Let P be a set of $n \geq d$ points in general position in $\mathbb{R}^d$. The affine hull of each d-tuple of points of P is a hyperplane. We denote the arrangement of these $\binom{n}{d}$ hyperplanes by $\mathcal{A}(P)$, or by $\mathcal{A}(p_1, \ldots, p_n)$ if $P = \{p_1, \ldots, p_n\}$.

We find it a very natural question to ask how many cells $\mathcal{A}(P)$ can have. Surprisingly, as far as we know, this question has been considered only in dimensions 2 and 3 so far. We study this question for general d. If P is in a "sufficiently general" position, the number of cells, denoted $f_d(n)$, depends only on n and d. By a computer assisted proof, we determine

39th European Workshop on Computational Geometry, Barcelona, Spain, March 29–31, 2023.

$f_d(n)$ for $d \leq 5$ and any $n \geq d$. We can also determine $f_6(n)$ using a result of Koizumi et al. [3] who studied the so-called characteristic polynomial of hyperplane arrangements in vector spaces. It turns out that Koizumi et al. compute the characteristic polynomial of arrangements up to dimension six which are equivalent to $\mathcal{A}(P)$. This is discussed in more detail in the initial part of Section 3. We also show that for any fixed d, $f_d(n)$ can be expressed as a polynomial in n of degree d^2, and in Theorem 3.2 we give exact formulas for the first two coefficients of this polynomial, which shows the growth rate of $f_d(n)$ for any fixed d. In the second part of Theorem 3.2 we actually show a stronger result that for any d, the first $d-1$ coefficients of $f_d(n)$ and $\Phi_d(\binom{n}{d})$ are equal.

When trying to find some results about the numbers $f_d(n)$, we found a discussion on mathoverflow [2] where the question was asked by Min Wu in February 2020. Few days later Richard Stanley outlined on the same place how to obtain $f_2(n)$ and $f_3(n)$. The computation of $f_3(n)$ required some case distinction and relatively complicated formulas appeared in the computation. Stanley wrote that there could be an error in the computation. It turns out that his formula for $f_3(n)$ was not quite correct but the method works. We managed to correct the formula and extend the method to higher dimensions.

We now prepare for the definition of the type of "sufficiently general" position which is suitable for us. Let P be a set of $n \geq d$ points in general position in $\mathbb{R}^d$. We say that a hyperplane arrangement $\mathcal{B}$ is *central* if it has a non-empty intersection, i.e., if $\bigcap_{H\in\mathcal{B}} H \neq \emptyset$. We associate with every central subarrangement $\mathcal{B}$ of $\mathcal{A}(P)$ a poset $\mathcal{P}_\mathcal{B}$ of sets ordered by inclusion defined as

$$\mathcal{P}_\mathcal{B} := \{F \subseteq P \mid \exists H \in \mathcal{B} : \cap\mathcal{B} \subseteq \operatorname{aff}(F) \subseteq H\}.$$

In other words, $\mathcal{P}_\mathcal{B}$ contains all tuples of points from P spanning an affine space that (i) contains the common intersection of $\mathcal{B}$, and simultaneously (ii) is contained in some hyperplane $H \in \mathcal{B}$. Our definition of a "sufficiently general" position ensures that if a subarrangement $\mathcal{B}$ of $\mathcal{A}(P)$ is central then the intersection $\bigcap\mathcal{B}$ is given by the structure of the minimal elements of $\mathcal{P}_\mathcal{B}$. The *support $\mathcal{S}_\mathcal{B}$ of $\mathcal{B}$* is the set system consisting of the minimal elements of $\mathcal{P}_\mathcal{B}$. If the sets in $\mathcal{S}_\mathcal{B}$ are denoted by $S_1, \dots, S_k$, we have $\bigcap\mathcal{B} = \bigcap_{i=1}^k \operatorname{aff}(S_i)$. It follows from a basic result in linear algebra that, under the assumption that $\mathcal{B}$ is a central arrangement, $\operatorname{codim}(\bigcap\mathcal{B}) \leq \sum_{i=1}^k \operatorname{codim}(\operatorname{aff}(S_i))$. Intuitively, the previous inequality is strict in case of certain degeneracy.

We say that P is in a *very general position*, if for any central subarrangement $\mathcal{B} \subseteq \mathcal{A}(P)$ with support $\{S_1, \dots, S_k\}$ we have

$$\operatorname{codim}\left(\bigcap\mathcal{B}\right) = \sum_{i=1}^k \operatorname{codim}(\operatorname{aff}(S_i)), \tag{1}$$

which is equivalent to

$$\dim\left(\bigcap\mathcal{B}\right) = \sum_{i=1}^k |S_i| - (k-1)\cdot(d+1) - 1. \tag{2}$$

An example of six points in general position in the plane which are not in very general position is depicted in Figure 1, where the arrangement $\mathcal{B}$ of the three lines p_iq_i, $i = 1, \dots, 3$, intersecting in a common point has support $\mathcal{S}_\mathcal{B} = \{\{p_1, q_1\}, \{p_2, q_2\}, \{p_3, q_3\}\}$, and it holds that $2 = \operatorname{codim}(\bigcap\mathcal{B}) \neq \sum_{i=1}^3 \operatorname{codim}(\operatorname{aff}(S_i)) = 1 + 1 + 1 = 3$.

In the full version of this paper, it is shown that any set in general position can be perturbed to a set in a very general position. This proposition is used in the proof of the

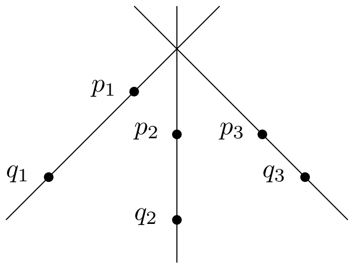


Figure 1 The lines given by pairs of points $\{p_i, q_i\}, i = 1, 2, 3$, have a common intersection.

main results, and it also easily implies that any hyperplane arrangement determined by a set of n points in $\mathbb{R}^d$ has at most $f_d(n)$ cells. Indeed, if we slightly perturb any point set to a set in general position and then to a set in very general position, the number of cells cannot decrease and it reaches exactly the value of $f_d(n)$.

We were able to find the related sequences in the On-Line Encyclopedia of Integer Sequences [7]. Sequence `A055503` [6] corresponds to the number of cells in an arrangement of lines determined by n points in very general position in the plane, i.e. $f_2(n)$. Sequence `A002817` [4] corresponds to the number of such cells which are bounded. Sequence `A037255` [5] corresponds to the number of cells in an arrangement of lines determined by a generic set of n points in the real projective plane. (The sequence for the real projective plane can be obtained from the previous two sequences as the arithmetic mean of the two sequences, since if we embed a real plane containing a line arrangement to a real projective plane then each pair of opposite unbounded cells merges into a single cell.)

Open problems

A natural open problem is to determine or estimate the maximum number of k-faces in a hyperplane arrangement determined by a set of n points in $\mathbb{R}^d$. It is widely open if there is a closed formula for $f_d(n)$ similar to the one given for $\Phi_d(n)$ in Proposition 1.

2 Warm-up: two-dimensional space

In this section, we count the number of cells in a line arrangement determined by a set P of n points in very general position in the two-dimensional space. For each cell we assume without loss of generality that the bottommost point is unique if it exists. Let P' be the set of intersections between all pairs of lines given by $\binom{P}{2}$, and let $Q = P' \setminus P$. Each cell C satisfies exactly one of the following three conditions:

1. the bottommost point of C belongs to P,
2. the bottommost point of C belongs to Q,
3. C does not have a bottommost point.

In the first case, every point $p \in P$ can be a bottommost point of a cell. There are $n-1$ lines passing through p, so there are $n-2$ cells with p as its bottommost point, see Figure 2 for an example. Thus, the number of cells satisfying condition 1. is

$$n \cdot (n-2). \tag{3}$$

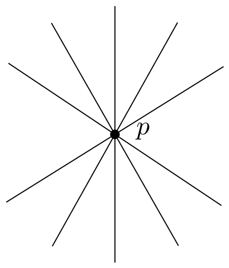


Figure 2 There are five lines passing through p and only four regions with p as its bottommost point.

In the second case, the cell C is given by a point $q \in Q$. The number of such points, and in turn the number of cells satisfying condition 2. is

$$\frac{1}{2} \cdot \binom{n}{2} \cdot \binom{n-2}{2}. \tag{4}$$

To count the number of cells that are unbounded from below, consider a horizontal line ℓ which lies below all points of P'. Then each unbounded cell is intersected by ℓ exactly once. Let L be the set of lines given by all pairs of points of P, and let Q be the set of points in which lines of L and ℓ intersect. Projecting each unbounded cell onto ℓ lets us count the number of unbounded cells as the number of line segments on ℓ given by Q and two additional cells are not bounded from either the left or the right. As such, the number of unbounded cells is

$$\binom{n}{2} + 1. \tag{5}$$

By (3), (4), and (5) we have

$$f_2(n) = n(n-2) + \binom{n}{2}\left(\frac{1}{2}\binom{n-2}{2} + 1\right) + 1. \tag{6}$$

3 Computing the characteristic polynomial

In this section, we focus on computing the characteristic polynomial of the arrangement using a mechanical method that can be implemented by a program. The number of cells as well as the number of bounded cells can then easily be retrieved from the polynomial.

First, let us formally introduce the characteristic polynomial. Recall that a hyperplane arrangement $\mathcal{A}$ is central if $\bigcap_{H \in \mathcal{A}} H \neq \emptyset$. The *rank* of $\mathcal{A}$, denoted by $\operatorname{rank}(\mathcal{A})$, is the dimension of the space spanned by the normals to the hyperplanes in $\mathcal{A}$. For any central arrangement $\mathcal{A}$, we have $\operatorname{rank}(\mathcal{A}) = \operatorname{codim}(\bigcap \mathcal{A})$. The *characteristic polynomial* of a hyperplane arrangement $\mathcal{A}$, denoted by $\chi_{\mathcal{A}}(t)$, is defined as

$$\chi_{\mathcal{A}}(t) = \sum_{\substack{\mathcal{B} \subseteq \mathcal{A} \\ \mathcal{B} \text{ central}}} (-1)^{|\mathcal{B}|} t^{d - \operatorname{rank}(\mathcal{B})} = \sum_{\substack{\mathcal{B} \subseteq \mathcal{A} \\ \mathcal{B} \text{ central}}} (-1)^{|\mathcal{B}|} t^{\dim(\bigcap \mathcal{B})}.$$

Note that the characteristic polynomial is typically defined using a so-called intersectional lattice associated with $\mathcal{A}$ and its Möbius function. The equivalence of the definition above is due to Whitney's theorem [8, Lemma 2.3.8]. We chose to omit the standard definition as our approach really boils down to computing the characteristic polynomial as a sum over all central subarrangements of $\mathcal{A}$. Note that this differs from the approach of Koizumi et al. [3]

who compute the Möbius function of the intersection lattice and only subsequently recover the polynomial via its more usual definition.

The connection between the characteristic polynomial and the number of cells of a hyperplane arrangement is a celebrated result by Zaslavsky [9]. The number of cells of an hyperplane arrangement $\mathcal{A}$ in a real d-dimensional space is equal to $(-1)^d \chi_{\mathcal{A}}(-1)$, while the number of bounded cells is obtained as $(-1)^{\mathrm{rank}(\mathcal{A})} \chi_{\mathcal{A}}(1)$.

3.1 Algorithm for general d-dimensional space

Inspired by the approach used in Section 2, we devise an algorithm that computes the characteristic polynomial of n points in very general position in d-dimensional space expressed as a polynomial in both t and n.

▶ **Theorem 3.1.** *There is an algorithm that receives an integer $d \in \mathbb{N}$ as input and outputs a polynomial $Q(t, n)$ such that for arbitrary integer $n \geq d$ and an arbitrary set of n points $P_n \subset \mathbb{R}^d$ in very general position, we have $Q(t, n) = \chi_{\mathcal{A}(P_n)}(t)$.*

Now, we briefly try to sketch the basic idea behind the algorithm. Recall that we associate with any subarrangement $\mathcal{B}$ a poset $\mathcal{P}_{\mathcal{B}}$ of sets ordered by inclusion defined in Section 1. Furthermore, recall that we defined the support $\mathcal{S}_{\mathcal{B}}$ as the collection of all minimal sets of $\mathcal{P}_{\mathcal{B}}$. Our goal is to show how central arrangements with different supports contribute to the characteristic polynomial.

We notice that there are only finitely many ways how the support of a central subarrangement can look like. We call these classes of isomorphic supports *types.* It is easy to prove that there are only finitely many non-isomorphic types as each can contain at most d sets. For example in three-dimensional space, a central arrangement $\mathcal{B}$ intersecting in a common line can have the following three possible types of support: (i) a single pair of points $\{p, q\}$ in the case when all the hyperplanes in $\mathcal{B}$ contain the line spanned by p and q, (ii) two disjoint triples of points $\{p_1, q_1, r_1\}, \{p_2, q_2, r_2\}$ in the case when $\mathcal{B}$ contains precisely the two hyperplanes spanned by these triples, and (iii) two triples sharing one common point $\{p, q_1, r_1\}, \{p, q_2, r_2\}$ which again corresponds to an arrangement $\mathcal{B}$ containing precisely the two hyperplanes.

The algorithm computes $Q(t, n)$ by enumerating all possible support types and summing the contributions of all central subarrangements with a given support type. However, we remark that this is far from a full description of the algorithm since there is a great deal of non-trivial care needed to handle overcounting.

3.2 Three-, four- and five-dimensional space

We were able to successfully compute the characteristic polynomials for $d = 4$ and $d = 5$ by implementing the algorithm of Subsection 3.1. We include below only the polynomials $f_3(n)$, $f_4(n)$ and the first three terms of the polynomial $f_5(n)$ counting the number of cells determined by the hyperplane arrangement determined by n points in very general position. The full characteristic polynomials can be found in [1].

$$
\begin{aligned}
f_3(n) &= \frac{1}{1296}n^9 - \frac{1}{144}n^8 - \frac{1}{27}n^7 + \frac{61}{72}n^6 - \frac{2237}{432}n^5 + \frac{2231}{144}n^4 - \frac{14945}{648}n^3 + \frac{41}{3}n^2 \\
&\quad - \frac{13}{18}n + 1 \\
f_4(n) &= \frac{1}{7962624}n^{16} - \frac{1}{331776}n^{15} + \frac{65}{1990656}n^{14} - \frac{157}{497664}n^{13} + \frac{1315}{442368}n^{12} - \frac{923}{124416}n^{11} \\
&\quad - \frac{486709}{1990656}n^{10} + \frac{198593}{55296}n^9 - \frac{201042623}{7962624}n^8 + \frac{108860747}{995328}n^7 - \frac{103295189}{331776}n^6 \\
&\quad + \frac{73347065}{124416}n^5 - \frac{120791941}{165888}n^4 + \frac{3824591}{6912}n^3 - \frac{259219}{1152}n^2 + \frac{531}{16}n + 1 \\
f_5(n) &= \frac{1}{2985984000000}n^{25} - \frac{1}{59719680000}n^{24} + \frac{47}{119439360000}n^{23} + O(n^{22})
\end{aligned}
$$

3.3 Asymptotic behavior

Although we were unable to compute the exact number of cells $f_d(n)$ for $d > 6$, we can use our techniques to obtain at least their asymptotic growth. It is not hard to deduce that $f_d(n)$ must be a polynomial in n of degree d^2. We can however precisely determine its first $d-1$ coefficients, which, somewhat surprisingly, are exactly the same as if the $\binom{n}{d}$ hyperplanes of $\mathcal{A}(P)$ were in a general position.

▶ **Theorem 3.2.** *For every $d \geq 3$*

$$f_d(n) = \frac{1}{(d!\,)^{d+1}} \cdot n^{d^2} + \frac{d^2 - d^3}{2 \cdot (d!\,)^{d+1}} \cdot n^{d^2-1} + O(n^{d^2-2}).$$

In fact, the first $d-1$ coefficients of $\Phi_d\left(\binom{n}{d}\right)$ and $f_d(n)$ are equal.

Acknowledgments. The work on this paper was supported by grant no. 21-32817S of the Czech Science Foundation (GAČR). We thank Kolja Knauer for pointing out some references to us. The research presented in this paper has been started during the KAMAK workshop in 2022. We are grateful to the organizers of this wonderful event.

References

1 Anonymized. URL: `https://anonymous.4open.science/r/polynomial-computation-8BC2`.

2 Min Wu (https://mathoverflow.net/users/123506/min wu). Number of regions formed by n points in general position. MathOverflow. URL: `https://mathoverflow.net/q/353042` (version: 2020-02-19). URL: `https://mathoverflow.net/q/353042`.

3 Hiroshi Koizumi, Yasuhide Numata, and Akimichi Takemura. On intersection lattices of hyperplane arrangements generated by generic points. *Ann. Comb.*, 16(4):789–813, 2012. `doi:10.1007/s00026-012-0161-6`.

4 OEIS Foundation Inc. (2022). Entry A002817 in The On-Line Encyclopedia of Integer Sequences. Accessed 2022-12-03. URL: `https://oeis.org/A002817`.

5 OEIS Foundation Inc. (2022). Entry A037255 in The On-Line Encyclopedia of Integer Sequences. Accessed 2022-12-03. URL: `https://oeis.org/A037255`.

6 OEIS Foundation Inc. (2022). Entry A055503 in The On-Line Encyclopedia of Integer Sequences. Accessed 2022-12-03. URL: `https://oeis.org/A055503`.

7 OEIS Foundation Inc. (2022). The On-Line Encyclopedia of Integer Sequences. Published electronically at `http://oeis.org`.

8 Peter Orlik and Hiroaki Terao. *Arrangements of hyperplanes*, volume 300. Springer Science & Business Media, 2013.

9 Thomas Zaslavsky. *Facing up to arrangements: Face-count formulas for partitions of space by hyperplanes: Face-count formulas for partitions of space by hyperplanes*, volume 154. American Mathematical Soc., 1975.

Map Matching Queries Under Fréchet Distance on Low-Density Spanners

Kevin Buchin[1], Maike Buchin[2], Joachim Gudmundsson[3], Aleksandr Popov[*4], and Sampson Wong[3]

1 **Department of Computer Science, TU Dortmund, Germany**
kevin.buchin@tu-dortmund.de
2 **Faculty of Computer Science, Ruhr-Universität Bochum, Germany**
maike.buchin@rub.de
3 **School of Computer Science, University of Sydney, Australia**
{joachim.gudmundsson, swon7907}@sydney.edu.au
4 **Department of Mathematics and Computer Science, TU Eindhoven, The Netherlands**
a.popov@tue.nl

Abstract

Map matching is a common task when analysing GPS tracks, like vehicle trajectories—the goal is to match a recorded noisy polygonal curve to a path on the map, usually represented as a geometric graph. Fréchet distance is a commonly used metric for curves, making it a natural fit. The map-matching problem is well-studied, yet until recently no-one tackled the data structure question: preprocess a given graph so that one can query the minimum Fréchet distance between all graph paths and a polygonal curve. Recently, Gudmundsson, Seybold, and Wong have studied this problem for arbitrary query polygonal curves and c-packed graphs. In this abstract, we relax the requirement on graphs to be λ-low-density t-spanners, more closely corresponding to real-world networks. We also show how to report a path that minimises the distance efficiently rather than only return the minimal distance.

1 Introduction

Location data is ubiquitous, and analysis of that data is a common task. GPS trajectories of vehicles or people often suffer from being noisy or skipping large portions of the movement. To analyse them more precisely, one may use *map matching.* The idea is that the vehicles move on a road network, so one could snap their trajectories to a road network in a way that most closely resembles the original, prior to performing further analysis. It is a popular problem [1, 6, 11, 12, 20, 21], also under the Fréchet distance [2, 4, 5, 7, 8, 10, 18]. There is also some work on the problem under *realistic input* assumptions, which aim to exclude particular types of degenerate instances to provide stronger results, including the work by Chen, Driemel, Guibas, Nguyen, and Wenk [7], where the graph is low density and the curve is c-packed. Finally, there is work by Gudmundsson, Seybold, and Wong [9] on the data structure formulation of the problem, assuming that the graph is c-packed and not making any assumptions on the query curve.

In this paper, we solve the *map matching query problem* under realistic assumptions:

▶ **Problem 1.** Given a geometric graph $P = (V, E)$ with $n = |V| + |E|$, construct a data structure that can answer the following queries: for a polygonal curve Q in $\mathbb{R}^2$ on m points,

* Supported by the Dutch Research Council (NWO) under the project number 612.001.801.

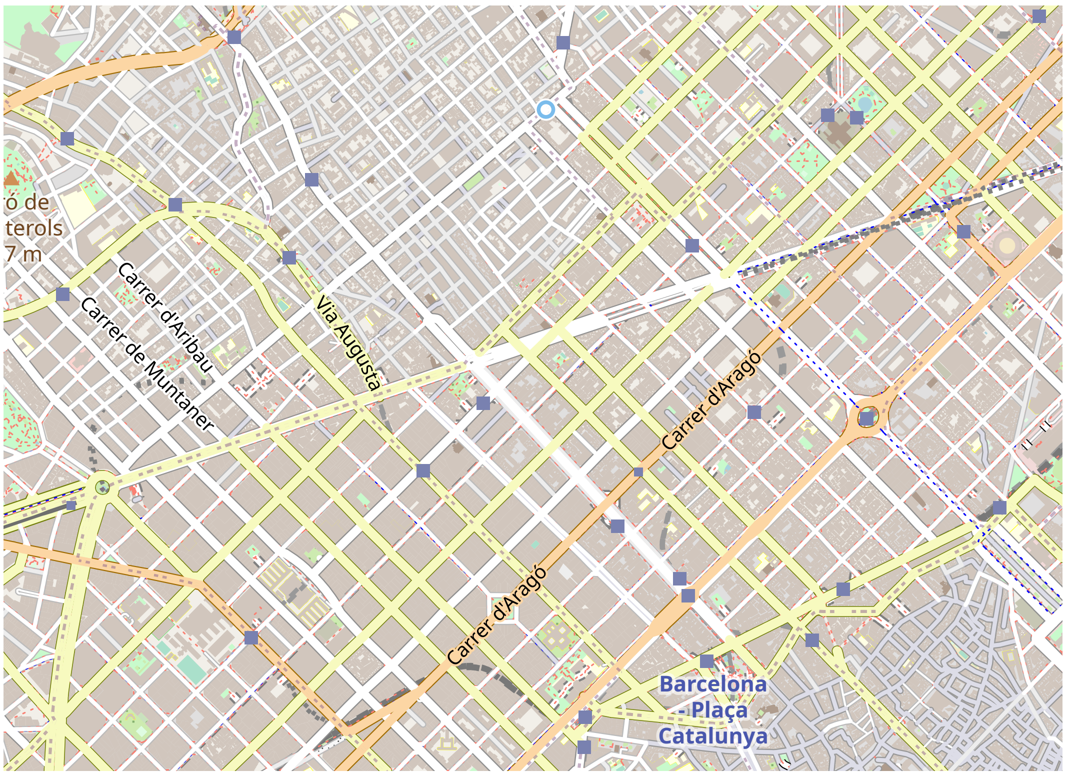


Figure 1 An example road network in the centre of Barcelona. The total road length in a disk of some radius r is closer to cr^2 than cr, so this road network is not c-packed; but it is λ-low-density and a t-spanner for bounded λ and t. Map data from OpenStreetMap [16].

1. compute $\min_\pi d_F(\pi, Q)$ and
2. report $\arg\min_\pi d_F(\pi, Q)$,

where π ranges over all paths between two vertices in P.

We use the following graph properties as the realistic input assumptions.

▶ **Definition 2.** A graph $P = (V, E)$ is λ-*low density* [17, 19] if for every disk of radius $r > 0$ in the plane, there are at most λ edges of length at least $2r$ that intersect the disk.

▶ **Definition 3.** A graph $P = (V, E)$ is called a t-*spanner* if for any two vertices $u, v \in P$, we have $d_P(u, v) \le t \cdot \|u - v\|$, where $\|\cdot\|$ is the Euclidean norm.

We present a $(1 + \varepsilon)$-approximation under the assumptions listed above. Our approach differentiates from previous work by Gudmundsson et al. [9] in two key aspects:

- we require the graph P to be λ-low density and a t-spanner, rather than c-packed, which is a more realistic assumption for a road network [3, 7, 15], while still allowing the query curve to remain unrestricted;
- we tackle the problem of reporting the path that minimises the Fréchet distance.

The relaxation of the graph assumptions is important. Consider the example map of Figure 1. It is clear that this road network is a t-spanner and λ-low-density for some low t and λ. However, it is *not* c-packed, as that would require the total length of roads be at most cr in all disks of radius r, and it is instead often much closer to cr^2. On some scale, this problem arises with many road networks, including city streets or motorways. Therefore, our assumptions make the approach significantly more applicable on real-life road networks.

In order to achieve these results, we have to use different techniques, albeit at the cost of a $\sqrt{n}$ factor replacing a polylogarithmic factor in the running time. Where the paper by Gudmundsson et al. [9] uses a semi-separated pair decomposition, we construct a hierarchy of small balanced separators and store appropriate associated data to guide the search for the optimal Fréchet distance. Combined with the changes in analysis and the capability to report the optimal curve, we get the following result.

▶ **Theorem 4.** *Suppose we are given a λ-low-density t-spanner of complexity n and a fixed $0 < \varepsilon < 1$. Let $\chi = {}^1\!/\!_{\varepsilon^2} \log {}^1\!/\!_\varepsilon$ and let $\varphi = ({}^\lambda\!/\!_{\varepsilon^3} + {}^{t^2}\!/\!_{\varepsilon^2})^2$. In time $\mathcal{O}(\lambda\chi^2 n^{5/2} \log n)$ and using $\mathcal{O}(\lambda\chi^2 n^{3/2})$ space, we can construct a data structure for Problem 1 achieving a $(1+\varepsilon)$-approximation that performs distance queries in time $\mathcal{O}(m\sqrt{n} \log mn \cdot \varphi \cdot {}^\lambda\!/\!_\varepsilon \cdot (\log^2 n + \log n \cdot \varphi + \varphi \cdot {}^\lambda\!/\!_\varepsilon))$, and answers the reporting queries for a path of length k in $\mathcal{O}({}^k\!/\!_\varepsilon)$ extra time.*

The rest of the paper is organised as follows. We first tackle the simpler problem of finding a path in the graph that most closely follows a line segment between two vertices of the graph in terms of the Fréchet distance in Section 3. In that setting, we find a 3-approximation. In Section 4, we generalise this to an arbitrary query line segment that does not have to start or end at a graph vertex, and show how to achieve a $(1+\varepsilon)$-approximation. We also describe how to report a path that corresponds to a $(1+\varepsilon)$-approximation. Finally, in Section 5, we discuss how to combine the segment queries in order to handle a polygonal curve.

2 Preliminaries

In this paper, we work with geometric graphs, that is, graphs embedded in the plane with straight-line edges. For a graph $P = (V, E)$, we denote its complexity as $n = |V| + |E|$. Depending on the context, P either means the graph or its set of vertices V. We denote the fact that a path π goes from $p \in V$ to $q \in V$ by $\pi : p \rightsquigarrow q$. We use one more graph property in this paper. An edge is *cut* by a disk if exactly one of its endpoints is inside the disk.

▶ **Definition 5.** A graph $P = (V, E)$ is *τ-lanky* [13] if for every disk of radius $r > 0$ centred at any vertex $v \in V$, there are at most τ edges of length at least r that are cut by the disk.

A τ-lanky graph has degree at most τ; and any λ-low-density graph is also λ-lanky.

The query is a polygonal curve in the plane, that is, a sequence of points in $\mathbb{R}^2$ connected with line segments. For the query curve Q, let m be the number of points in the sequence.

3 Straightest Path Queries

In this section, we present a 3-approximation to the following problem, so that for the value r that we return, we have $\min_\pi d_{\mathrm{F}}(\pi, pq) \leq r \leq 3 \cdot \min_\pi d_{\mathrm{F}}(\pi, pq)$. Recall that $n = |V| + |E|$.

▶ **Problem 6.** Given a geometric graph P, construct a data structure that can answer the following queries: for a pair of vertices $p, q \in P$, compute $\min_\pi d_{\mathrm{F}}(\pi, pq)$, where $\pi : p \rightsquigarrow q$.

In order to solve the problem efficiently, we impose an additional constraint on P: we require that P has a graph property satisfying two criteria:

1. the property is decreasing monotone, so holds on all induced subgraphs;
2. and any graph with the property admits an $\mathcal{O}(\sqrt{n})$-size separator.

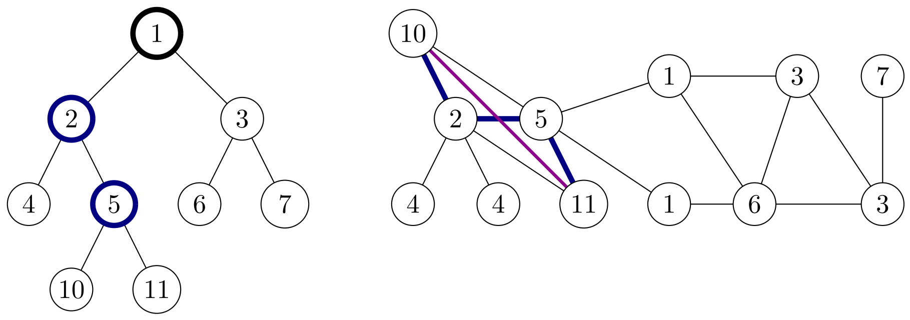


Figure 2 A representation of the hierarchy (left) for the graph (right). A query segment is shown in magenta, a possible path in blue. We check nodes 5, 2, and 1. If we pick the transit vertex in 5, then the path may be $10 \to 2 \to 5$, so we may need to go up the tree to find the next transit pair.

An example of such a property is planarity: any subgraph of a planar graph is planar, and the existence of small separators in planar graphs is a classical result [14]. However, not all road networks are planar, as most road networks include bridges and tunnels. Instead, we require that P is τ-lanky [13]. It is trivial to show that any subgraph of a τ-lanky graph is also τ-lanky; and Le and Than [13] show that a τ-lanky graph of complexity n admits a balanced separator of size $\mathcal{O}(\tau\sqrt{n})$ that can be found in $\mathcal{O}(\tau n)$ expected time.

Intuition. When constructing the data structure, we can invoke an existing algorithm by Alt, Efrat, Rote, and Wenk in order to compute the Fréchet distance between a line segment and paths in a graph [2]. Broadly, the idea is to find sufficient structure in the graph to be able to find a small set of vertices so that any path in the graph passes through at least one of these vertices; we call them *transit* vertices. Then we can precompute the distances between the optimal path and the line segment when going from any vertex of the graph to one of the transit vertices. At query time, we then only need to find an optimal transit vertex. Since we are composing two paths, the computed distance is only a 3-approximation.

More specifically, a balanced separator in a graph forms a set of transit vertices. We can compute them hierarchically and store them with the precomputed distances in a binary tree. At query time, we can efficiently find all the relevant transit vertices. See Figure 2.

Data structure. We construct a hierarchy of separators on the graph and store it as a binary tree with extra information. Each node in the tree represents both an induced subgraph of P, and a separator of that induced subgraph. Consider the node i corresponding to some induced subgraph P_i of P. Node i represents the balanced separator S_i, i.e. the subset of vertices of P_i splitting it into two subgraphs A_i and B_i. The root stores the top-level balanced separator for the entire graph. The two children of each node correspond to the subgraphs A_i and B_i. The recursion ends when the subgraphs in the leaves are of some constant size. In a leaf i, assign $S_i = P_i$. For every pair of vertices $(p, s) \in P_i \times S_i$, we store $\min_\pi d_F(\pi, ps)$, where $\pi : p \rightsquigarrow s$ in P. (Note that a path π may leave P_i.) We call all vertices in S_i *transit* vertices; and all pairs $(p, s) \in P_i \times S_i$ are called *transit pairs.*

In addition, for each vertex $v \in P$, we store the pointer to the tree node i so that $v \in S_i$. There is exactly one such node in the tree for every vertex: a vertex is either in a separator or in one of the induced subgraphs (or eventually in a leaf, treated as S_i).

Construction. We construct the hierarchy top–down, computing the separators on the induced subgraphs at every level using the result of Le and Than [13]. For each transit pair (p, s) in a node, we can compute the appropriate Fréchet distance in the original graph using the algorithm by Alt et al. [2], extended by Gudmundsson et al. [9, Lemma 13]. As we construct the separators, also store in a table the pointer for each vertex to the correct node.

Distance query. Suppose the query is to find the minimal Fréchet distance between the segment pq and some path between p and q. Initialise opt $= \infty$. First, we use the table to find the pointers to the two nodes in the tree i and j so that $p \in S_i$ and $q \in S_j$. Then we find their lowest common ancestor, call it node a. For every node a' on the path from a to the root of the tree, perform the following procedure.

Denote $D_{uv} = \min_\pi d_{\mathrm{F}}(\pi, uv)$ over all $\pi : u \rightsquigarrow v$. For the query pq, denote $D'_u = \min_{r \in pq} \|r - u\|$, so the shortest distance between u and any point on pq. For all $s \in S_{a'}$, fetch the stored D_{ps} and D_{sq} and compute D'_s. Then compute $D = \max(D_{ps}, D_{sq}) + D'_s$ and finally assign opt $= \min(\text{opt}, D)$. At the end, return opt.

Correctness. We show that we consider all the relevant transit vertices.

▶ **Lemma 7.** *For the query pq, the procedure considers a transit vertex s such that s lies on the optimal path $\pi = \arg\min_{\pi'} d_{\mathrm{F}}(\pi', pq)$, where $\pi' : p \rightsquigarrow q$.*

Proof. We consider two cases based on where the lowest common ancestor is found. Suppose that the lowest common ancestor a contains p, q, or both in the separator, i.e. $p \in S_a$ or $q \in S_a$, and so $s = p$ or $s = q$. Then π passes through s, and we test s as a transit vertex.

Now assume that S_a does not contain p or q; then p and q are separated by S_a. If the path π stays within the subgraph P_a, then it has to cross through some $s \in S_a$, which we test. If not, then it intersects some separator that separates P_a from the rest of the graph; and we test exactly all the vertices in these separators, on the path from a to the root. ◀

Omitting some further analysis, we get the result of this section.

▶ **Theorem 8.** *Given a τ-lanky graph of complexity n, we can construct the data structure for Problem 6 in time $\mathcal{O}(\tau n^{5/2} \log n)$ and using $\mathcal{O}(\tau n \sqrt{n})$ space, so the distance queries can be answered in time $\mathcal{O}(\tau \sqrt{n})$.*

4 Map Matching Segment Queries

In this section, we briefly discuss the extension to a $(1 + \varepsilon)$-approximation and reporting. Let xy be an arbitrary query segment. We can immediately use the approach of Section 3 on arbitrary query segments, with a tweak. To get a $(1 + \varepsilon)$-approximation, we construct an exponential grid around each graph vertex that is denser closer to the vertex. For every transit pair, we compute the distances for all pairs of grid points. At query time, we sample the query segment to find the right spot in the Fréchet alignment for the transit vertex.

Next we discuss the modifications that are required to also report a curve that realises $(1 + \varepsilon)$-approximate distance. First, we impose fixed coordinates for the sample points that we take on the query segment when aligning the transit vertex: they have to be located at points that are $\mathcal{O}(k/\varepsilon)$ away from a fixed point on the line containing xy for some natural k.

Next, we describe the modifications to the data structure. To handle reporting, with each transit pair, for each pair of grid points, in addition to the Fréchet distance, we also store the first vertex on the optimal path, so $p' \in P$ such that for $\pi' = \arg\min_\pi d_{\mathrm{F}}(\pi, xz)$, the path π' is of the shape $\pi' : p \to p' \rightsquigarrow s$. (Here z is the point on xy that maps to s.)

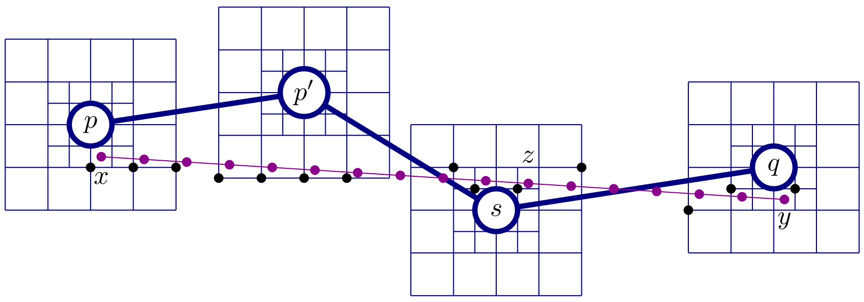


Figure 3 A query trajectory xy is shown in magenta, and the optimal path in the graph is shown in blue. We sample points on xy at regular distance and snap them to the exponential grid around the graph vertices. Once we find z on xy that maps to the transit vertex s, we can query the pair ps with (snapped) xz to find the next vertex p'.

The query proceeds as follows. Perform the distance query for xy and record the optimal transit vertex s. Find the point z among the $\mathcal{O}(1/\varepsilon)$ samples on xy that aligns with s. Query the pairs (p, s) and (s, q) with xz and zy, respectively, and retrieve the stored adjacent vertices p' and q'. Repeat until the path is reported. If p and q are both in a leaf, we proceed as if $s = q$. Suppose $s \in S_i$. If the optimal path leaves P_i, then it may be that $p' \notin P_i$, and the pair (p', s) is not stored in node i. However, then p' must be in some separator between P_i and a subgraph P_j, on the path from i to the root. Thus, we can go up until we find $p' \in S_j$, and continue the procedure for the transit pair (s, p'). See Figure 3.

The modifications do not affect construction time or space, and cost extra $\mathcal{O}(\log n + k/\varepsilon)$ time to report a path of length k. We show that we report a $(1+\varepsilon)$-approximate path.

▶ **Lemma 9.** *For query xy, if $\pi = \arg\min_{\pi^*} d_F(\pi^*, xy)$ is of the shape $\pi : p \to p' \rightsquigarrow s$, and p' is aligned under the Fréchet alignment to some $x' \in xy$, then $\pi' = \arg\min_{\pi^*} d_F(\pi^*, x'y)$ of the shape $\pi' : p' \rightsquigarrow s$ is a subpath of π.*

Proof. The sample points are placed on the line segment independently from context, so the location of sample points is the same on xy and $x'y$. We snap these sample points to the grid, and the grid does not depend on the path. Thus, we can view xy as a sequence of grid points that all possible sample points would snap to; and $x'y$ then snaps to a subsequence of those grid points. For the pairs of grid points, the distances are computed exactly; so compared to a distance query, we do not introduce extra error, and so we get a $(1+\varepsilon)$-approximation. ◀

5 Complete Map Matching Queries

The rest of the approach is the same as for Gudmundsson et al. [9]; we need to show it still can be used in our setting, but we omit the proof here. The remaining data structures aim to select a small number of points in the graph in any window of size r that are εr-close to any point in the graph. This allows us to test the possible start and end points for (sub)paths. In the end, we obtain Theorem 4.

References

1 Mohamed Ali, John Krumm, Travis Rautman, and Ankur Teredesai. ACM SIGSPATIAL GIS cup 2012. In Isabel Cruz, Craig Knoblock, Peer Kröger, Egemen Tanin, and Peter Widmayer, editors, *Proceedings of the 20th International Conference on Advances in Geo-*

graphic Information Systems (SIGSPATIAL 2012), pages 597–600, New York, NY, USA, 2012. ACM. doi:10.1145/2424321.2424426.

2 Helmut Alt, Alon Efrat, Günter Rote, and Carola Wenk. Matching planar maps. *Journal of Algorithms*, 49(2):262–283, 2003. doi:10.1016/S0196-6774(03)00085-3.

3 Boris Aronov, Kevin Buchin, Maike Buchin, Bart Jansen, Tom de Jong, Marc van Kreveld, Maarten Löffler, Jun Luo, Rodrigo I. Silveira, and Bettina Speckmann. Connect the dot: Computing feed-links for network extension. *Journal of Spatial Information Science*, 3:3–31, 2011. doi:10.5311/JOSIS.2011.3.47.

4 Sotiris Brakatsoulas, Dieter Pfoser, Randall Salas, and Carola Wenk. On map-matching vehicle tracking data. In Kjell Bratbergsengen, editor, *Proceedings of the 31st International Conference on Very Large Data Bases (VLDB 2005)*, pages 853–864, Los Angeles, CA, USA, 2005. VLDB Endowment. URL: https://dl.acm.org/doi/10.5555/1083592.1083691.

5 Erin Chambers, Brittany Terese Fasy, Yusu Wang, and Carola Wenk. Map-matching using shortest paths. *ACM Transactions on Spatial Algorithms and Systems*, 6(1):6:1–6:17, 2020. doi:10.1145/3368617.

6 Pingfu Chao, Yehong Xu, Wen Hua, and Xiaofang Zhou. A survey on map-matching algorithms. In Renata Borovica-Gajic, Jianzhong Qi, and Weiqing Wang, editors, *Databases Theory and Applications: 31st Australasian Database Conference (ADC 2020)*, volume 12008 of *Lecture Notes in Computer Science*, pages 121–133, Berlin, Germany, 2020. Springer. doi:10.1007/978-3-030-39469-1_10.

7 Daniel Chen, Anne Driemel, Leonidas J. Guibas, Andy Nguyen, and Carola Wenk. Approximate map matching with respect to the Fréchet distance. In Matthias Müller-Hannemann and Renato Werneck, editors, *Proceedings of the 2011 Workshop on Algorithm Engineering and Experiments (ALENEX)*, pages 75–83, Philadelphia, PA, USA, 2011. SIAM. doi:10.1137/1.9781611972917.8.

8 Daniel Chen, Christian Sommer, and Daniel Wolleb. Fast map matching with vertex-monotone Fréchet distance. In Matthias Müller-Hannemann and Federico Perea, editors, *21st Symposium on Algorithmic Approaches for Transportation Modelling, Optimization, and Systems (ATMOS 2021)*, volume 96 of *Open Access Series in Informatics (OASIcs)*, pages 10:1–10:20, Dagstuhl, Germany, 2021. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/OASIcs.ATMOS.2021.10.

9 Joachim Gudmundsson, Martin P. Seybold, and Sampson Wong. Map matching queries on realistic input graphs under the Fréchet distance. In Nikhil Bansal and Viswanath Nagarajan, editors, *Proceedings of the 2023 Annual ACM–SIAM Symposium on Discrete Algorithms (SODA)*, pages 1464–1492, Philadelphia, PA, USA, 2023. SIAM. doi:10.1137/1.9781611977554.ch53.

10 Joachim Gudmundsson and Michiel Smid. Fast algorithms for approximate Fréchet matching queries in geometric trees. *Computational Geometry: Theory & Applications*, 48(6):479–494, 2015. doi:10.1016/j.comgeo.2015.02.003.

11 Mahdi Hashemi and Hassan A. Karimi. A critical review of real-time map-matching algorithms: Current issues and future directions. *Computers, Environment and Urban Systems*, 48:153–165, 2014. doi:10.1016/j.compenvurbsys.2014.07.009.

12 Matej Kubicka, Arben Cela, Hugues Mounier, and Silviu-Iulian Niculescu. Comparative study and application-oriented classification of vehicular map-matching methods. *IEEE Intelligent Transportation Systems Magazine*, 10(2):150–166, 2018. doi:10.1109/MITS.2018.2806630.

13 Hung Le and Cuong Than. Greedy spanners in Euclidean spaces admit sublinear separators. In Joseph Naor and Niv Buchbinder, editors, *Proceedings of the 2022 Annual ACM–SIAM Symposium on Discrete Algorithms (SODA)*, pages 3287–3310, Philadelphia, PA, USA, 2022. SIAM. doi:10.1137/1.9781611977073.130.

14 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.

15 Stig Nordbeck. Computing distances in road networks. *Papers in Regional Science*, 12(1):207–220, 1964. doi:10.1111/j.1435-5597.1964.tb01266.x.

16 OpenStreetMap. Map data, 2023. URL: https://openstreetmap.org.

17 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.

18 Martin P. Seybold. Robust map matching for heterogeneous data via dominance decompositions. In Nitesh Chawla and Wei Wang, editors, *Proceedings of the 2017 SIAM International Conference on Data Mining (SDM)*, pages 813–821, Philadelphia, PA, USA, 2017. SIAM. doi:10.1137/1.9781611974973.91.

19 A. Frank van der Stappen. *Motion Planning Amidst Fat Obstacles.* PhD thesis, Universiteit Utrecht, 1994. URL: https://webspace.science.uu.nl/~stapp101/PhDThesis_AFvanderStappen.pdf.

20 Hong Wei, Yin Wang, George Forman, and Yanmin Zhu. Map matching: Comparison of approaches using sparse and noisy data. In Craig Knoblock, Peer Kröger, John Krumm, Markus Schneider, and Peter Widmayer, editors, *Proceedings of the 21st International Conference on Advances in Geographic Information Systems (SIGSPATIAL 2013)*, pages 444–447, New York, NY, USA, 2013. ACM. doi:10.1145/2525314.2525456.

21 Yu Zheng and Xiaofang Zhou, editors. *Computing with Spatial Trajectories.* Springer, Berlin, Germany, 2011. doi:10.1007/978-1-4614-1629-6.

Simultaneous Representation of Interval Graphs in the Sunflower Case

Ignaz Rutter[1] **and Peter Stumpf**[1]

1 **Faculty of Computer Science and Mathematics, University of Passau, Germany**
{rutter,stumpf}@fim.uni-passau.de

Abstract

A simultaneous representation of graphs $G_1, \dots, G_k$ consists of a (geometric) intersection representation R_i for each graph G_i such that each vertex v is represented by the same geometric object in each R_i for which G_i contains v. While the existence of simultaneous interval representations for $k = 2$ can be tested efficiently [8], testing it for graphs where k is part of the input is NP-complete [3]. An important special case of simultaneous representations is the *sunflower case*, where $G_i \cap G_j$ is the same *shared graph* S for each $i \neq j$. We give a polynomial-time algorithm for deciding the existence of a simultaneous interval representation for the sunflower case, even when k is not fixed. This answers an open question of Jampani and Lubiw [10].

1 Introduction

A fundamental problem in the area of intersection graphs is the *recognition* problem, where the task is to decide whether a given graph G admits a particular type of (geometric) intersection representation. The simultaneous representation problem is a generalization of the recognition problem which asks for k input graphs $G_1, \dots, G_k$ whether there exist corresponding representations $R_1, \dots, R_k$ such that each vertex v that is shared by two graphs G_i and G_j is represented by the same geometric object in R_i and in R_j. For ease of notation, we refer to $\mathcal{G} = (G_1, \dots, G_k)$ as a *simultaneous graph*, and to $\mathcal{R} = (R_1, \dots, R_k)$ as a simultaneous representation. In a *sunflower* simultaneous graph, any two graphs G_i and G_j with $i \neq j$ share the same shared graph S.

Simultaneous representations have first been studied in the context of graph embeddings where the goal is to embed each simultaneous graph without edge crossings while any shared vertices have the same coordinates in all embeddings; see [1] for a survey. The notion of simultaneous representation of general intersection graph classes was introduced by Jampani and Lubiw, who gave an $O(n^3)$-algorithms for recognizing simultaneous sunflower permutation graphs [10], proved NP-completeness for sunflower chordal graphs [10], and gave an $O(n^2 \log n)$-time algorithm for simultaneous interval graphs with $k = 2$ [8]. The running times of the algorithms were subsequently reduced to optimal linear time [2, 12].

Since then, the simultaneous representation problem has also been studied for proper and unit interval graphs [13] as well as circle graphs [6]. Bok and Jedličková showed that recognizing simultaneous non-sunflower interval graphs with k not fixed is NP-complete [3]. We show how to efficiently recognize simultaneous sunflower interval graphs when the number of input graphs is not fixed, thereby answering the open question of Jampani and Lubiw [9].

2 Preliminaries

For $n \in \mathbb{N}$ we set $[n] = \{j \in \mathbb{N} \mid 1 \leq j \leq n\}$. In this paper all graphs are simple. Theorems and lemmas marked with a star ($\star$) are proven in the full version.

39th European Workshop on Computational Geometry, Barcelona, Spain, March 29–31, 2023.
This is an extended abstract of a presentation given at EuroCG'23. 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.

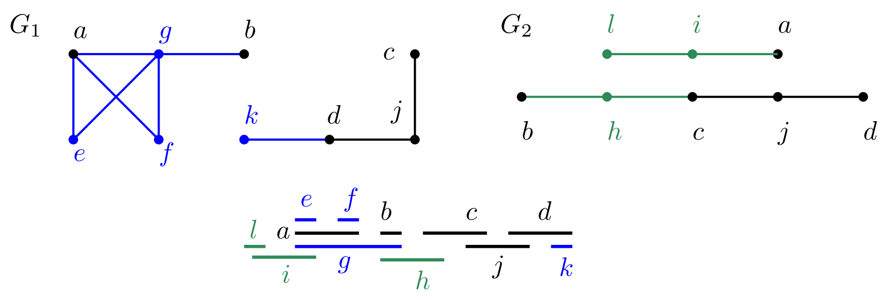


Figure 1 A simultenous graph $\mathcal{G} = (G_1, G_2)$ and a simultenous interval representation of $\mathcal{G}$

Simultaneous Interval Graphs An *interval representation* $R = \{I_v \mid v \in V\}$ of a graph $G = (V, E)$ associates with each vertex $v \in V$ an interval $I_v = [x, y] \subset \mathbb{R}$ such that for each pair of vertices $u, v \in V$ we have $I_u \cap I_v \neq \emptyset \Leftrightarrow uv \in E$; see Figure 1. In the following we only consider *sunflower graphs* $\mathcal{G} = (G_1, \dots, G_k)$ with shared graph S. Note that it is necessary that S is an induced subgraph of each input graph G_i. For a graph G and a vertex $v \in V(G)$ let $\mathcal{C}(v)$ denote its maximal cliques containing v. *A valid clique ordering* of G is a linear ordering of the maximal cliques of G such that for each $v \in V(G)$ the set $\mathcal{C}(v)$ is consecutive.

▶ Proposition 2.1 (Fulkerson and Gross [7]). A graph is an interval graph if and only if it admits a valid clique ordering.

Let $\mathcal{G} = (G_1, \dots, G_k)$ be a sunflower graph with shared graph S. Let $\mathcal{C}_i$ denote the set of maximal cliques of G_i and let $\mathcal{C} = \bigcup_{i=1}^{k} \mathcal{C}_i$. For a vertex $v \in V(G_i)$, we define $\mathcal{C}_i(v) = \{C \in \mathcal{C}_i \mid v \in C\}$ and for $v \in V(S)$, we define $\mathcal{C}(v) = \{C \in \mathcal{C} \mid v \in C\}$. We further define $\mathcal{C}(S)$ as the set of maximal cliques in the shared graph S. A *simultaneous clique ordering* of $\mathcal{G}$ is a linear ordering σ of $\mathcal{C}$ such that (i) for each $v \in V(S)$ the set $\mathcal{C}(v)$ is consecutive, and (ii) the restriction of σ to $\mathcal{C}_i$ is a clique ordering of G_i for $i \in [k]$. The following theorem provides a combinatorial description of sunflower interval graphs.

▶ **Theorem 2.2** ($\star$). *A sunflower graph $\mathcal{G}$ is a simultaneous interval graph if and only if it admits a simultaneous clique ordering.*

PQ-Trees. A *PQ-tree* T on a set M of leaves is a rooted ordered tree where each inner node is either a *P-node* or a *Q-node* [5]. The children of a P-node can be permuted arbitrarily, whereas the order of the children of a Q-node is fixed up to reversal. In this paper, we treat P-nodes with two children as Q-nodes. A PQ-tree *represents* all linear orderings of M obtained this way. Given a set M and sets $S_1, \dots, S_l \subseteq M$, a PQ-tree that represents precisely those linear orderings of M where each of the subsets $S_1, \dots, S_r$ is consecutive can be computed in $O(|M| + \sum_{i=1}^{r} |S_i|)$ time [5]. For an inner node μ of a PQ-tree, let $L(\mu)$ denote the leaves of the subtree rooted at μ. For a leaf μ let $L(\mu) = \{\mu\}$. We denote the lowest common ancestor of a set $N \subseteq V(T)$ by $\text{lca}_T(N)$.

Given a PQ-tree T on the set M and a subset $M' \subseteq M$, there exists a PQ-tree T', called the *projection* of T to M', that represents exactly the linear orders of M' that are restrictions of orderings represented by T. For two PQ-trees T_1, T_2 on the set M, there exists a PQ-tree T, called the *intersection* $T_1 \cap T_2$ representing precisely the linear orders of M represented by T_1 and T_2. Both the projection and the intersection can be computed in $O(|M|)$ time [4]. The PQ-tree representing an empty set of linear orderings is the *null-tree*.

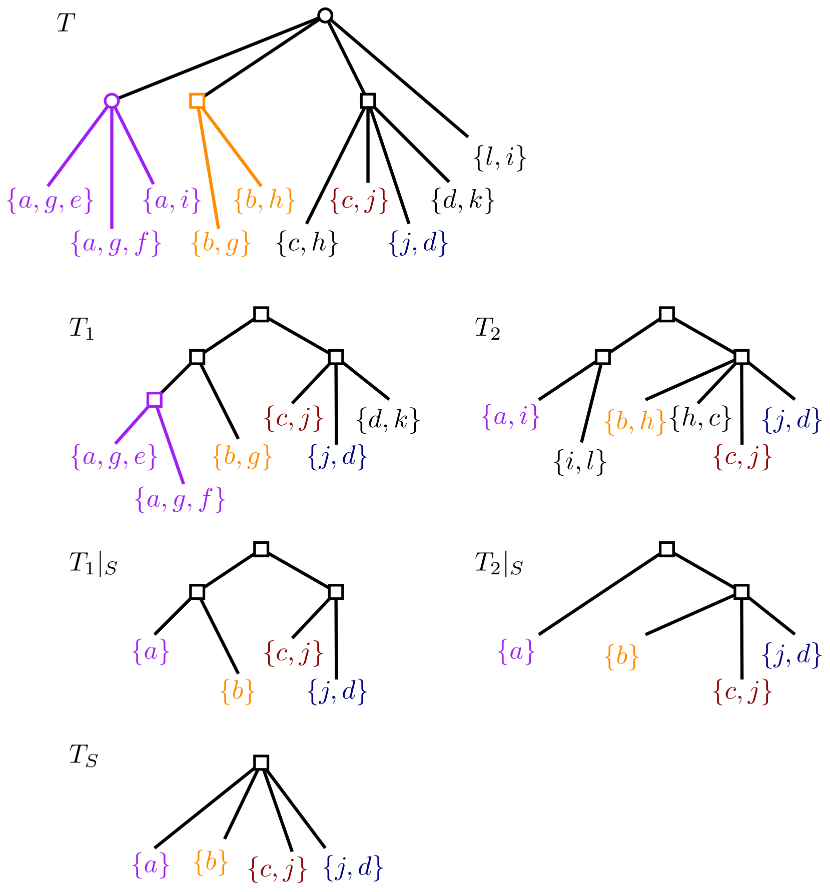


Figure 2 A PQ-tree is a rooted ordered tree where each inner node is either a P-node (circles) or a Q-node (squares). Depicted are the constructed PQ-trees for $\mathcal{G}$ from Figure 1 with circles for P-nodes and squares for Q-nodes.

3 Recognition Algorithm

We aim for a characterization of sunflower interval graphs that can be tested efficiently. Let $\mathcal{G} = (G_1, \dots, G_k)$ be a sunflower graph and let T be the PQ-tree on $\mathcal{C}$ that enforces consecutivities of each set $\mathcal{C}(v)$ with $v \in V(S)$; see Figure 2. Note that if T is the null-tree, then there exists no simultaneous clique ordering and by Theorem 2.2 there is no simultaneous interval representation of $\mathcal{G}$. For $i \in [k]$, let T_i be the PQ-tree on $\mathcal{C}_i$ that enforces consecutivities of each set $\mathcal{C}_i(v)$ with $v \in V(G_i)$. By Proposition 2.1, if any T_i is the null-tree, then G_i is not an interval graph and there is no simultaneous representation. Without loss of generality, we can assume that all linear orderings represented by T_i are represented by the projection $T_i^\star$ of T onto $\mathcal{C}_i$. Otherwise, we can replace T_i by $T_i \cap T_i^\star$ since a simultaneous clique ordering for $\mathcal{G}$ only induces orders on $\mathcal{C}_i$ that are induced by T.

We aim to describe the clique orderings for S that can be induced by $T_1, \dots, T_k$ with PQ-trees. This allows us to find a clique ordering for S that is compatible with all $T_1, \dots, T_k$. With a 2-SAT formula taking care of Q-nodes, we can then merge their orders in T, yielding a simultaneous clique ordering. In order to synchronize $T_1, \dots, T_k$, we consider the relation between consecutive sets and nodes of a PQ-tree.

▶ **Lemma 3.1** ($\star$). *Let M be a finite set and let $\{S_1, \dots, S_k\} \subseteq 2^M$ with $|S_i| \geq 2$ for $i \in [k]$.*

Let T be the PQ-tree on M obtained by making $S_1, \dots, S_k$ consecutive. Then:

(i) For each $S_i \in S$, there is either a P-node λ with $L(\lambda) = S_i$ or a Q-node μ with a consecutive subset of children $\nu_1, \dots, \nu_l$ such that $\bigcup_{i=1}^{l} L(\nu_i) = S_i$.

(ii) Let μ be a P-node and let ν be a child of μ that is not a leaf. Then if ν is a P-node, there is an S_i with $L(\nu) = S_i$. If ν is a Q-node, there is an S_i with $\bigcup_{i=1}^{l} L(\nu_i) = S_i$ for a consecutive subset of children $\nu_1, \dots, \nu_l$ of ν.

Proof. (i) follows straightforwardly from the construction [5]; see the full version.

For (ii), let ν be a P-node and suppose there is no S_i with $L(\nu) = S_i$. First observe that by (i), for any S_i with $S_i \cap L(\mu) \neq \emptyset$, we have either $L(\mu) \subseteq S_i$ or $S_i \subseteq L(\lambda)$ for some child λ of μ. This means, that after contracting the arc $\mu\nu$, no S_i can be violated. This contradicts the definition of T since T represents more linear orderings of M after the contraction of $\mu\nu$.

Next let ν be a Q-node and suppose there is no S_i with $\bigcup_{i=1}^{l} L(\nu_i) = S_i$ for any consecutive subset of children $\nu_1, \dots, \nu_l$ of ν. If ν has precisely two children, we treat it as a P-node and argue as above that there is an S_i with $L(\nu) = S_i$. Hence, assume that ν has at least three children. By (i), for any S_i with $S_i \cap L(\nu) \neq \emptyset$, we then have either $L(\mu) \subseteq S_i$ or $S_i \subseteq L(\lambda)$ for some child λ of μ. This means that after switching the label of μ from Q-node to P-node, still all represented linear orderings have all S_i consecutive. This contradicts the choice of T, since by making μ a P-node, T represents additional linear orderings. ◀

For any clique $C_S \in \mathcal{C}(S)$, observe that the set $M_i(C_S) = \bigcap_{v \in C_S}(\mathcal{C}(v) \cap \mathcal{C}_i)$ of maximal cliques of G_i containing C_S is consecutive in T_i, since the intersection of consecutive sets is itself consecutive in a linear order. With Lemma 3.1 (i), we obtain the following lemma which associates maximal cliques of the shared graph S with nodes of T_i. Note that if a set is consecutive in all linear orderings represented by a PQ-tree $T^\star$, it can be added to the set $\{S_1, \dots, S_k\}$ of consecutive sets without changing $T^\star$.

▶ **Lemma 3.2.** *For $i \in [k], C_S \in \mathcal{C}(S)$, there is either a P-node λ in T_i with $L(\lambda) = M_i(C_S)$ or a Q-node μ with a consecutive set of children $\nu_1, \dots, \nu_l$ such that $\bigcup_{j=1}^{l} L(\nu_j) = M_i(C_S)$.*

We can now construct for each T_i a PQ-tree describing the corresponding orderings of $\mathcal{C}(S)$. For $i \in [k]$, let $T_i|_S$ be the PQ-tree on $\mathcal{C}(S)$ obtained from T_i by doing for each $C_S \in \mathcal{C}(S)$ the following. Note that for distinct $C_1, C_2 \in \mathcal{C}(S)$, the sets $M_i(C_1)$ and $M_i(C_2)$ are disjoint, since the set of shared vertices in a maximal clique $C \in \mathcal{C}_i$ would otherwise be C_1 as well as C_2. If there is a P-node λ in T_i with $L(\lambda) = M_i(C_S)$, then replace the subtree of λ by a leaf C_S. Otherwise, by Lemma 3.2, there is a Q-node μ with a consecutive set of children $\nu_1, \dots, \nu_l$ such that $\bigcup_{j=1}^{l} L(\nu_j) = M_i(C_S)$. We then replace the subtrees of $\nu_1, \dots, \nu_l$ by a single leaf C_S (as a child of μ at the former position of $\nu_1, \dots, \nu_l$). Finally, let $T_i|_S$ be the projection of the resulting PQ-tree onto $\mathcal{C}(S)$.

We can now describe all orders of $\mathcal{C}(S)$ that occur in a representation of each of $G_1, \dots, G_k$ with the intersection of these PQ-trees $T_S = \bigcap_{i=1}^{k} T_i|_S$. Note that if T_S is the null-tree, then there is no simultaneous clique ordering. Recall that we consider P-nodes with only two children as Q-nodes. As the last part of our construction, we define a 2-SAT formula that ensures that the flips of the Q-nodes of various PQ-trees order cliques the same way. More precisely, we define ϕ as a 2-SAT formula with a variable x_μ for each Q-node μ in T, $T_1, \dots, T_k$, $T_1|_S, \dots, T_k|_S$ or T_S, a variable y_{C_1,C_2} for each pair of cliques $C_1, C_2 \in \mathcal{C}_i$ and a variable $<_{C_3,C_4}$ for some $i \in [k]$ and for each pair of cliques $C_3, C_4 \in \mathcal{C}(S)$. The variables express whether the order of children of a Q-node is reversed or whether e.g. C_1 is left of C_2.

For $T^\star \in \{T, T_1, \dots, T_k, T_1|_S, \dots, T_k|_S, T_S\}$ and $C_1, C_2 \in L(T^\star)$ where $\mu = \mathrm{lca}_{T^\star}(C_1, C_2)$ is a Q-node, we add $x_\mu \Leftrightarrow y_{C_1,C_2}$ to ϕ if C_1 is ordered before C_2 in $T^\star$ and $x_\mu \Leftrightarrow \neg y_{C_1,C_2}$

otherwise. The clauses ensure that two Q-nodes that decide the order of two leaves are flipped in a way such that they decide for the same order of the two leaves. Additionally, for $i \in [k]$ and $C_1, C_2 \in \mathcal{C}(S)$ such that $\mu = \mathrm{lca}_{T_i}(M_i(C_1) \cup M_i(C_2))$ is a Q-node, we add $x_\mu \Leftrightarrow y_{C_1,C_2}$ to ϕ if $M_i(C_1)$ is ordered before $M_i(C_2)$ in T_i and $x_\mu \Leftrightarrow \neg y_{C_1,C_2}$ otherwise. By Lemma 3.2 the sets $M_i(C_1)$ and $M_i(C_2)$ both are actually consecutive in T_i. These clauses are similar to the other clauses, but consider sets $M_i(C_1)$ and $M_i(C_2)$ instead of actual leaves. They ensure that each T_i is consistent with the projection $T_i|_S$. Note that a simultaneous clique ordering provides an ordering of the cliques that arranges all PQ-trees such that ϕ is satisfied. If ϕ is not satisfyable, then there is no simultaneous clique ordering. On the other hand, with this the necessary conditions are also sufficient.

▶ **Theorem 3.3** ($\star$)**.** *$(G_1, \dots, G_k)$ is a sunflower interval graph if and only if ϕ is satisfyable and neither T nor T_S is the null-tree.*

sketch of proof. If $(G_1, \dots, G_k)$ is a sunflower interval graph the requirements are necessary as discussed above. Hence, assume that neither T nor T_S are the null-tree and that ϕ has a satisfying assignment Γ. By Theorem 2.2, it suffices to find a simultaneous clique ordering. We aim to operate on T and each T_i such that the order σ of T induces the order of each T_i, thus ensuring that σ is a simultaneous clique ordering. We first flip all Q-nodes according to Γ. This ensures that any two cliques C_1, C_2 in $\mathcal{C}$ or $\mathcal{C}(S)$ are ordered as described by y_{C_1,C_2} in each PQ-tree where $\mathrm{lca}(C_1, C_2)$ is a Q-node and the sets $M_i(C_1)$, $M_i(C_2)$ are ordered as described by y_{C_1,C_2} in T_i, if $\mathrm{lca}(M_i(C_1) \cup M_i(C_2))$ is a Q-node in T_i, for $i \in [k]$.

We next have to deal with the P-nodes of T. Let μ be a P-node of T. Then, by Lemma 3.1 (ii) for each child ν of μ that is an inner node, there is a vertex $v \in S$ such that $C(v) \subseteq L(\nu)$. We choose for each child ν of μ a clique $C(\nu) \in \mathcal{C}(S)$ that contains v. We then order the children ν of μ that are inner nodes according to the order of the $C(\nu)$ given by T_S. For $i \in [k]$, we order the sets $M_i(C(\nu))$ in T_i accordingly. Note that these sets are not empty since each G_i contains a clique containing $C(\nu)$. Further note that this does not flip any Q-nodes of T_i since projection and intersection preserve Q-nodes that order two leaves of the projection set [2]. I.e., for each pair of cliques $C_1, C_2 \in \mathcal{C}(S)$ such that the order of $M_i(C_1), M_i(C_2)$ is decided by a Q-node μ in T_i, there is a Q-node in T_S deciding the order of C_1, C_2 the same way as μ orders $M_i(C_1)$, $M_i(C_2)$ (after flipping Q-nodes according to ϕ).

With this, for any P-node μ of T and any two children ν_1, ν_2 of μ that are inner nodes, each T_i orders any pair of $C_1 \in L(\nu_1) \cap \mathcal{C}_i$ and $C_2 \in L(\nu_2) \cap \mathcal{C}_i$ the same way as T. This allows us to order the children of μ simultaneously according to $T_1, \dots, T_k$ where each inner node ν is ordered as any leaf $C \in L(\nu) \cap C_i$. Note that each child of μ that is a leaf is contained in a single T_i, i.e., it can be placed solely considering the order in T_i (which ensures the correct order with regards to the children of μ that are inner nodes). Since $L(\nu) \cap C_i$ is consecutive in T_i, the choice of $C(\nu)$ does not matter. I.e., we find an order of all children of μ, which is compatible with the orders given by $T_1, \dots, T_k$. By construction, T provides a simultaneous clique ordering. ◀

This allows to recognize sunflower interval graphs in polynomial time by constructing T, T_S and ϕ and then using Theorem 3.3. If $\mathcal{G}$ is a simultaneous interval graph, we obtain a simultaneous clique ordering of $\mathcal{G}$ by following the construction in the proof of Theorem 3.3. With the construction in Theorem 2.2, we then obtain a simultaneous interval representation.

▶ **Corollary 3.4.** *Sunflower interval graphs can be recognized in polynomial time. For yes-instances simultaneous interval representations can also be provided in polynomial time.*

With a more careful construction of the 2-SAT formula and a more sophisticated implementation the running time can be improved to $O(\sum_{i=1}^{k}(|V(G_i)| + |E(G_i)|))$; see the full version.

References

1 Thomas Bläsius, Stephen G. Kobourov, and Ignaz Rutter. Simultaneous embedding of planar graphs. *CoRR*, abs/1204.5853, 2012. URL: `http://arxiv.org/abs/1204.5853`.

2 Thomas Bläsius and Ignaz Rutter. Simultaneous PQ-ordering with applications to constrained embedding problems. *ACM Trans. Algorithms*, 12(2):16:1–16:46, 2015. URL: `http://doi.acm.org/10.1145/2738054`, `doi:10.1145/2738054`.

3 Jan Bok and Nikola Jedličková. A note on simultaneous representation problem for interval and circular-arc graphs. *arXiv preprint arXiv:1811.04062*, 2018.

4 Kellogg S. Booth. *PQ Tree Algorithms*. PhD thesis, University of California, Berkeley, 1975.

5 Kellogg S. Booth and George S. Lueker. Testing for the consecutive ones property, interval graphs, and graph planarity using PQ-tree algorithms. *Journal of Computer and System Sciences*, 13(3):335–379, 1976. `doi:http://dx.doi.org/10.1016/S0022-0000(76)80045-1`.

6 Steven Chaplick, Radoslav Fulek, and Pavel Klavík. Extending partial representations of circle graphs. *J. Graph Theory*, 91(4):365–394, 2019. `doi:10.1002/jgt.22436`.

7 Delbert Fulkerson and Oliver Gross. Incidence matrices and interval graphs. *Pacific journal of mathematics*, 15(3):835–855, 1965.

8 Krishnam Raju Jampani and Anna Lubiw. Simultaneous interval graphs. In Otfried Cheong, Kyung-Yong Chwa, and Kunsoo Park, editors, *Algorithms and Computation: 21st International Symposium, ISAAC 2010, Jeju Island, Proceedings, Part I*, pages 206–217. Springer, 2010. URL: `http://dx.doi.org/10.1007/978-3-642-17517-6_20`, `doi:10.1007/978-3-642-17517-6_20`.

9 Krishnam Raju Jampani and Anna Lubiw. Simultaneous interval graphs. In *Algorithms and Computation*, pages 206–217. Springer, 2010.

10 Krishnam Raju Jampani and Anna Lubiw. The simultaneous representation problem for chordal, comparability and permutation graphs. *Journal of Graph Algorithms and Applications*, 16(2):283–315, 2012. `doi:10.7155/jgaa.00259`.

11 Norbert Korte and Rolf H Möhring. A simple linear-time algorithm to recognize interval graphs. In *International Workshop on Graph-Theoretic Concepts in Computer Science*, pages 1–16. Springer, 1986.

12 Miriam Münch, Ignaz Rutter, and Peter Stumpf. Partial and simultaneous transitive orientations via modular decompositions. In Sang Won Bae and Heejin Park, editors, *Proceedings of the 33rd International Symposium on Algorithms and Computation (ISAAC'22)*, volume 248 of *LIPIcs*, pages 51:1–51:16. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022. `doi:10.4230/LIPIcs.ISAAC.2022.51`.

13 Ignaz Rutter, Darren Strash, Peter Stumpf, and Michael Vollmer. Simultaneous representation of proper and unit interval graphs. In Michael A. Bender, Ola Svensson, and Grzegorz Herman, editors, *Proceedings of the 27th Annual European Symposium on Algorithms (ESA'19)*, volume 144 of *LIPIcs*, pages 80:1–80:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019. `doi:10.4230/LIPIcs.ESA.2019.80`.

On Computing Local Separators for Skeletonization[*]

J. Andreas Bærentzen, Rasmus E. Christensen, Emil Toftegaard Gæde, and Eva Rotenberg

Technical University of Denmark, Lyngby, Denmark
`{janba,etoga,erot}@dtu.dk`

Abstract

The notion of local separators for computing curve skeletons stems from the recent algorithm by Bærentzen and Rotenberg in [ACM Tran. Graphics'21]. Here the computation of such local separators plays an intrinsic role, with expensive computation becoming prohibitive for practical application to larger inputs.

In this work, we dive into these computations, examining and analysing in greater detail the individual steps, to clarify what bottlenecks exist for theoretical and empirical running times.

We give a simple modification to a phase of the computation, asymptotically improving the running time, and present empirical results that demonstrate the increase in practical performance.

1 Introduction

Curve skeletons are simplified, stick-like representations of shapes, that can be used for a wide variety of applications [7, 17, 4, 14, 20], and can be computed through a wide variety of approaches [8, 16, 3, 19, 13, 22, 15, 21, 1, 6, 11].

J.A. Bærentzen and E. Rotenberg propose in [2] a new algorithm for computing a curve skeleton using local separators, which we will refer to as the *Local Separator Skeletonization* algorithm, or simply LSS. The LSS algorithm seemingly generates output of high quality, while making relatively little assumptions about the input, requiring it only to be a spatially embedded graph. It works by a three phased approach, as seen in Figure 1, in which first a number of minimal local separators are computed, then a set of non-overlapping separators are selected, and finally a skeleton is extracted.

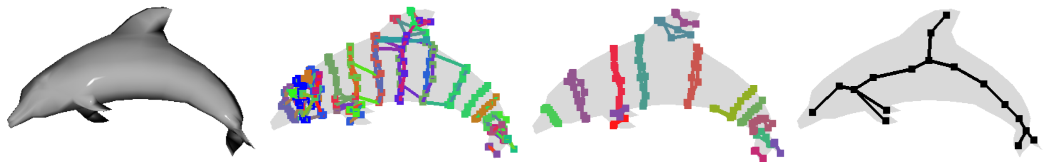


Figure 1 Visualisation of the three phases of the LSS algorithm. From left to right: A shaded render of the input, a number of computed minimal separators, a non-overlapping subset of the separators, and the resulting skeleton after extraction.

The LSS algorithm has the drawback that finding minimal local separators is computationally costly, requiring the input to be simplified in order for the running time not to be prohibitive. In this paper we give a brief analysis of the cost of computing local separators,

[*] *Emil Toftegaard Gæde* and *Eva Rotenberg* are supported by Eva Rotenberg's Carlsberg Foundation Young Researcher Fellowship CF21-0302 - "Graph Algorithms with Geometric Applications"

examine practical bottlenecks, and propose a simple modification that we show improves the running time of LSS, without altering the output.

1.1 Preliminaries

We consider a spatially embedded straight line graph, $G = \langle V, E\rangle$, with no assumptions about the origin of the graph, such as whether it is sampled from the surface of a manifold, created from a point cloud, or otherwise. If G is not connected, we simply run our algorithms on each component separately, thus we assume G to be connected.

In graph theory, an induced subgraph $G[V']$, is the graph $G' = \langle V', E'\rangle$ where E' is a subset of E that contains any edge where both endpoints are in V'. We define the closed neighbourhood of a vertex, $v \in V$, denoted $N(v)$, to be set of vertices adjacent to v and v itself. For a set of vertices, $S \subseteq V$, we define the neighbourhood as $N(S) = \bigcup N(s), s \in S$.

A *vertex separator* is a subset of vertices whose removal disconnects the graph. A minimal vertex separator is a separator where no proper subset is itself a separator. In [2], this notion is extended to *local separators*, defined as a subset of vertices, $S \subset V$, that is a vertex separator of $G[N(S)]$. Intuitively, we cannot remove a vertex from a *minimal local separator* without the remaining set ceasing to be a local separator. Formally we define a minimal local separator as $S \subset V$ s.t. S is a separator of $G[N(S)]$ and there exists no subset $S' \subset S$ s.t. S' is a separator of $G[N(S')]$. We note that S' does not need to separate the same induced subgraph as S.

2 Computing Local Separators

We recall the local separator construction algorithm of LSS, and supply a theoretical analysis of the worst case running time. With this in mind we then perform an empirical examination, detailing how the running time is distributed amongst the phases of computation in practice. Finally, we use this analysis to show our improvement to the running time of the algorithm, both asymptotically and empirically.

The algorithm for computing local separators is a heuristic algorithm that, intuitively, captures structural features of the input. It works in two phases. First, given a vertex $v \in V$, a local separator is constructed as follows (Figure 2): we maintain a candidate separator Σ, and what is called the front $F = N(\Sigma) \setminus \Sigma$. Additionally, we maintain some bounding sphere $B(\Sigma)$ that contains Σ (and possibly other vertices in V).

Initially, Σ contains v, F contains the vertices adjacent to v, and the bounding sphere has its centre at v and radius 0. We then iteratively pick, from F, the vertex closest to the centre of the bounding sphere, say s, and add it to Σ. We also remove it from F and add all neighbours of s not in Σ to F, and then update the bounding sphere to encapsulate s.

This procedure is repeated until the graph induced by F consists of more than one connected component (i.e. Σ is a local separator of $G[N(\Sigma)]$). The process is visualised in Figure 2, with (A) showing the initial configuration, (B-E) showing the iterative growing of the separator, and (F) showing the final separator after disconnecting the front.

We subsequently elaborate on shrinking a separator that has been constructed in the previous phase. First the vertices of Σ are ordered from high to low according to the distance from the centre of $B(\Sigma)$ to each vertex, after performing a Laplacian smoothing of Σ. We then perform a pass over Σ and move any vertices that are adjacent to exactly one component of F, but whose removal from Σ would not reconnect F, to that adjacent component. After such a linear scan over Σ, we restart the process (as removing vertices later in the order,

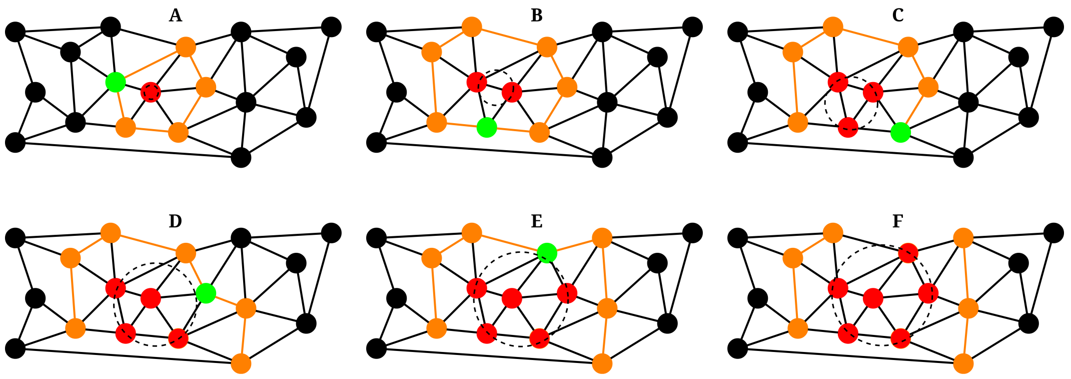


Figure 2 The process of growing a separator. Red vertices are those currently in the separator, orange vertices and edges are those currently in the front, and green vertices are the vertices of the front that are closest to the centre of the bounding sphere, which is indicated by the dotted circle. This figure is heavily inspired by Figure 6 of [2].

may make vertices earlier in the order eligible for removal), until no vertices are able to be removed (i.e. Σ is a minimal local separator).

We finally analyse the complexity of computing a local separator. Let V_F, E_F denote the vertices and edges of the F respectively, and consider each iteration of the growing phase:

1. Find closest $v \in F$ to $B(\Sigma)$	$O(\|V_F\|)$ by linear scan through V_F
2. Move v to Σ and update $B(\Sigma)$	$O(1)$
3. Add $N(v) \setminus \Sigma$ to V_F	$O(\textsc{deg}(v))$
4. Check if F is connected	$O(\|E_F\|)$ by breadth first search through F

Let Σ^* denote the result of the growing phase. We spend $|\Sigma^*|$ iterations growing, and in the worst case Σ^* and F are proportional to G s.t. $|\Sigma^*| = O(|V|), |V_F| = O(|V|), |E_F| = O(|E|)$. Thus we spend a total of $O(|V|^2)$ time on step 1, $O(|V|)$ time on step 2, $O(|E|)$ time on step 3, and $O(|V||E|)$ on step 4. The total running time spent in the growing phase, in the worst case, is then $O(|V|^2 + |E||V|)$.

To shrink the separator we compute a smoothing of positions, by considering for each vertex the positions of its neighbours, and order the vertices accordingly. Having smoothed and sorted Σ^*, the procedure then iterates over Σ^* and removes vertices until Σ^* is minimal:

5. Compute smoothing of Σ^*	$O(\textsc{deg}(s))$ for all $s \in \Sigma^*$ totalling $O(E)$
6. Order Σ^* according to smoothed positions	$O(\|\Sigma^*\| \log \|\Sigma^*\|)$
7. Iterate over $s \in \Sigma^*$ in their sorted order	$O(\|\Sigma^*\|)$ total time
▪ Check if s is incident to exactly one connected component of F.	$O(1)$
▪ If it is, remove it from Σ^* and inform $N(v) \cap \Sigma^*$	$O(\textsc{deg}(v))$, uniquely charged to v.
8. Repeat step 7 until Σ^* is minimal	$O(\|\Sigma^*\|)$ repetitions

In the worst case we remove only a constant number of vertices in each iteration of step 7, requiring $O(|\Sigma^*|)$ passes over $O(|\Sigma^*|)$ vertices, totalling $O(|\Sigma^*|^2)$ time. When removing a vertex v from Σ^* we inform the neighbours in Σ^* that they are now adjacent to the connected component of F that v was moved to. This incurs at most a cost of $O(\textsc{deg}(s))$ for all $s \in \Sigma^*$ totalling $O(E)$. The worst case running time for shrinking is then $O(|V|^2 + |E|)$, and the total time to compute a minimal local separator, including both growing and shrinking, in the worst case, is then $O(|V|^2 + |V||E|)$.

In order to further our understanding of what makes the computation slow in practice, we perform a series of measurements that also include details about how time is spent in each step.

Input	Vertices	Edges	Step 1 (s)	Step 3 & 4 (s)	Step 5 & 6 (s)	Step 7 & 8 (s)
`wsm0.25`	4946	14856	0.29	19.1	2.11	0.33
`wsm0.5`	9898	29712	1.38	89.9	10.3	1.17
`wsm1`	19803	59427	9.19	589.4	89.9	5.7
`wsv60`	6166	63136	1.50	193.6	17.8	1.2
`wsv90`	20966	233615	34.9	4372.9	478.9	14.4

Table 1 Measurements of the phases of computing a separator on a small sample of increasingly simplified meshes (`wsm*`) and voxel grids (`wsv*`) all constructed from the `wooden_statue`.

From our resulting measurements, shown in Table 1, we see that the dominating is step 4 (checking for connectivity), making it the most fitting candidate for optimisation. In Figure 3 we show the `wooden_statue` used to generate `wsm*` and `wsv*` respectively, as well as computed non-overlapping separators on both mesh and voxel grid.

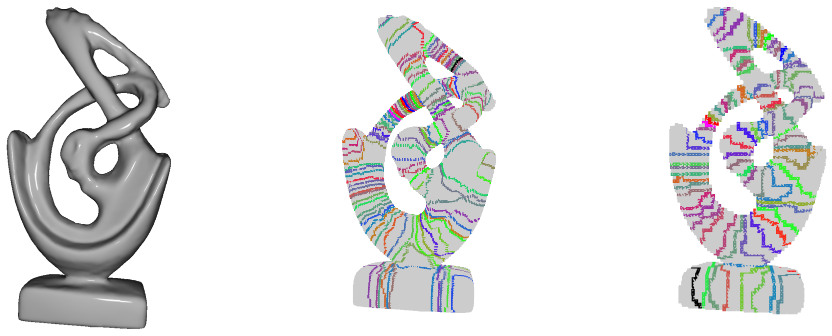


Figure 3 Left: the `wooden_statue` input used to generate `wsm*` and `wsv*`. Centre: separators computed on `wsm1`. Right: separators computed on `wsv90`.

3 Dynamic Connectivity

The simplest and most intuitive solution to the fully dynamic connectivity problem is the use of augmented Euler Tour Trees [9]. These augmented trees give $O(\log |V|)$ insertions, $O(|V| \log |V|)$ deletions and $O(\log |V|)$ connectivity queries.

In order to improve and bring into balance the update times, one can use the dynamic connectivity data structure of Holm, de Lichtenberg and Thorup [10]. It is directly cited in [2] as a suggestion, and it has been previously examined in practice [12]. We refer to this data structure as the HLT data structure.

The HLT data structure maintains a hierarchical decomposition of forests of augmented Euler Tour Trees, using the levels of the hierarchy to limit redundant searches when reconnecting. This allows for updates in amortized $O(\log^2 |V|)$ time [10].

To integrate the HLT data structure into our computations of local separators, we simply maintain the front as part of the iterative growing. The complexity of growing is then:

1. Find closest $v \in F$ to $B(\Sigma)$ $O(|V_F|)$ by linear scan through V_F

2. Move v to Σ and update $B(\Sigma)$ $O(1)$
3. Add $N(v) \setminus \Sigma$ to F $O(\text{DEG}(v) \log^2 |V_F|)$ amortized using HLT.
4. Check if F is connected $O(1)$ by maintaining the number of components in HLT

The total time becomes $O(|V|^2 + |E| \log^2 |V|)$ by the same observations as previously.

In practice, it is suggested [12] to limit the height of the hierarchy of the dynamic connectivity data structure, by not using the hierarchy, once forests reach small size. We have implemented and integrated the structure into our computations, and measured that the optimal threshold is one that is large enough to effectively eliminate the use of the hierarchy.

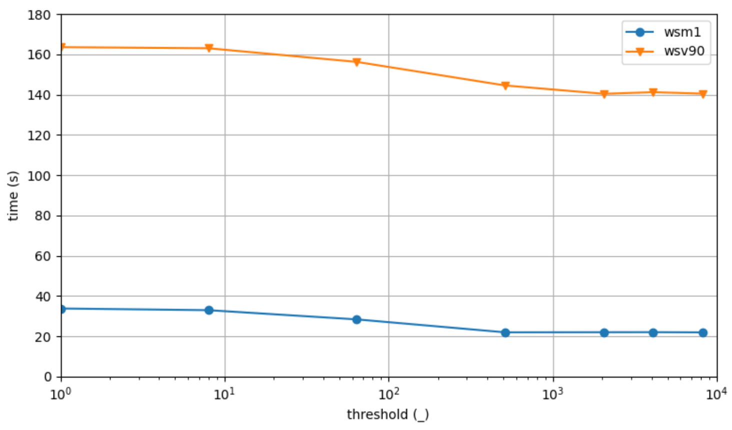


Figure 4 Time spent computing separators for a mesh (`wsm1`) and a voxel grid (`wsv90`) as a function of threshold value for the dynamic connectivity data structure.

As shown in Figure 4, the running time decreases as the threshold increases up to a certain point before flattening out. At this point, the threshold is large enough that the hierarchy consists of a single level, essentially reducing the structure to simply an augmented forest of Euler Tour Trees.

4 Empirical Results

Here we present our main result, showing how our variation compares to LSS in terms of running time. All values shown are the medians of three runs on a HP Elitebook 840 G8 with an i7 processor. The source code is publicly available through the GEL repository [5] and is compiled using `O3` optimisations. In Figure 5, we compare the old method to the one using dynamic connectivity (dyn), and find that for both meshes and voxels there is a large improvement. It is worth noting that this improvement seems even greater for voxel grids.

However, we also examine running times on a more refined scale, in order to examine how the relation between phases has changed, as seen in Table 2.

Of note is that not only has the check for connectivity been drastically reduced, especially for voxel grids, but it also seems that the bottleneck has shifted in this case towards shrinking.

In addition to measuring on `wooden_statue`, we have also measured the time spent checking connectivity on the Groningen Skeletonization Benchmark [18], the results of which can be seen in Figure 6. We note that input for which the old method exceeded half an hour was left out.

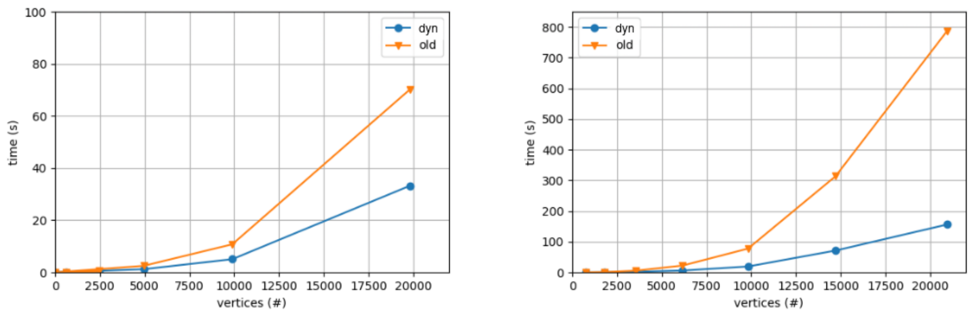


Figure 5 Running times of local separator computations without (old) and with (dyn) the use of dynamic connectivity. Left shows performance on meshes while right shows performance on voxels.

Input	Vertices	Edges	Step 1 (s)	Step 3 & 4 (s)	Step 5 & 6 (s)	Step 7 & 8 (s)
`wsm0.25`	4946	14856	0.21	4.28	1.43	0.22
`wsm0.5`	9898	29712	1.05	17.2	7.78	0.85
`wsm1`	19803	59427	8.68	109.1	80.8	5.08
`wsv60`	6166	63136	1.4	22.7	16.2	1.07
`wsv90`	20966	233615	21.9	214.7	313.9	9.75

Table 2 Measurements of the phases of computing a separator on a small sample of meshes (wsm*) and voxel grids (wsv*) all of similar structure, using the dynamic connectivity data structure.

Acknowledgements

The authors would like to thank Ivor van der Hoog for his extensive and useful feedback.

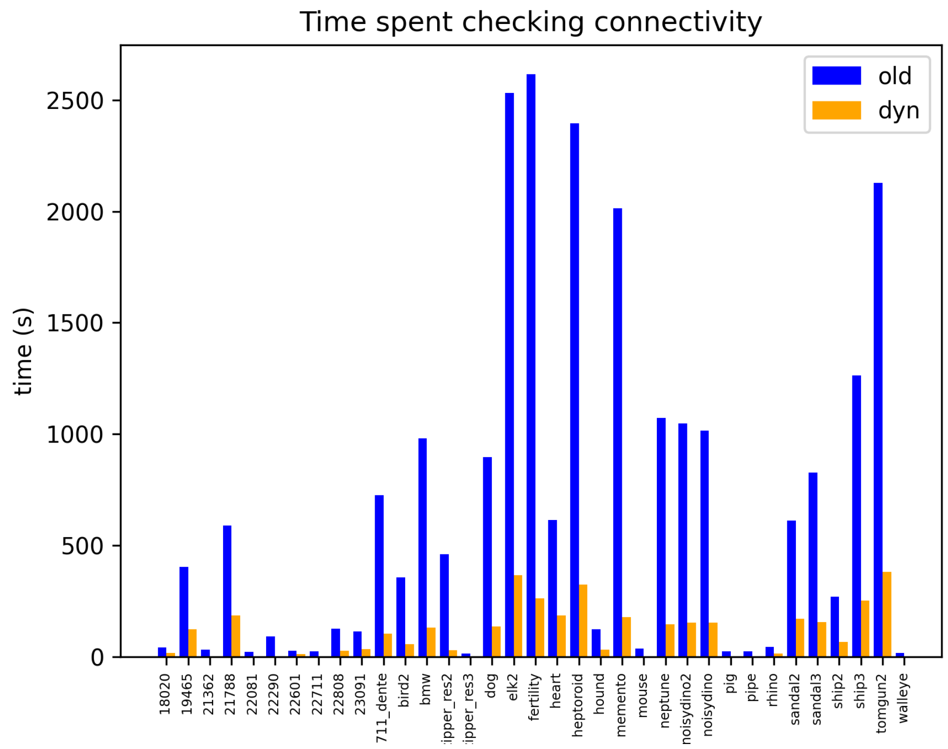


Figure 6 Time spent checking connectivity on input from the Groningen Skeletonization Benchmark.

References

1 Oscar Kin-Chung Au, Chiew-Lan Tai, Hung-Kuo Chu, Daniel Cohen-Or, and Tong-Yee Lee. Skeleton extraction by mesh contraction. *ACM Trans. Graph.*, 27(3):44, 2008. `doi:10.1145/1360612.1360643`.

2 Andreas Bærentzen and Eva Rotenberg. Skeletonization via local separators. *ACM Trans. Graph.*, 40(5):187:1–187:18, 2021. `doi:10.1145/3459233`.

3 Harry Blum. A transformation for extracting new descriptions of shape. *Models for the perception of speech and visual form*, pages 362–380, 1967.

4 Angela Brennecke and Tobias Isenberg. 3d shape matching using skeleton graphs. In Thomas Schulze, Stefan Schlechtweg, and Volkmar Hinz, editors, *Simulation und Visualisierung 2004 (SimVis 2004) 4-5 März 2004, Magdeburg*, pages 299–310. SCS Publishing House e.V., 2004. URL: `http://wwwisg.cs.uni-magdeburg.de/graphik/pub/files/Brennecke_2004_3SM.pdf`.

5 J. Andreas Bærentzen. Gel library, 2022. URL: `https://github.com/janba/GEL`.

6 Junjie Cao, Andrea Tagliasacchi, Matt Olson, Hao Zhang, and Zhixun Su. Point cloud skeletons via laplacian based contraction. In *SMI 2010, Shape Modeling International Conference, Aix en Provence, France, June 21-23 2010*, pages 187–197. IEEE Computer Society, 2010. `doi:10.1109/SMI.2010.25`.

7 Nicu D. Cornea, Deborah Silver, and Patrick Min. Curve-skeleton properties, applications, and algorithms. *IEEE Trans. Vis. Comput. Graph.*, 13(3):530–548, 2007. `doi:10.1109/TVCG.2007.1002`.

8 Tamal K. Dey and Jian Sun. Defining and computing curve-skeletons with medial geodesic function. In Alla Sheffer and Konrad Polthier, editors, *Proceedings of the Fourth Eurographics Symposium on Geometry Processing, Cagliari, Sardinia, Italy, June 26-28, 2006*, volume 256 of *ACM International Conference Proceeding Series*, pages 143–152. Eurographics Association, 2006. doi:10.2312/SGP/SGP06/143-152.

9 Monika Henzinger and Valerie King. Randomized dynamic graph algorithms with polylogarithmic time per operation. *Journal of the ACM*, 46:502–516, 01 1995. doi:10.1145/225058.225269.

10 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, 2001. doi:10.1145/502090.502095.

11 Hui Huang, Shihao Wu, Daniel Cohen-Or, Minglun Gong, Hao Zhang, Guiqing Li, and Baoquan Chen. L_1-medial skeleton of point cloud. *ACM Trans. Graph.*, 32(4):65:1–65:8, 2013. doi:10.1145/2461912.2461913.

12 Raj D. Iyer, David Karger, Hariharan S. Rahul, and Mikkel Thorup. An experimental study of poly-logarithmic fully-dynamic connectivity algorithms. *Acm Journal of Experimental Algorithmics*, 6:4, 2001. doi:10.1145/945394.945398.

13 Andrei C. Jalba, Jacek Kustra, and Alexandru C. Telea. Surface and curve skeletonization of large 3d models on the GPU. *IEEE Trans. Pattern Anal. Mach. Intell.*, 35(6):1495–1508, 2013. doi:10.1109/TPAMI.2012.212.

14 Diane Perchet, Catalin I. Fetita, and Françoise J. Prêteux. Advanced navigation tools for virtual bronchoscopy. In Edward R. Dougherty, Jaakko Astola, and Karen O. Egiazarian, editors, *Image Processing: Algorithms and Systems III, San Jose, California, USA, January 18, 2004*, volume 5298 of *SPIE Proceedings*, pages 147–158. SPIE, 2004. doi:10.1117/12.533096.

15 Dennie Reniers, Jarke J. van Wijk, and Alexandru C. Telea. Computing multiscale curve and surface skeletons of genus 0 shapes using a global importance measure. *IEEE Trans. Vis. Comput. Graph.*, 14(2):355–368, 2008. doi:10.1109/TVCG.2008.23.

16 Andrea Tagliasacchi, Ibraheem Alhashim, Matt Olson, and Hao Zhang. Mean curvature skeletons. *Comput. Graph. Forum*, 31(5):1735–1744, 2012. doi:10.1111/j.1467-8659.2012.03178.x.

17 Andrea Tagliasacchi, Thomas Delamé, Michela Spagnuolo, Nina Amenta, and Alexandru C. Telea. 3d skeletons: A state-of-the-art report. *Comput. Graph. Forum*, 35(2):573–597, 2016. doi:10.1111/cgf.12865.

18 Alexandru Telea. 3d skeletonization benchmark. https://webspace.science.uu.nl/~telea001/Shapes/SkelBenchmark, 2016. Accessed: 2022-10-31.

19 Alexandru C. Telea and Andrei C. Jalba. Computing curve skeletons from medial surfaces of 3d shapes. In Hamish A. Carr and Silvester Czanner, editors, *Theory and Practice of Computer Graphics, Rutherford, United Kingdom, 2012. Proceedings*, pages 99–106. Eurographics Association, 2012. doi:10.2312/LocalChapterEvents/TPCG/TPCG12/099-106.

20 Ming Wan, Frank Dachille, and Arie E. Kaufman. Distance-field-based skeletons for virtual navigation. In Thomas Ertl, Kenneth I. Joy, and Amitabh Varshney, editors, *12th IEEE Visualization Conference, IEEE Vis 2001, San Diego, CA, USA, October 24-26, 2001, Proceedings*, pages 239–246. IEEE Computer Society, 2001. doi:10.1109/VISUAL.2001.964517.

21 Yu-Shuen Wang and Tong-Yee Lee. Curve-skeleton extraction using iterative least squares optimization. *IEEE Trans. Vis. Comput. Graph.*, 14(4):926–936, 2008. doi:10.1109/TVCG.2008.38.

22 Yajie Yan, Kyle Sykes, Erin W. Chambers, David Letscher, and Tao Ju. Erosion thickness on medial axes of 3d shapes. *ACM Trans. Graph.*, 35(4):38:1–38:12, 2016. doi:10.1145/2897824.2925938.

Primal-Dual Cops and Robber

Minh Tuan Ha[1], **Paul Jungeblut**[2], **and Torsten Ueckerdt**[3]

1 **Karlsruhe Institute of Technology**
`uwpwm@student.kit.edu`
2 **Karlsruhe Institute of Technology**
`paul.jungeblut@kit.edu`
3 **Karlsruhe Institute of Technology**
`torsten.ueckerdt@kit.edu`

Abstract

Cops and Robber is a family of two-player games played on graphs in which one player controls a number of cops and the other player controls a robber. In alternating turns, each player moves (all) their figures. The cops try to capture the robber while the latter tries to flee indefinitely. In this paper we consider a variant of the game played on a planar graph where the robber moves between adjacent vertices while the cops move between adjacent faces. The cops capture the robber if they occupy all incident faces. We prove that a constant number of cops suffices to capture the robber on any planar graph of maximum degree Δ if and only if $\Delta \leq 4$.

Related Version *Full Version*: `https://arxiv.org/abs/2301.05514` [9]

1 Introduction

Cops and Robber is probably the most classical combinatorial pursuit-evasion game on graphs. The robber models an intruder in a network that the cops try to capture. Two players play with complete information on a fixed finite graph $G = (V, E)$. The cop player controls a set of k cops, each occupying a vertex of G (possibly several cops on the same vertex), while the robber player controls a single robber that also occupies a vertex of G. The players take alternating turns, where the cop player in his turn can decide for each cop individually whether to stay at its position or move the cop along an edge of G onto an adjacent vertex. Similarly, the robber player on her turn can leave the robber at its position or move it along an edge of G. The cop player starts by choosing starting positions for his k cops and wins the game as soon as at least one cop occupies the same vertex as the robber, i.e., when the robber is captured. The robber player, seeing the cops' positions, chooses the starting position for her robber and wins if she can avoid capture indefinitely. The least integer k for which, assuming perfect play on either side, k cops can always capture the robber, is called the *cop number* of G, usually denoted by $c(G)$.

In this paper, we introduce *Primal-Dual Cops and Robber* which is played on a plane graph G, i.e., with a fixed plane embedding. Here, the cops occupy the faces of G and can move between adjacent faces (i.e., faces that share an edge), while the robber still moves along edges between adjacent vertices of G. In this game, the robber is captured if *every* face incident to the robber's vertex is occupied by at least one cop. Analogously, we call the least integer k for which k cops can always capture the robber in the Primal-Dual Cops and Robber game the *primal-dual cop number* of G and denote it by $c^*(G)$.

An obvious lower bound for $c^*(G)$ is the maximum number of faces incident to any vertex in G: The robber can choose such a vertex as its start position and just stay there indefinitely (note that there is no *zugzwang*, i.e., no obligation to move during one's turn). In particular, if G has maximum degree $\Delta(G)$ and there exists a vertex v with $\deg(v) = \Delta(G)$, which is not a cut-vertex, then $c^*(G) \geq \Delta(G)$. E.g., $c^*(K_{2,n}) = \Delta(K_{2,n}) = n$ for any $n \geq 2$.

Our contribution. We investigate whether the primal-dual cop number $c^*(G)$ is bounded in terms of $\Delta(G)$ for all plane graphs G. The answer is 'Yes' if $\Delta(G) \leq 4$ and 'No' otherwise.

▶ **Theorem 1.1.** *Each of the following holds.*

1. *For every plane graph G with $\Delta(G) \leq 3$ we have $c^*(G) \leq 3$.*
2. *For every plane graph G with $\Delta(G) \leq 4$ we have $c^*(G) \leq 12$.*
3. *For some n-vertex plane graphs G with $\Delta(G) = 5$ we have $c^*(G) = \Omega\big(\sqrt{\log(n)}\big)$.*

Related work. Let us just briefly mention that Cops and Robber was introduced by Nowakowski and Winkler [11] and Quillot [13] for one cop and Aigner and Fromme [1] for k cops 40 years ago. Since then numerous results and variants were presented, see e.g., [2, 3]. Perhaps most similar to our new variant are the recent surrounding variant of Burgess et al. [5] with vertex-cops and the containment variant of Crytser et al. [6, 12] with edge-cops. In these variants the robber is captured if every adjacent vertex, respectively every incident edge, is occupied by a cop. The smallest number of cops that always suffices for any planar graph G is 3 in the classical variant [1], 7 in the surrounding variant [4], $7\Delta(G)$ in the containment variant [6] and 3 when both, cops and robber, move on edges [7].

2 Cops win always if the maximum degree is at most four

We start with an observation that simplifies the proofs of statements 1 and 2 in Theorem 1.1.

▶ **Observation 2.1.** *Let the robber be on a vertex u with a neighbor v of degree 1. Then the robber is never required to move to v to evade the cops.*

This is true because the set of faces required to capture the robber at v is a subset of the faces required to capture him at u. Further, his only possible moves at v are either staying there or moving back to u. As there is no zugzwang, he could just stay at u all along.

In both of the following proofs we assume that the graph contains only degree-3-vertices (respectively degree-4-vertices) and degree-1-vertices. This can always be achieved by adding leaves to vertices not yet having the correct degree.

Proof of statement 1 in Theorem 1.1. We give a winning strategy for three cops c_1, c_2, c_3 in a planar graph G with $\Delta(G) \leq 3$. First the cops choose arbitrary faces to start on. Then the robber chooses its start vertex u, which we assume to be of degree 3 by Observation 2.1 (it is trivial to capture him if all vertices have degree 1). Let $\angle_1^u, \angle_2^u, \angle_3^u$ be the three angles incident to u. We denote the face containing an angle $\angle$ by $f(\angle)$ and define for each cop c_i a *target face* f_i, $i = 1, 2, 3$. Initially we set $f_i = f(\angle_i^u)$. The goal of each cop is to reach his target face, thereby capturing the robber when all three cops arrive. If the robber moves, each cop updates his target face. Our strategy guarantees that the total distance of all three cops to their target faces decreases over time, so it reaches zero after finitely many turns.

Clearly, in every game the robber has to move at some point to avoid being captured. Assume that the robber moves from vertex u to vertex v (both of degree 3 by Observation 2.1). Without loss of generality the angles around u and v are labeled as in Figure 1 with $f_i = f(\angle_i^u)$ being the current target face of cop c_i, $i = 1, 2, 3$.

First assume that c_3 (or symmetrically c_2) has not reached his target face yet. In this case we assign the new target faces $f_1 = f(\angle_1^v)$, $f_2 = f(\angle_2^v)$ and $f_3 = f(\angle_3^v)$. Note that for $i = 1, 2$ faces $f(\angle_i^u)$ and $f(\angle_i^v)$ are adjacent, so cop c_i can keep his distance to his target face unchanged (or even decrease it) during his next turn. Further note that $f(\angle_3^u) = f(\angle_3^v)$,

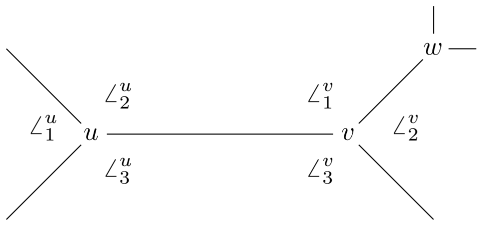


Figure 1 Labeling of the angles for a robber move from u to v (and possibly further to w).

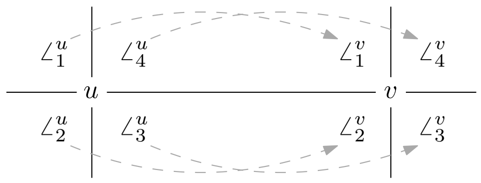


Figure 2 A vertex cop and its four accompanying face-cops moving from u to v.

so cop c_3 can even decrease his distance by one during the next turn. Thus the total distance of the three cops to their target faces decreased by at least one.

It remains the case that c_2 and c_3 have already reached their target faces (but c_1 has not, as the game would be over otherwise). In this case we move c_1 one step towards his target face $f_1 = f(\angle_1^u)$ and c_2, c_3 both to $f(\angle_2^v)$. Now its the robber's turn again. If she does not move, we assign target faces $f_i = f(\angle_i^v)$, $i = 1, 2, 3$, and the total distance decreases after the cops' next turn. If she moves back to u, we assign target faces $f_i = f(\angle_i^u)$, $i = 1, 2, 3$, and the total distance decreases after the cops' next turn. The last possibility for the robber is to move towards another neighbor w of v, see Figure 1. Then we assign $f_1 = f(\angle_1^v)$ and f_2, f_3 to be the faces containing the other two angles at w. In their next turn, c_2 and c_3 can again reach their target faces, while c_1 can decrease his distance to his target face $f(\angle_1^v)$ by one compared to the initial situation with the robber at vertex u. Again, the total distance is decreased, which concludes the proof. ◀

To prove statement 2 in Theorem 1.1, we reduce our Primal-Dual Cops and Robber to the classical Cops and Robber with cops on vertices of G and then use a result from the literature.

▶ **Lemma 2.2.** *In a plane graph G with $\Delta(G) \leq 4$, four face-cops can simulate a vertex-cop.*

Proof. Let c be a vertex-cop starting at a vertex $u \in V(G)$ with up to four incident angles $\angle_i^u$ (for $i \in \{1, 2, 3, 4\}$). We place four face-cops on the (up to) four faces $f(\angle_i^u)$. If the vertex-cop moves to an adjacent vertex v, the four face cops around it can in one step also move to faces containing the angles incident to v, see Figure 2 for the case that u and v both have degree 4. For vertices of degree less then 4 it only gets easier for the face-cops. ◀

An immediate corollary of Lemma 2.2 is that $c^*(G) \leq 4 \cdot c(G)$ for planar graphs G with $\Delta(G) \leq 4$. With $c(G) \leq 3$ for all planar graphs G [1], statement 2 in Theorem 1.1 follows.

3 Robber wins sometimes if the maximum degree is at least five

In this section we prove statement 3 in Theorem 1.1, i.e., that $c^*(G) = \Omega\big(\sqrt{\log(n)}\big)$ for some n-vertex plane graphs G with $\Delta(G) \geq 5$. We utilize a result of Nisse and Suchan [10]

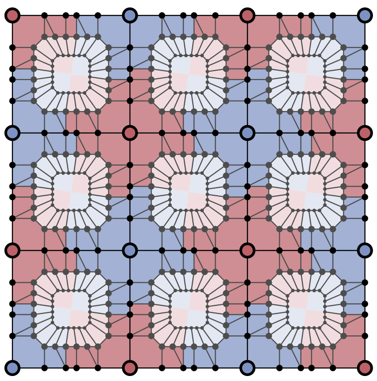


Figure 3 $G_{4,2,2}$: An $n \times n$ grid with each edge subdivided four times and two rings. Faces are colored according to their closest grid vertex. Deep and shallow faces are light and dark, respectively.

about the cop number $c_{p,q}(G)$ for a different variant of Cops and Robber for any graph G and positive integers p and q. Here (as in the classical variant) the cops and the robber are on the vertices of G. However, in each turn the cops may traverse up to p edges of G, while the robber may traverse up to q edges of G. We refer to p and q as the *velocities* of the cops and the robber, respectively.

▶ **Theorem 3.1** ([8, 10]). *Let G_n be the $n \times n$ grid graph, p be the velocity of the cops and q be the velocity of the robber. If $p < q$, then $c_{p,q}(G_n) = \Omega\big(\sqrt{\log(n)}\big)$.*

The idea to prove statement 3 in Theorem 1.1 is to construct a "grid-like" graph $G_{n,s,r}$ for positive integers n, s, r in which the robber in the primal-dual variant can move around faster than the cops. Then she can simulate the evasion strategy of the robber in the variant of Nisse and Suchan.

We start with the $n \times n$ grid graph G_n, $n \geq 3$, with a planar embedding such that the 4-faces are the inner faces. We call the vertices of G_n the *grid vertices*. Then, each edge of G_n is subdivided by $2s$ new vertices, called *subdivision vertices*, to obtain $G_{n,s}$. Two grid vertices are called *neighboring* if they are adjacent in G_n. Further, inside each inner face of $G_{n,s}$ we add r nested cycles, called *rings*, of length $12s$ each and call their vertices the *ring vertices*. Between any two consecutive rings we add a planar matching of $12s$ edges. Each inner face of $G_{n,s}$ has $8s$ subdivision vertices on its boundary and $12s$ ring vertices on its outermost ring. At last, we add (in a crossing-free way) three edges from each subdivision vertex to the outermost ring vertices in the two incident faces of $G_{n,s}$ such that two edges go to one ring, the third edge to the other ring, and every ring vertex receives exactly one such edge. Along the $2s$ vertices of each subdivision path in $G_{n,s}$ the side with two edges to the ring should always switch. Thus each inner face of $G_{n,s}$ receives $12s$ edges which are connected to the $12s$ vertices of the outermost ring such that the drawing remains planar.

Call the resulting graph $G_{n,s,r}$ and note that $\Delta(G_{n,s,r}) = 5$. See also Figure 3. We shall use a robber strategy in which she only focuses on grid vertices and moves between these through the paths of subdivision vertices, i.e., only plays on $G_{n,s}$. The purpose of the additional rings in $G_{n,s,r}$ is to slow down the cops and force them to stay close to grid and subdivision vertices, too, thereby simulating the game of Nisse and Suchan on G_n.

Formally, we call an inner face of $G_{n,s,r}$ *shallow* if it is incident to some subdivision vertex, and *deep* otherwise. Lemma 3.2 below implies that, due to the number of rings, cops should not use deep faces. Omitted proofs of statements marked with ($\star$) can be found in the full version [9].

▶ **Lemma 3.2** (⋆). *Let a_1, a_2 be two shallow faces of $G_{n,s,r}$ inside the same inner face A of G_n. If $r > 3s$, then any cop moving from a_1 to a_2 along a shortest path without leaving A uses only shallow faces.*

We have to hinder the cops from taking shortcuts through the outer face f_0 of $G_{n,s,r}$. To this end let $G'_{n,s,r}$ be a copy of $G_{n,s,r}$ with outer face f'_0. Change the outer face of $G'_{n,s,r}$ such that f'_0 is an inner face (while not changing the cyclic ordering of the edges around the vertices) and define $\overline{G}_{n,s,r}$ to be the graph obtained from gluing $G_{n,s,r}$ into face f'_0 of $G'_{n,s,r}$ and identifying corresponding vertices. The robber will always stay on vertices of $G_{n,s,r}$ and whenever a cop uses a vertex v' of $G'_{n,s,r}$ she acts as if he was on the corresponding vertex v of $G_{n,s,r}$. Without loss of generality, we can therefore assume below that the game is played on $G_{n,s,r}$ with the cops being prohibited to enter the outer face.

For a face $f \in F$, we denote by v_f the grid vertex closest to f, breaking ties arbitrarily.

▶ **Lemma 3.3** (⋆). *Let a, b be two shallow faces whose closest grid vertices v_a, v_b have distance d in G_n. If $r > 3s$, then in $G_{n,s,r}$ the robber moving from v_a to v_b needs at most $(2s+1)d$ steps, while any cop moving from a to b needs at least $3s(d-4)$ steps.*

Proof of statement 3 in Theorem 1.1. Nisse and Suchan [10] (see also [8] for the omitted proofs) describe an evasion strategy for a robber with velocity q that requires $\Omega(\sqrt{\log(n)})$ vertex-cops with velocity p to capture him in G_n, provided $q > p$; see Theorem 3.1. We describe how a robber with velocity 1 in $G_{n,s,r}$ (for sufficiently large n, s, r) can simulate this strategy against face-cops with velocity 1.

We choose $p = 15$, $q = 16$ and consider the game of Nisse and Suchan for these velocities. For their graph G_n in which the robber can win against $k = \Omega(\sqrt{\log(n)})$ vertex-cops, we then consider $G_{n,s,r}$ with $s = 16$ and $r = 3s + 1 = 49$. Now we copy the evasion strategy $\mathcal{S}$ for the robber as follows: Whenever it is the robber's turn and the face-cops occupy faces $f_1, f_2, \ldots, f_k$ in $G_{n,s,r}$, consider the corresponding situation in G_n where the vertex-cops occupy $v_{f_1}, v_{f_2}, \ldots, v_{f_k}$. Based on these positions, $\mathcal{S}$ tells the robber to go to a vertex v at distance $d \leq q = 16$ from the current position of the robber in G_n. By Lemma 3.3, the robber in $G_{n,r,s}$ can go to v in at most $(2s+1)d \leq (2 \cdot 16 + 1) \cdot 16 = 528$ turns.

In the meantime, each face-cop also makes up to 528 moves in $G_{n,r,s}$, traveling from some face a to some face b, which is interpreted in G_n as the corresponding vertex-cop traveling from v_a to v_b. For v_a and v_b to be at distance $d' \geq 16$ in G_n, by Lemma 3.2 the face-cop needs at least $3s(d'-4) \geq 3 \cdot 16 \cdot 12 = 576$ turns, which is strictly more than 528. Thus, after 528 turns, each vertex-cop made at most $p = 15$ steps in G_n, as required for strategy $\mathcal{S}$.

Hence, the robber can evade k face-cops in $G_{n,s,r}$, proving $c^*(G_{n,s,r}) > k$. Since $G_{n,s,r}$ for $s, r \in O(1)$ has $O(n^2)$ vertices, this completes the proof. ◀

4 Conclusions

Let c^*_Δ denote the largest primal-dual cop number among all plane graphs with maximum degree Δ. We have shown that $c^*_3 = 3$, $c^*_4 \leq 12$ (this bound is certainly not optimal), and $c^*_5 = \infty$, while it is easy to see that $c^*_1 = 1$, $c^*_2 = 2$, and $c^*_\Delta = \infty$ for all $\Delta > 5$. Let us remark that our proof for $\Delta = 5$ also holds for a variant of the game where the robber is already captured when one cop is on one incident face. On the other hand, our proof for $\Delta = 3$ holds verbatim to prove that three cops also suffice in a variant of the game where the graph is embedded without crossings in any other surface, which makes it is interesting to consider $\Delta = 4$ here.

Another interesting direction would be to identify classes of plane graphs with unbounded maximum degree in which $c^*(G) \leq f(\Delta(G))$ for some function f. For example, what about plane 3-trees, also known as stacked triangulations?

References

1 Martin S. Aigner and M. Fromme. A Game of Cops and Robbers. *Discrete Applied Mathematics*, 8(1):1–12, 1984. `doi:10.1016/0166-218X(84)90073-8`.

2 Anthony Bonato. *An Invitation to Pursuit-Evasion Games and Graph Theory.* American Mathematical Society, 2022.

3 Anthony Bonato and Richard J. Nowakowski. *The Game of Cops and Robbers on Graphs.* American Mathematical Society, 2011. `doi:10.1090/stml/061`.

4 Peter Bradshaw and Seyyed Aliasghar Hosseini. Surrounding Cops and Robbers on Graphs of Bounded Genus, 2019. `arXiv:1909.09916`.

5 Andrea C. Burgess, Rosalind A. Cameron, Nancy E. Clarke, Peter Danziger, Stephen Finbow, Caleb W. Jones, and David A. Pike. Cops that surround a robber. *Discrete Applied Mathematics*, 285:552–566, 2020. `doi:10.1016/j.dam.2020.06.019`.

6 Danny Crytser, Natasha Komarov, and John Mackey. Containment: A Variation of Cops and Robber. *Graphs and Combinatorics*, 36(3):591–605, 2020. `doi:10.1007/s00373-020-02140-5`.

7 Andrzej Dudek, Przemysław Gordinowicz, and Paweł Prałat. Cops and Robbers playing on edges. *Journal of Combinatorics*, 5(1):131–153, 2014. `doi:10.4310/JOC.2014.v5.n1.a6`.

8 Fedor V. Fomin, Petr A. Golovach, Jan Kratochvíl, Nicolas Nisse, and Karol Suchan. Pursuing a fast robber on a graph. *Theoretical Computer Science*, 411(7–9):1167–1181, 2010. `doi:10.1016/j.tcs.2009.12.010`.

9 Minh Tuan Ha, Paul Jungeblut, and Torsten Ueckerdt. Primal-Dual Cops and Robber, 2023. `arXiv:2301.05514`.

10 Nicolas Nisse and Karol Suchan. Fast Robber in Planar Graphs. In Hajo Broersma, Thomas Erlebach, Tom Friedetzky, and Daniel Paulusma, editors, *Graph-Theoretic Concepts in Computer Science (WG 2008)*, volume 5344 of *Lecture Notes in Computer Science*, pages 312–323, 2008. `doi:10.1007/978-3-540-92248-3_28`.

11 Richard J. Nowakowski and Peter Winkler. Vertex-to-Vertex Pursuit in a Graph. *Discrete Mathematics*, 43(2–3):235–239, 1983. `doi:10.1016/0012-365X(83)90160-7`.

12 Paweł Prałat. Containment Game Played on Random Graphs: Another Zig-Zag Theorem. *The Electronic Journal of Combinatorics*, 22(2), 2015. `doi:10.37236/4777`.

13 Alain Quilliot. *Jeux et pointes fixes sur les graphes.* PhD thesis, Université de Paris VI, 1978.

The Complexity of Recognizing Geometric Hypergraphs

Daniel Bertschinger[1], Nicolas El Maalouly[1], Linda Kleist[2], Tillmann Miltzow[3], and Simon Weber[1]

1 **Department of Computer Science, ETH Zurich**
`{daniel.bertschinger,nicolas.elmaalouly,simon.weber}@inf.ethz.ch`

2 **Department of Computer Science, TU Braunschweig**
`kleist@ibr.cs.tu-bs.de`

3 **Department of Information and Computing Sciences, Utrecht University**
`t.miltzow@uu.nl`

Abstract

As set systems, hypergraphs are omnipresent and have various representations. In a *geometric representation* of a hypergraph $H = (V, E)$, each vertex $v \in V$ is a associated with a point $p_v \in \mathbb{R}^d$ and each hyperedge $e \in E$ is associated with a connected set $s_e \subset \mathbb{R}^d$ such that $\{p_v \mid v \in V\} \cap s_e = \{p_v \mid v \in e\}$ for all $e \in E$. We say that a given hypergraph H is *representable* by some (infinite) family $\mathcal{F}$ of sets in $\mathbb{R}^d$, if there exist $P \subset \mathbb{R}^d$ and $S \subseteq \mathcal{F}$ such that (P, S) is a geometric representation of H. For a family $\mathcal{F}$, we define RECOGNITION($\mathcal{F}$) as the problem to determine if a given hypergraph is representable by $\mathcal{F}$. It is known that the RECOGNITION problem is $\exists\mathbb{R}$-hard for halfspaces in $\mathbb{R}^d$. We study the families of balls and ellipsoids in $\mathbb{R}^d$, as well as other convex sets, and show that their RECOGNITION problems are also $\exists\mathbb{R}$-complete. This means that these recognition problems are equivalent to deciding whether a multivariate system of polynomial equations with integer coefficients has a real solution.

Related Version arXiv:2302.13597

1 Introduction

As set systems, hypergraphs appear in various contexts, such as databases, clustering, and machine learning. A hypergraph can be represented in various ways. As a generalization of graphs, one can represent vertices by points and hyperedges by connected sets in $\mathbb{R}^d$ such that each set contains exactly the points of a hyperedge. It is desirable that these sets satisfy additional properties, e.g., being (strictly) convex, similar or even translates of each other.

For an introductory example, suppose we are organizing a workshop in Barcelona in 2023 and have a list of accepted talks. Clearly, each participant wants to quickly identify talks of their specific interest. In order to create a good overview, we want to find a good representation. To this end, we label each talk by several tags, e.g., `hypergraphs`, `graph drawing`, `complexity theory`, `planar graphs`, etc. Then, we create a representation, where each tag is represented by a unit disk (or another nice geometric object of our choice) containing points representing the talks that have this tag, see Figure 1 for an example. In other words, we are interested in a geometric representation of the hypergraph where the vertex set is given by the talks and the tags define the hyperedges.

In this work, we investigate the complexity of deciding whether a given hypergraph has such a geometric representation. We start with a formal definition.

Problem Definition. In a *geometric representation* of a hypergraph $H = (V, E)$, each vertex $v \in V$ is associated with a point $p_v \in \mathbb{R}^d$ and each hyperedge $e \in E$ is associated

39th European Workshop on Computational Geometry, Barcelona, Spain, March 29–31, 2023.
This is an extended abstract of a presentation given at EuroCG'23. 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.

$H = (V, E)$

$V = \{1, 2, 3, 4\}$

$E = \{\,\{1, 2, 3\}, \{1, 4\}, \{3, 4\}\,\}$

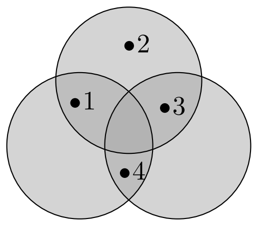


Figure 1 An abstract hypergraph and a geometric representation with unit disks.

with a connected set $s_e \subset \mathbb{R}^d$ such that $\{p_v \mid v \in V\} \cap s_e = \{p_v \mid v \in e\}$ for all $e \in E$. We say that a given hypergraph H is *representable* by some (possibly infinite) family $\mathcal{F}$ of sets in $\mathbb{R}^d$, if there exist $P \subset \mathbb{R}^d$ and $S \subseteq \mathcal{F}$ such that (P, S) is a geometric representation of H. For a family $\mathcal{F}$ of geometric objects, we define Recognition($\mathcal{F}$) as the problem to determine whether a given hypergraph is representable by $\mathcal{F}$.

Next, we give some definitions describing the geometric families studied in this work.

Bi-curved, Difference-separable, and Computable Convex Sets. We study convex sets that are bi-curved, difference-separable and computable. While the first two properties are needed for $\exists\mathbb{R}$-hardness, the last one is used to show $\exists\mathbb{R}$-membership.

Let $C \subset \mathbb{R}^d$ be a convex set. We call C *computable* if for any point $p \in \mathbb{R}^d$ we can decide on a real RAM whether p is contained in C. We say that C is *bi-curved* if there exists a unit vector $v \in \mathbb{R}^d$, such that there are two distinct tangent hyperplanes on C with normal vector v; with each of these hyperplanes intersecting C in a single point, and C being *smooth* at both of these intersection points. Informally, a convex set is bi-curved, if its boundary has two smoothly curved parts in which the tangent hyperplanes are parallel. Note that a convex, bi-curved set is necessarily bounded. As a matter of fact, any strictly convex bounded set in any dimension is bi-curved. For such sets, any unit vector v fulfills the conditions. As can be seen in Figure 2 (left), being strictly convex is not necessary for being bi-curved.

We call C *difference-separable* if for any two translates C_1, C_2 of C, there exists a hyperplane which strictly separates $C_1 \setminus C_2$ from $C_2 \setminus C_1$. Being difference-separable is fulfilled by any convex set in $\mathbb{R}^2$, see Figure 2 (middle) for an example. For a proof of this fact we refer to [28, Corollary 2.1.2.2]. However, in higher dimensions this is not the case: for a counterexample, consider two 3-cubes as in Figure 2 (right). In higher dimensions, the bi-curved and difference-separable families include the balls and ellipsoids. We are not aware of other natural geometric families with those two properties.

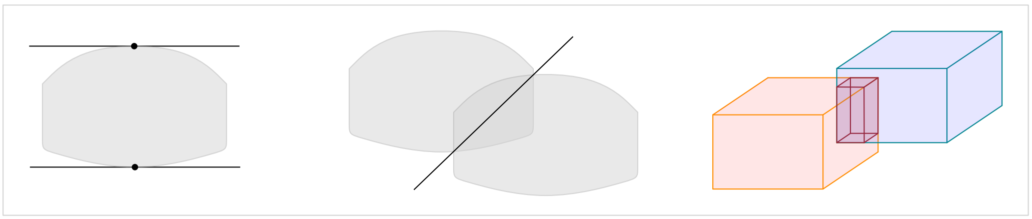


Figure 2 Left: two parallel tangent hyperplanes of a burger-like set proving its bi-curvedness. Middle: a hyperplane separating the symmetric difference of two translates of the burger-like set. Right: two cubes in $\mathbb{R}^3$ whose symmetric difference cannot be separated by a plane.

We are now ready to state our results.

Results. Our main contribution is to revive the study of recognition of geometric hypergraphs. We first consider the maybe simplest type of geometric hypergraphs, namely those that stem from halfspaces. It is known due to Tanenbaum, Goodrich, and Scheinerman [47] that the RECOGNITION problem for geometric hypergraphs of halfspaces is NP-hard, but their proof actually implies $\exists\mathbb{R}$-hardness as well. We complement this by proving $\exists\mathbb{R}$-membership to yield the following theorem:

▶ **Theorem 1** (Tanenbaum, Goodrich, Scheinerman [47]). *For every $d \geq 2$, RECOGNITION($\mathcal{F}$) is $\exists\mathbb{R}$-complete for the family $\mathcal{F}$ of halfspaces in $\mathbb{R}^d$.*

Next we consider families of objects that are translates of a given object.

▶ **Theorem 2.** *For $d \geq 2$, let $C \subseteq \mathbb{R}^d$ be a convex, bi-curved, difference-separable and computable set, and let $\mathcal{F}$ be the family of all translates of C. Then RECOGNITION($\mathcal{F}$) is $\exists\mathbb{R}$-complete.*

We note that for $d = 1$, the RECOGNITION problems of halfspaces and translates of convex sets can be solved by sorting and thus can be decided in polynomial time.

One might be under the impression that the RECOGNITION problem is $\exists\mathbb{R}$-complete for every reasonable family of geometric objects of dimension at least two. We show that is not the case by looking at translates of polygons.

▶ **Theorem 3.** *Let P be a simple polygon with integer coordinates, and $\mathcal{F}$ the family of all translates of P. Then RECOGNITION($\mathcal{F}$) is contained in NP.*

Organization. In the remainder of Section 1, we give an overview over related work. We sketch our proof techniques in Section 2. All details can be found in the full version of this paper.

Open problems. As mentioned above, we are not aware of interesting families of bi-curved and difference-separable sets in higher dimensions beyond balls and ellipsoids. The families of translates of a given polygon show the need for some curvature in order to show $\exists\mathbb{R}$-hardness. We wonder if it is sufficient for $\exists\mathbb{R}$-hardness to assume curvature at only one boundary part instead of two opposite ones. Another open question is to consider families that include rotated copies or homothetic copies of a fixed geometric object. Allowing for rotation, it is conceivable that $\exists\mathbb{R}$-hardness even holds for polygons.

1.1 Related work

In this section we give a concise overview over related work on the complexity class $\exists\mathbb{R}$, geometric intersection graphs, and on other set systems related to hypergraphs.

The Existential Theory of the Reals. The complexity class $\exists\mathbb{R}$ (pronounced as 'ER' or 'exists R') is defined via its canonical complete problem ETR (short for *Existential Theory of the Reals*) and contains all problems that polynomial-time many-one reduce to it. In an instance of ETR, we are given a sentence of the form

$$\exists x_1, \ldots, x_n \in \mathbb{R} : \varphi(x_1, \ldots, x_n),$$

where φ is a well-formed and quantifier-free formula consisting of polynomial equations and inequalities in the variables and the logical connectives $\{\wedge, \vee, \neg\}$. The goal is to decide whether this sentence is true.

The complexity class $\exists\mathbb{R}$ gains its importance from its numerous influential complete problems. Important $\exists\mathbb{R}$-completeness results include the realizability of abstract order types [32, 44], geometric linkages [37], and the recognition of geometric intersection graphs, as further discussed below. More results concern graph drawing [17, 18, 27, 38], the Hausdorff distance [23], polytopes [16, 35], Nash-equilibria [8, 10, 11, 20, 40], training neural networks [4, 9], matrix factorization [14, 41, 42, 43, 48], continuous constraint satisfaction problems [31], geometric packing [5], the art gallery problem [2], and covering polygons with convex polygons [1].

Geometric Hypergraphs Many aspects of hypergraphs with geometric representations have been studied. Hypergraphs represented by touching polygons in $\mathbb{R}^3$ have been studied by Evans et al. [19]. Bounds on the number of hyperedges in hypergraphs representable by homothets of a fixed convex set S have been established by Axenovich and Ueckerdt [7]. Smorodinsky studied the chromatic number and the complexity of coloring of hypergraphs represented by various types of sets in the plane [45]. Dey and Pach [15] generalize many extremal properties of geometric graphs to hypergraphs where the hyperedges are induced simplices of some point set in $\mathbb{R}^d$. Haussler and Welzl [21] defined ϵ-nets, subsets of vertices of hypergraphs called range spaces with nice properties. Such ϵ-nets of geometric hypergraphs have been studied quite intensely [6, 29, 33, 34].

While there are many structural results, we are not aware of any research into the complexity of recognizing hypergraphs given by geometric representations, other than the recognition of embeddability of simplicial complexes, as we will discuss in the next paragraph.

Other Representations of Hypergraphs. Hypergraphs are in close relation with abstract simplicial complexes. In particular, an abstract simplicial complex (complex for short) is a set system that is closed under taking subsets. A k-complex is a complex in which the maximum size of a set is k. In a geometric representation of an abstract simplicial complex $H = (V, E)$ each ℓ-set of E is represented by a ℓ-simplex such that two simplices of any two sets intersect exactly in the simplex defined by their intersection (and are disjoint in case of an empty intersection). Note that 1-complexes are graphs and hence deciding the representability in the plane corresponds to graph planarity (which is in P). In stark contrast, Abrahamsen, Kleist and Miltzow recently showed that deciding whether a 2-complex has a geometric embedding in $\mathbb{R}^3$ is $\exists\mathbb{R}$-complete [3]; they also prove hardness for other dimensions.

Recognizing Geometric Intersection Graphs. Given a set of geometric objects, its intersection graph has a vertex for each object, and an edge between any two intersecting objects. The complexity of recognizing geometric intersection graphs has been studied for various geometric objects. We summarize these results in Figure 3. While intersection graphs of circle chords (Spinnrad [46]), unit intervals (Looges and Olariu [26]) and intervals (Booth and Lueker [12]) can be recognized in polynomial time, recognizing string graphs (Schaefer and Sedgwick [39]) is NP-complete. In contrast, $\exists\mathbb{R}$-completeness of recognizing intersection graphs has been proved for (unit) disks by McDiarmid and Müller [30], convex sets by Schaefer [36], downward rays by Cardinal et al. [13], outer segments by Cardinal et al. [13], unit segments by Hoffmann et al. [22], segments by Kratochvíl and Matoušek [25], k-polylines by Hoffmann et al. [22], and unit balls by Kang and Müller [24].

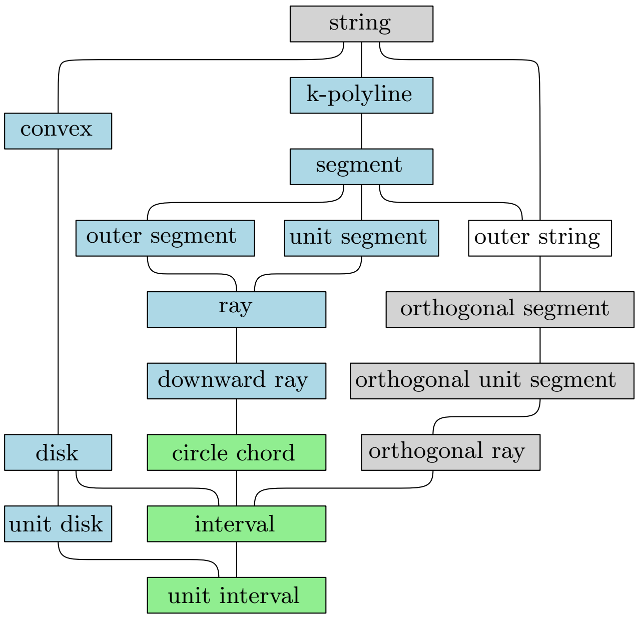


Figure 3 Containment relations of geometric intersection graphs. Recognition of a green class is in P, of a grey class is NP-complete, of a blue class is ∃ℝ-complete, and of a white class is unknown.

The existing research landscape indicates that recognition problems of intersection graphs are ∃ℝ-complete in case that the family of objects satisfy two conditions: Firstly, they need to be "geometrically solid", i.e., not strings. Secondly, some non-linearity must be present by either allowing rotations, or by the objects having some curvature. Our results indicate that this general intuition might translate to the recognition of geometric hypergraphs.

2 Proof Techniques

Membership We prove containment in ∃ℝ and NP using standard arguments, providing witnesses and verification algorithms. For Theorem 2, we simply give translation vectors and coordinates for all translates of C and points. To verify this certificate, we use that the convex set C is computable. For Theorem 3, the proof uses a similar argument to the one used to show that the problem of packing translates of polygons inside a polygon is in NP [5]: we triangulate the polygon P, and the NP-certificate consists of the triangles of each polygon a point p is contained in, see Figure 4. Verifying this certificate reduces to linear programming.

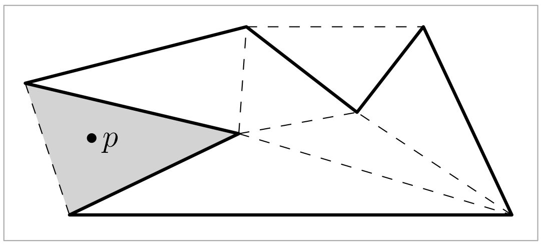


Figure 4 The polygon P, its triangulation, and the triangle that p is contained.

Hardness We prove the hardness parts of Theorems 1 and 2 by reduction from stretchability of simple pseudohyperplane arrangements. The hypergraph we build from the given arrangement differs from the one built in the proof of Theorem 1 given in [47], because we it can also be used for the proof of Theorem 2 and thus we have a single construction which works nicely for both theorems. Given a simple pseudohyperplane arrangement $\mathcal{A}$, we construct a hypergraph H as follows: We double each pseudohyperplane by giving it a parallel *twin*. In this arrangement, we place a point in every d-dimensional cell. These points represent the vertices of H. Every pseudohyperplane ℓ then defines a hyperedge, which contains all of the points on the same side of ℓ as its twin pseudohyperplane. The points between the two pseudohyperplanes are thus contained in both hyperedges. See Figure 5 for an illustration of this construction.

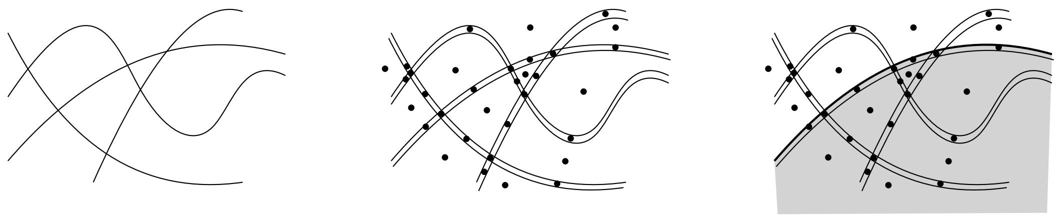


Figure 5 Illustration of the reduction used in the proofs of Theorems 1 and 2. Construction of the hypergraph H from a simple pseudohyperplane arrangement $\mathcal{A}$.

Because this construction can also be performed on a simple hyperplane arrangement, it is straightforward to prove that if $\mathcal{A}$ is stretchable, H can be represented by halfspaces. Conversely, we show that the hyperplanes bounding the halfspaces in a representation of H must be a stretching of $\mathcal{A}$.

For Theorem 2, bi-curvedness of a set C implies that locally, C can approximate any halfspace with normal vector close to v as in the definition of bi-curved. Since the given pseudohyperplane arrangement $\mathcal{A}$ is simple, adding some small amount of curvature to the hyperplanes in a stretching of $\mathcal{A}$ does not change the combinatorical structure of the arrangement. This allows us to prove that stretchability of $\mathcal{A}$ implies representability of H by translates of C. To prove that representability of H by translates of C implies stretchablity of $\mathcal{A}$, we use that the set C is difference-separable. Given two translates of C representing a pair of twin hyperedges, the hyperplane guaranteed by separability of C is used as the stretched hyperplane of the corresponding pseudohyperplane. The proof that this constructed hyperplane arrangement has the same combinatorical structure as $\mathcal{A}$ works similarly to the proof of Theorem 1.

References

1 Mikkel Abrahamsen. Covering Polygons is Even Harder. In Nisheeth K. Vishnoi, editor, *2021 IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 375–386, 2022. doi:10.1109/FOCS52979.2021.00045.

2 Mikkel Abrahamsen, Anna Adamaszek, and Tillmann Miltzow. The Art Gallery Problem is ∃ℝ-complete. In *STOC 2018: Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, pages 65–73, 2018. doi:10.1145/3188745.3188868.

3 Mikkel Abrahamsen, Linda Kleist, and Tillmann Miltzow. Geometric embeddability of complexes is ∃r-complete. *CoRR*, abs/2108.02585, 2021. URL: https://arxiv.org/abs/2108.02585, arXiv:2108.02585.

4 Mikkel Abrahamsen, Linda Kleist, and Tillmann Miltzow. Training Neural Networks is ER-complete. In Marc A. Ranzato, Alina Beygelzimer, K. Nguyen, Percy Liang, Jennifer W. Vaughan, and Yann Dauphin, editors, *Advances in Neural Information Processing Systems (NeurIPS 2021)*, volume 34, 2021.

5 Mikkel Abrahamsen, Tillmann Miltzow, and Nadja Seiferth. Framework for ER-Completeness of Two-Dimensional Packing Problems. In *2020 IEEE 61st Annual Symposium on Foundations of Computer Science (FOCS)*, pages 1014–1021, 2020. `doi:10.1109/FOCS46700.2020.00098`.

6 Boris Aronov, Esther Ezra, and Micha Sharir. Small-size ϵ-nets for axis-parallel rectangles and boxes. *SIAM Journal on Computing*, 39(7):3248–3282, 2010. `doi:10.1137/090762968`.

7 Maria Axenovich and Torsten Ueckerdt. Density of range capturing hypergraphs. *Journal of Computational Geometry*, 7(1), 2016. `doi:10.20382/jocg.v7i1a1`.

8 Marie L. T. Berthelsen and Kristoffer A. Hansen. On the Computational Complexity of Decision Problems About Multi-player Nash Equilibria. In Dimitris Fotakis and Evangelos Markakis, editors, *International Symposium on Algorithmic Game Theory*, volume 11801 of *Lecture Notes in Computer Science*, pages 153–167, 2019. `doi:10.1007/978-3-030-30473-7_11`.

9 Daniel Bertschinger, Christoph Hertrich, Paul Jungeblut, Tillmann Miltzow, and Simon Weber. Training fully connected neural networks is $\exists\mathbb{R}$-complete. *arXiv preprint arXiv:2204.01368*, 2022.

10 Vittorio Bilò and Marios Mavronicolas. A Catalog of EXISTS-R-Complete Decision Problems About Nash Equilibria in Multi-Player Games. In Nicolas Ollinger and Heribert Vollmer, editors, *33rd Symposium on Theoretical Aspects of Computer Science (STACS 2016)*, Leibniz International Proceedings in Informatics (LIPIcs), pages 17:1–17:13, 2016. `doi:10.4230/LIPIcs.STACS.2016.17`.

11 Vittorio Bilò and Marios Mavronicolas. Existential-R-Complete Decision Problems about Symmetric Nash Equilibria in Symmetric Multi-Player Games. In Vollmer Heribert and Brigitte Vallée, editors, *34th Symposium on Theoretical Aspects of Computer Science (STACS 2017)*, volume 66 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 13:1–13:14, 2017. `doi:10.4230/LIPIcs.STACS.2017.13`.

12 Kellogg S. Booth and George S. Lueker. Testing for the consecutive ones property, interval graphs, and graph planarity using pq-tree algorithms. *Journal of Computer and System Sciences*, 13(3):335–379, 1976. `doi:10.1016/S0022-0000(76)80045-1`.

13 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):273–294, 2018. `doi:10.7155/jgaa.00470`.

14 Dmitry Chistikov, Stefan Kiefer, Ines Marusic, Mahsa Shirmohammadi, and James Worrell. On Restricted Nonnegative Matrix Factorization. In Ioannis Chatzigiannakis, Michael Mitzenmacher, Yuval Rabani, and Davide Sangiorgi, editors, *43rd International Colloquium on Automata, Languages, and Programming (ICALP 2016)*, volume 55 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 103:1–103:14, 2016. `doi:10.4230/LIPIcs.ICALP.2016.103`.

15 T. K. Dey and J. Pach. Extremal problems for geometric hypergraphs. *Discrete & Computational Geometry*, 19(4):473–484, Apr 1998. `doi:10.1007/PL00009365`.

16 Michael G. Dobbins, Andreas Holmsen, and Tillmann Miltzow. A Universality Theorem for Nested Polytopes. arXiv preprint, 2019. `arXiv:1908.02213`.

17 Michael G. Dobbins, Linda Kleist, Tillmann Miltzow, and Paweł Rzążewski. Completeness for the complexity class $\forall\exists\mathbb{R}$ and area-universality. *Discrete & Computational Geometry (DCG)*, 2022. `doi:10.1007/s00454-022-00381-0`.

18 Jeff Erickson. Optimal Curve Straightening is ∃ℝ-Complete. arXiv preprint, 2019. `arXiv:arXiv:1908.09400`.

19 William Evans, Paweł Rzążewski, Noushin Saeedi, Chan-Su Shin, and Alexander Wolff. Representing graphs and hypergraphs by touching polygons in 3d. In Daniel Archambault and Csaba D. Tóth, editors, *Graph Drawing and Network Visualization*, pages 18–32, Cham, 2019. Springer International Publishing. `doi:10.1007/978-3-030-35802-0_2`.

20 Jugal Garg, Ruta Mehta, Vijay V. Vazirani, and Sadra Yazdanbod. ∃ℝ-Completeness for Decision Versions of Multi-Player (Symmetric) Nash Equilibria. *ACM Transactions on Economics and Computation*, 6(1):1:1–1:23, 2018. `doi:10.1145/3175494`.

21 David Haussler and Emo Welzl. Epsilon-nets and simplex range queries. In *Proceedings of the second annual symposium on Computational geometry*, pages 61–71, 1986.

22 Michael Hoffmann, Tillmann Miltzow, Simon Weber, and Lasse Wulf. Recognition of unit segment graphs is ∃ℝ-complete. Unpublished, in preparation.

23 Paul Jungeblut, Linda Kleist, and Tillmann Miltzow. The complexity of the hausdorff distance. In Xavier Goaoc and Michael Kerber, editors, *38th International Symposium on Computational Geometry, SoCG 2022, June 7-10, 2022, Berlin, Germany*, volume 224 of *LIPIcs*, pages 48:1–48:17. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022. `doi:10.4230/LIPIcs.SoCG.2022.48`.

24 Ross J. Kang and Tobias Müller. Sphere and dot product representations of graphs. *Discrete & Computational Geometry*, 47(3):548–568, Apr 2012. `doi:10.1007/s00454-012-9394-8`.

25 Jan Kratochvíl and Jiří Matoušek. Intersection Graphs of Segments. *Journal of Combinatorial Theory, Series B*, 62(2):289–315, 1994. `doi:10.1006/jctb.1994.1071`.

26 Peter J. Looges and Stephan Olariu. Optimal greedy algorithms for indifference graphs. *Computers & Mathematics with Applications*, 25(7):15–25, 1993. `doi:10.1016/0898-1221(93)90308-I`.

27 Anna Lubiw, Tillmann Miltzow, and Debajyoti Mondal. The Complexity of Drawing a Graph in a Polygonal Region. In Therese Biedl and Andreas Kerren, editors, *GD 2018: Graph Drawing and Network Visualization*, volume 11282 of *Lecture Notes in Computer Science*, pages 387–401, 2018. `doi:10.1007/978-3-030-04414-5_28`.

28 Lihong Ma. *Bisectors and Voronoi Diagrams for Convex Distance Functions.* PhD thesis, FernUniversität Hagen, 2000.

29 Jiří Matoušek, Raimund Seidel, and E. Welzl. How to net a lot with little: Small ϵ-nets for disks and halfspaces. In *Proceedings of the Sixth Annual Symposium on Computational Geometry*, SCG '90, page 16–22, New York, NY, USA, 1990. Association for Computing Machinery. `doi:10.1145/98524.98530`.

30 Colin McDiarmid and Tobias Müller. Integer realizations of disk and segment graphs. *Journal of Combinatorial Theory, Series B*, 103(1):114–143, 2013.

31 Tillmann Miltzow and Reinier F. Schmiermann. On Classifying Continuous Constraint Satisfaction Problems. In Nisheeth K. Vishnoi, editor, *2021 IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 781–791, 2022. `doi:10.1109/FOCS52979.2021.00081`.

32 Nikolai E. Mnëv. The Universality Theorems on the Classification Problem of Configuration Varieties and Convex Polytopes Varieties. In Oleg Y. Viro and Anatoly M Vershik, editors, *Topology and Geometry — Rohlin Seminar*, volume 1346 of *Lecture Notes in Mathematics*, pages 527–543. Springer, 1988. `doi:10.1007/BFb0082792`.

33 János Pach and Gábor Tardos. Tight lower bounds for the size of epsilon-nets. *Journal of the American Mathematical Society*, 26(3):645–658, 2013.

34 János Pach and Gerhard Woeginger. Some new bounds for epsilon-nets. In *Proceedings of the Sixth Annual Symposium on Computational Geometry*, SCG '90, page 10–15, New York, NY, USA, 1990. Association for Computing Machinery. `doi:10.1145/98524.98529`.

35 Jürgen Richter-Gebert and Günter M. Ziegler. Realization Spaces of 4-Polytopes are Universal. *Bulletin of the American Mathematical Society*, 32(4):403–412, 1995. `doi:10.1090/S0273-0979-1995-00604-X`.

36 Marcus Schaefer. Complexity of some geometric and topological problems. In David Eppstein and Emden R. Gansner, editors, *Graph Drawing*, pages 334–344, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.

37 Marcus Schaefer. *Realizability of Graphs and Linkages*, pages 461–482. Thirty Essays on Geometric Graph Theory. Springer, 2013. `doi:10.1007/978-1-4614-0110-0_24`.

38 Marcus Schaefer. Complexity of Geometric k-Planarity for Fixed k. *Journal of Graph Algorithms and Applications*, 25(1):29–41, 2021. `doi:10.7155/jgaa.00548`.

39 Marcus Schaefer, Eric Sedgwick, and Daniel Štefankovič. Recognizing string graphs in np. *Journal of Computer and System Sciences*, 67(2):365–380, 2003. Also STOC 2002.

40 Marcus Schaefer and Daniel Štefankovič. Fixed Points, Nash Equilibria, and the Existential Theory of the Reals. *Theory of Computing Systems*, 60:172–193, 2017. `doi:10.1007/s00224-015-9662-0`.

41 Marcus Schaefer and Daniel Štefankovič. The Complexity of Tensor Rank. *Theory of Computing Systems*, 62(5):1161–1174, 2018. `doi:10.1007/s00224-017-9800-y`.

42 Yaroslav Shitov. A Universality Theorem for Nonnegative Matrix Factorizations. arXiv preprint, 2016. `arXiv:1606.09068`.

43 Yaroslav Shitov. The complexity of positive semidefinite matrix factorization. *SIAM Journal on Optimization*, 27(3):1898–1909, 2017. `doi:10.1137/16M1080616`.

44 Peter W. Shor. Stretchability of Pseudolines is NP-Hard. In Peter Gritzmann and Bernd Sturmfels, editors, *Applied Geometry And Discrete Mathematics*, volume 4 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, pages 531–554, 1991. `doi:10.1090/dimacs/004/41`.

45 Shakhar Smorodinsky. On the chromatic number of geometric hypergraphs. *SIAM Journal of Discrete Mathematics*, 21(3):676–687, 2007. `doi:10.1137/050642368`.

46 Jeremy Spinrad. Recognition of circle graphs. *Journal of Algorithms*, 16(2):264–282, 1994.

47 Paul J. Tanenbaum, Michael T. Goodrich, and Edward R. Scheinerman. Characterization and recognition of point-halfspace and related orders. In Roberto Tamassia and Ioannis G. Tollis, editors, *Graph Drawing*, pages 234–245, Berlin, Heidelberg, 1995. Springer Berlin Heidelberg.

48 Levent Tuncel, Stephen Vavasis, and Jingye Xu. Computational complexity of decomposing a symmetric matrix as a sum of positive semidefinite and diagonal matrices. *arXiv preprint arXiv:2209.05678*, 2022.

Packing Fréchet Balls

Peyman Afshani[1] and Aniket Basu Roy[2]

1 **Aarhus University**
`peyman@cs.au.dk`
2 **Aarhus University**
`aniket@cs.au.dk`

Abstract

Given a collecton of polygonal chains we define a ball for every chain under the Fréchet and Hausdorff metric and study the intersection graphs of these balls. We show that computing the maximum independent set for the discrete and continuous variants is a different ball game altogether. In particular, we show that the discrete variant admits a PTAS. However, the problem becomes hard to approximate beyond a constant even when the polygonal chains are as long as 7 and in the plane. For the discrete case, we use the fact that both the Fréchet and Hausdorff distance metrics have a constant doubling dimension for constant ambient dimension and constant length of the polygonal chain. Thus, one can use the known sublinear separators to run a polynomial time local search algorithm. On the other hand, for the continuous variant, we reduce the problem of finding the maximum independent set of boxes in d-dimensions to a unit ball graph for curves of length $O(d)$. For $d = 2$, the former problem, known as the Maximum Independent Set of Rectangles, enjoys a constant-factor approximation algorithm [Mitchell 2021, Galvez et al. 2022]. It is already APX-hard for $d > 2$ [Chlevík and Chlevíková 2007], thus implying that finding a maximum independent set of unit balls under continuous (weak) Fréchet or Hausdorff metric is hard to approximate for even small polygonal chains in the plane.

1 Introduction

Studying similarities between curves has been a major area of research in Computational Geometry. Two such measures are Fréchet and Hausdorff distances, named after Maurice Fréchet [12] and Felix Hausdorff [16]. In Computational Geometry, it was Alt and Godau [3, 13] who initiated the idea to use Fréchet distance measure to study similarities between curves. Later, Eiter and Mannila [11] proposed the idea of discrete Fréchet distance in an attempt to approximate the curves.

In this article, we study optimization problems, such as packing, on a family of curves with respect to Fréchet and Hausdorff distance measures. More precisely, for every curve we define a metric ball of some radius with respect to the above distance measures. Thus for n curves we get n different balls and we study graph optimization problems e.g., maximum independent set, on the intersection graph of these balls.

1.1 Related work

The main focus involving Fréchet metric has been to get faster algorithms to compute the Fréchet distance between two input polygonal chains. Let m be the length of the polygonal chains. Godau [13] gave an $O(m^3)$-time algorithm for computing the continuous Fréchet distance which was improved by Alt and Godau [3] to $O(m^2 \log m)$, whereas Eiter and Mannila [11] gave an $O(m^2)$ time dynamic programming based algorithm for the discrete variant. Subsequently, the running times were improved to $O(m^2(\log\log m)^2)$ for the continuous variant by Buchin et al. [6] and to $O(m^2 \log\log m/\log m)$ for the discrete variant

39th European Workshop on Computational Geometry, Barcelona, Spain, March 29–31, 2023.
This is an extended abstract of a presentation given at EuroCG'23. 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.

by Agarwal et al. [2]. However, there are results that prove strongly subquadratic time algorithms do not exist assuming SETH [4].

There has also been a host of work done to compute approximate distances [5, 7] and involving geometric range searching [1]. Recently, there has been bounds given on the VC-dimension of set systems defined by Fréchet and Hausdorff balls [9].

1.2 Our Contribution

We make an attempt to distinguish the hardness of approximation for a representative set of problems, viz., MaxIndependentSet, MinDominatingSet, and MinVertexCover, between discrete and continuous distance measures, viz., Fréchet and Hausdorff distance metric, focusing on polygonal chains of constant length. To that end, we propose the following results.

▶ **Theorem 3.1.** *Local search yields a PTAS for MaxIndependentSet, MinDominatingSet, MinVertexCover for ball graphs with respect to $(\mathbb{X}^m, \delta_{dF})$ and $(\mathbb{X}^m, \delta_{dH})$ where $m = O(1)$ and $\mathbb{X}$ is a doubling metric.*

▶ **Theorem 4.4.** *The problems MaxIndependentSet, MinDominatingSet, MinClique are APX-hard for unit ball graphs under the (weak) continuous Fréchet and Hausdorff metrics even when the polygonal chains are of length 7 in the plane.*

2 Preliminaries

Given a point p_i and a positive real number r_i in a metric space $(\mathbb{X}, \delta)$, we define a metric ball $B(p_i, r_i) := \{x \in X \mid \delta(p_i, x) \leq r_i\}$. We shall often refer to $B(p_i, r_i)$ as B_i. In our setting we are usually given a set of n metric balls that we also refer as a neighborhood system, $\Gamma = \{B_1, \dots, B_n\}$ and we define the depth of a point x with respect to Γ, $\mathsf{depth}_\Gamma(x) := |\{B_i \in \Gamma \mid x \in B_i\}|$. We define the ply of the system as $\mathsf{ply}(\Gamma) := \max_{x \in X} \mathsf{depth}_\Gamma(x)$. We may refer to Γ as a k-ply neighborhood system when $\mathsf{ply}(\Gamma) = k$. The intersection graph $G = (V, E)$ of Γ if for every $B_i \in \Gamma$ there is a vertex $v_i \in V$ and $(v_i, v_j) \in E$ for $i \neq j$ iff $B_i \cap B_j \neq \emptyset$. The intersection graph G may also be referred as the ball graph with respect to the metric space $(\mathbb{X}, \delta)$.

Our algorithmic results for the discrete distance metrics hold for points embedded in doubling metrics whereas we show hardness when the curves are in the Euclidean plane. Given a sequence of points $(p_1, \dots, p_m)$, where $p_i \in \mathbb{R}^d$, a polygonal chain or curve C is the union of the line segments $\overline{p_i, p_{i+1}}$ for $1 \leq i < m$. We say that m is the length of C. Given a metric space $\mathbb{X}$ e.g., Euclidean space $\mathbb{R}^d$, we shall refer to $\mathbb{X}^m$ to be the space of all point tuples of length at most m in $\mathbb{X}$.

▶ **Definition 2.1** (Traversal). Given two polygonal chains $P = (p_1, \dots, p_{m_P})$ and $Q = (q_1, \dots, q_{m_Q})$ in $\mathbb{R}^d$, we say $T = (i_1, j_1), \dots, (i_t, j_t)$ is a traversal for P and Q, where $1 \leq i_1, \dots, i_t \leq m_P$ and $1 \leq j_1, \dots, j_t \leq m_Q$, if the following conditions are satisfied.

1. $i_1, j_1 = 1$, $i_t = m_P$, $j_t = m_Q$
2. $i_{k+1} - i_k \in \{0, 1\}$ and $j_{k+1} - j_k \in \{0, 1\}$ for $1 \leq k < t$
3. $(i_{k+1} - i_k) + (j_{k+1} - j_k) \in \{1, 2\}$ for $1 \leq k < t$

▶ **Definition 2.2** (Discrete Fréchet distance). Given two polygonal chains $P = (p_1, \dots, p_{m_P})$ and $Q = (q_1, \dots, q_{m_Q})$ in $\mathbb{R}^d$, let $\mathcal{T}$ be the set of possible traversals for P and Q. We define the discrete Fréchet distance between P and Q as follows.

$$\delta_{dF}(P, Q) := \min_{T \in \mathcal{T}} \max_{(i_k, j_k) \in T} \|p_{i_k} - q_{j_k}\|$$

Observe that any polygonal chain $P = (p_1, \ldots, p_{m_P})$ in $\mathbb{R}^d$ has a uniform parametrization which allows us to express it as a parametrized curve $p : [0,1] \to \mathbb{R}^d$.

▶ **Definition 2.3** (Continuous Fréchet distance)**.** Given two polygonal chains $P = (p_1, \ldots, p_{m_P})$ and $Q = (q_1, \ldots, q_{m_Q})$ in $\mathbb{R}^d$, we define the continuous Fréchet distance between P and Q as follows.

$$\delta_F(P,Q) := \inf_{f:[0,1]\to[0,1]} \max_{\alpha\in[0,1]} \|p(\alpha) - q(f(\alpha))\|$$

where f ranges over all continuous and monotone mappings such that $f(0) = 0$ and $f(1) = 1$. If we drop the restriction of monotonicity then we call it the weak Fréchet distance.

Now we define the Hausdorff distance measure. Although we are concerned with polygonal chains, we define it for a general subset of points.

▶ **Definition 2.4** (Directed Hausdorff distance)**.** Given two subsets $P, Q \subseteq \mathbb{R}^d$, the directed Hausdorff distance from P to Q is as follows.

$$\delta_{\overrightarrow{H}}(P,Q) := \sup_{p\in P} \inf_{q\in Q} \|p - q\|$$

▶ **Definition 2.5** (Hausdorff distance)**.** Given two subsets $P, Q \subseteq \mathbb{R}^d$, the Hausdorff distance from P to Q is as follows.

$$\delta_H(P,Q) := \max\{\delta_{\overrightarrow{H}}(P,Q), \delta_{\overrightarrow{H}}(Q,P)\}$$

When both P and Q are discrete point sets then we say that the Hausdorff distance is discrete. Thus when P and Q are polygonal chains, the discrete Hausdorff distance between them is the Hausdorff distance between the two sets of points defining the chains.

It is a known fact that the above distance measures are metrics and therefore, we can define a ball graph with respect to them which shall be our objects of interest in this article. We shall denote the metrics, discrete Fréchet, discrete Hausdorff, continuous Fréchet, continuous Hausdorff, and weak Fréchet as $(\mathbb{X}^m, \delta_{dF})$, $(\mathbb{X}^m, \delta_{dH})$, $(\mathbb{X}^m, \delta_F)$, $(\mathbb{X}^m, \delta_H)$, and $(\mathbb{X}^m, \delta_{wF})$, respectively. Once we define a ball graph with respect to these metrics, we address a few graph optimization questions like, MaxIndependentSet, MinDominatingSet, MinVertexCover, etc.

Below we state a few theorems that we need to prove our results. We elaborate on these theorems in the full version.

▶ **Theorem 2.6** ([15])**.** *Local search yields a PTAS for MaxIndependentSet, MinDominatingSet, MinVertexCover for graphs with strongly sublinear separators.*

▶ **Lemma 2.7** (Theorem 46 in [10])**.** *The doubling dimension of the discrete Fréchet metric* $\mathit{ddim}(\mathbb{X}^m, \delta_{dF}) \le (4\gamma)^m$ *if that of the ambient metric space* $\mathit{ddim}(\mathbb{X}, \delta) \le \gamma$.

▶ **Lemma 2.8.** *The doubling dimension of the discrete Hausdorff metric* $\mathit{ddim}(\mathbb{X}^m, \delta_{dH}) \le (4\gamma)^m$ *if the doubling dimension of the ambient metric space* $\mathit{ddim}(\mathbb{X}, \delta) \le \gamma$.

▶ **Theorem 2.9** ([18])**.** *Let Γ be a k-ply neighborhood system of n balls in a metric space with doubling dimension γ. Then there exists a balanced separator S in the intersection graph of Γ, where*

$$|S| = O\left(k^{1/(\gamma+1)} \left(\frac{n}{\log n}\right)^{1-1/(2\gamma+2)}\right).$$

▶ **Theorem 2.10** (Theorem 7 in [8])**.** *Each of the problems, MaxIndependentSet, MinDominatingSet, MinVertexCover is APX-hard when restricted to intersection graphs of axis-parallel d-dimensional boxes, for $d \ge 3$.*

3 Discrete Fréchet and Hausdorff Distances

As a consequence of Theorem 2.9, we have the following results for ball graphs with respect to $(\mathbb{X}^m, \delta_{dF})$ and $(\mathbb{X}^m, \delta_{dH})$.

▶ **Theorem 3.1.** *Local search yields a PTAS for* MaxIndependentSet*,* MinDominatingSet*,* MinVertexCover *for ball graphs with respect to* $(\mathbb{X}^m, \delta_{dF})$ *and* $(\mathbb{X}^m, \delta_{dH})$ *where* $m = O(1)$ *and* $\mathbb{X}$ *is a doubling metric.*

Proof. As the doubling dimension of $\mathbb{X}$ is constant and m is also a constant, from Lemmata 2.7 and 2.8, the size of the separator of any subset of balls in the ball graph is sublinear in the size of the subset. Hence, from Theorem 2.6 it follows that local search gives a $(1+\epsilon)$-approximate solution for MaxIndependentSet, MinDominatingSet, and MinVertexCover in time $n^{O(1/\epsilon^2)}$. ◀

The above list of optimization problems is by no means exhaustive. The purpose is to pick a representative set of problems to show the contrast in hardness between the discrete and continuous variants.

4 Continuous Fréchet Distance

Our main result of this section is that the intersection graph of boxes in $\mathbb{R}^d$ can be realized as the unit ball graphs of polygonal chains of length $3d$ in $\mathbb{R}^2$ under the continuous Hausdorff and (weak) Fréchet metrics. For the sake of brevity, we only say continuous Fréchet metrics. We first describe a slightly less efficient realization that uses slightly more line segments and then we show how the number can be cut down to the claimed amount.

The main gadget that we use is the following embedding of intervals into curves consisting of 5 line segments.

▶ **Lemma 4.1.** *Consider a set S of n intervals, $[a_i, b_i]$, for $1 \leq i \leq n$. Each interval $I_i \in S$ can be embedded to a curve C_i such that the following three properties hold. (i) if I_i and I_j are disjoint, then the curves C_i and C_j have Fréchet distance two but (ii) if I_i and I_j are intersecting, then C_i and C_j have Fréchet distance one. (iii) Furthermore, each C_i starts at some fix X-coordinate X_s and ends at some fixed X-coordinate X_e where $X_e - X_s = O(n)$.*

Proof. Observe that we can assume that the coordinates of the intervals are even integers, between X_s and $X_e = X_s + 4n$, by sorting them and then relabelling them with even integers starting X_s. Now, C_i will be a Y-monotone curve, i.e., a function from X to Y, defined on the range $[X_s, X_e]$, as follows (see Figure 1):

$$C_i(x) = \begin{cases} 0 & x \leq a_i \\ 1 & a_i < x \leq b_i \\ 2 & b_i < x \end{cases} \tag{1}$$

The claimed property of this embedding can be easily verified by a case analysis (See Figure 2).

- Consider two intersecting intervals $I_i = [a_i, b_j]$ and $I_j = [a_j, b_j]$ and w.l.o.g, assume that $a_j \leq a_i$, which implies we must have $b_j < a_i$ as otherwise, they will be intersecting. Since the coordinates are even integers, it follows that $a_i \geq b_j + 2$ which in turn implies that the point $V_j = (a_j, C_j(a_j))$ has distance at least two to the curve C_i.

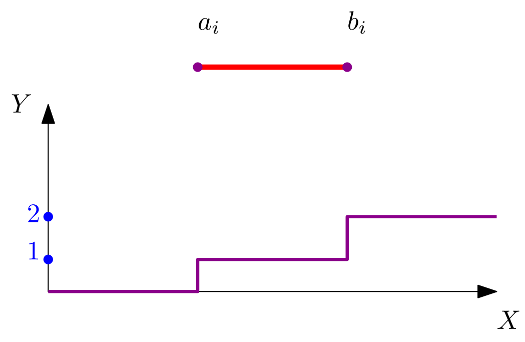


Figure 1 Embedding an interval as a curve.

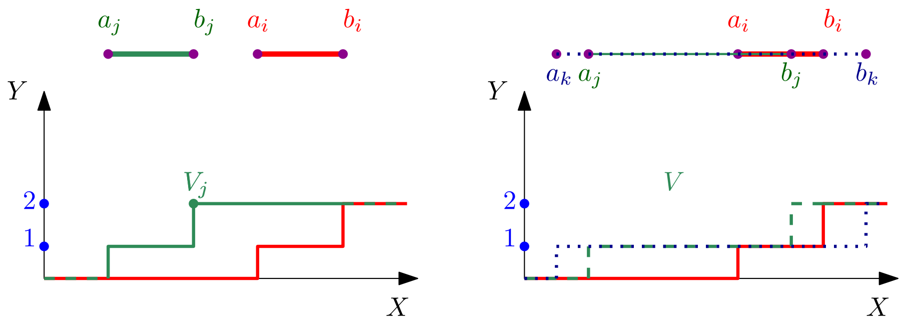


Figure 2 (left) The case for two disjoint intervals $I_i = [a_i, b_i]$ and $I_j = [a_j, b_j]$. (right) The case for the interval $I_i = [a_i, b_i]$ intersecting the two intervals $I_j = [a_j, b_j]$ and $I_k = [a_k, b_k]$.

- If I_i and I_j are intersecting, then we must have $a_i \leq b_j$. Now, one can verify that $|C_j(x) - C_i(x)| \leq 1$ for every value of x and thus the curves have Fréchet distance of 1. ◀

▶ **Lemma 4.2.** *Given a set of boxes $\mathcal{B} = \{B_1, \ldots, B_n\}$ in $\mathbb{R}^d$, for every B_i we can construct a polygonal chain C_i, consisting of $6d - 1$ line segments each, such that the intersection graph of $\mathcal{B}$ is isomorphic to the unit ball graph defined over C_i, for $2 \leq i \leq n$ under the continuous (weak) Fréchet distance metric.*

Proof. A box in d-dimension is a Cartesian product of d intervals. Let $B_i = [a_{i1}, b_{i1}] \times \cdots \times [a_{id}, b_{id}]$. Observe that two boxes B_i and B_j intersect if and only if for every k, $1 \leq k \leq d$, the intervals $[a_{ik}, b_{ik}]$ and $[a_{jk}, b_{jk}]$ intersect. Next, we can embed the k-th dimension of the boxes into the integers in the range $[4n(k-1), 4nk)$ using the interval embedding outlined in the above lemma. See Figure 3. ◀

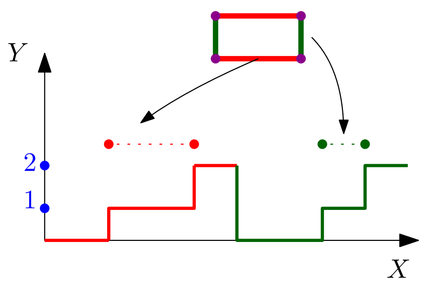


Figure 3 Each interval representing a side of the box is embedded into a curve.

In the above representation we are using at most six line segments to encode every dimension of the box. Thus, a d-dimensional box would require polygonal chains consisting of $6d-1$ line segments. Below, we show that this can be reduced significantly by some slight compromises: first, by embedding the x-coordinates on an exponential grid, and second, by having curves representing non-intersecting boxes Fréchet distance of $2-\varepsilon$, for some fixed $\varepsilon > 0$. We state the lemma without the proof (which can be found in the full version) below.

▶ **Lemma 4.3.** *Given a set of boxes $\mathcal{B} = \{B_1, \dots, B_n\}$ in $\mathbb{R}^d$, for every B_i we can construct a polygonal chain C_i of $2d+1$ complexity such that if two boxes B_i and B_j intersect, then C_i and C_j have (weak) Fréchet distance of 1 but otherwise, they have Fréchet distance of $2-\varepsilon$, where $\varepsilon > 0$ is a fixed constant.*

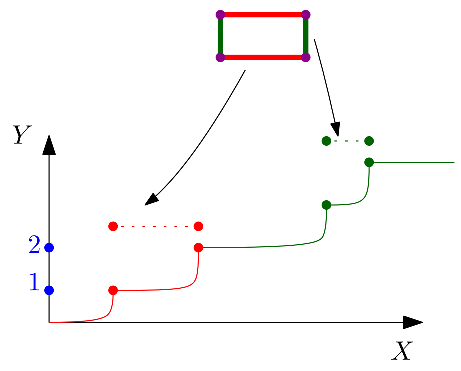


Figure 4 Embedding the construction on the exponential grid allows us to represent each dimension of the box with only two line segments. This picture is drawn with X-axis in the log-scale.

▶ **Theorem 4.4.** *The problems* MaxIndependentSet*,* MinDominatingSet*,* MinClique *are APX-hard for unit ball graphs under the (weak) continuous Fréchet and Hausdorff metrics even when the polygonal chains are of length 7 in the plane.*

Proof. This follows from Lemma 4.3 in combination with Theorem 2.10. ◀

5 Conclusion

We find it worthwhile to note that by our construction, finding a PTAS for the (weak) continuous Fréchet or Hausdorff distance in the plane for polygonal chains consisting of 5 line segments, would also improve the polynomial time constant-factor approximation algorithm for the intersection graphs of boxes in $\mathbb{R}^2$ by Mitchell [17] and Galvez et al. [14], which has been a major open problem.

For boxes in $\mathbb{R}^d$, the only known technique is to divide and conquer by projecting the boxes in $\mathbb{R}^{d-1}$ and incur an $O(\log n)$ multiplicative overhead in the overall approximation factor. Therefore, the current best known approximation factor is $O(\log^{d-2} n)$ achievable in polynomial time. As solving the MaxIndependentSet for unit balls for polygonal chains of length $2d+1$ in the plane is at least as hard as for the d-dimensional boxes, we ask whether there exists a polynomial time $O(\log^m n)$-approximation algorithm for MaxIndependentSet for polygonal chains of length m in the plane, for $m > 5$.

References

1 Peyman Afshani and Anne Driemel. On the complexity of range searching among curves. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 898–917. SIAM, 2018.

2 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.

3 Helmut Alt and Michael Godau. Computing the Fréchet distance between two polygonal curves. *International Journal of Computational Geometry & Applications*, 5(01n02):75–91, 1995.

4 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.

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

6 Kevin Buchin, Maike Buchin, Wouter Meulemans, and Wolfgang Mulzer. Four Soviets walk the dog—with an application to Alt's conjecture. In *Proceedings of the Twenty-Fifth annual ACM-SIAM Symposium on Discrete algorithms*, pages 1399–1413. SIAM, 2014.

7 Timothy M Chan and Zahed Rahmati. An improved approximation algorithm for the discrete Fréchet distance. *Information Processing Letters*, 138:72–74, 2018.

8 Miroslav Chlebík and Janka Chlebíková. The complexity of combinatorial optimization problems on d-dimensional boxes. *SIAM Journal on Discrete Mathematics*, 21(1):158–169, 2007.

9 Anne Driemel, André Nusser, Jeff M Phillips, and Ioannis Psarros. The VC dimension of metric balls under Fréchet and Hausdorff distances. *Discrete & Computational Geometry*, 66(4):1351–1381, 2021.

10 Anne Driemel, Ioannis Psarros, and Melanie Schmidt. Sublinear data structures for short Fréchet queries, 2019. `arXiv:1907.04420`.

11 Thomas Eiter and Heikki Mannila. Computing discrete Fréchet distance. Technical Report CD-TR 94/64, Technical University of Vienna, 1994.

12 M Maurice Fréchet. Sur quelques points du calcul fonctionnel. *Rendiconti del Circolo Matematico di Palermo (1884-1940)*, 22(1):1–72, 1906.

13 Michael Godau. A natural metric for curves — computing the distance for polygonal chains and approximation algorithms. In *Annual Symposium on Theoretical Aspects of Computer Science*, pages 127–136, 1991.

14 Waldo Gálvez, Arindam Khan, Mathieu Mari, Tobias Mömke, Madhusudhan Reddy, and Andreas Wiese. A (2+ϵ)-approximation algorithm for maximum independent set of rectangles, 2021. `arXiv:2106.00623`.

15 Sariel Har-Peled and Kent Quanrud. Approximation algorithms for polynomial-expansion and low-density graphs. In *Algorithms – ESA 2015 - 23rd Annual European Symposium, Patras, Greece, September 14–16, 2015, Proceedings*, pages 717–728, 2015.

16 Felix Hausdorff. *Grundzüge der mengenlehre*, volume 7. von Veit, 1914.

17 Joseph S. B. Mitchell. Approximating maximum independent set for rectangles in the plane. In *2021 IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 339–350, 2022.

18 Yingchao Zhao and Shang hua Teng. Combinatorial and spectral aspects of nearest neighbor graphs in doubling dimensional and nearly-euclidean spaces. *Theoretical Computer Science*, 410(11):1081–1092, 2009. Algorithms, Complexity and Models of Computation.

Non-convex position of lines in $\mathbb{R}^3$

Barbora Dohnalová[1], Miroslav Horský[2], and Ondřej Chwiedziuk[3]

1 Faculty of Mathematics and Physics, Charles University
bdohnalova@matfyz.cz
2 Faculty of Mathematics and Physics, Charles University
mhorsky@matfyz.cz
3 Faculty of Mathematics and Physics, Charles University
chwiedziuk@matfyz.cz

Abstract

Bárány, Kalai and Pór recently proved that for each $d, k \geq 1$ there exists a finite number $n = n(d, k)$ such that there exist n k-flats in $\mathbb{R}^d$ which are not in convex position, that is there exists no convex set touching all the k-flats. In particular, for $k = 1, d = 3$ there exist n lines in $\mathbb{R}^3$, which are not in convex position. We discovered a specific upper bound for certain special cases and for the general position.

1 Introduction

1.1 Motivation

We consider the following problem: what is the minimum number of points in $\mathbb{R}^2$ in general position which can be in non-convex position? This is a special case of the problem proposed in [2]. The solution can be easily seen from figure 1.

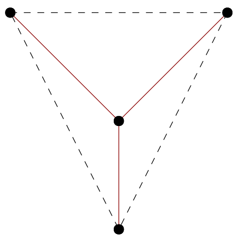


Figure 1 Four points in non-convex position in the plane.

1.2 Problem Setup

This problem is generalised in [2]. We will consider another particular version: assume that we have lines in general position. Then we ask if there is a convex set such that every line touches the convex set but does not pass through it. How many lines in an appropriate arrangement do we need so that there will not be any such convex set? From [2] we know that this number is finite but we would like to know at least an upper bound.

Precise Problem Formulation.

▶ **Definition 1.1.** We say that lines $\{p_i\}_{i=1}^n : \forall i p_i \subset \mathbb{R}^3$ are in *convex position* if there exists an open convex nonempty set $M \subset \mathbb{R}^3$ such that $\forall i \in \{1, \ldots, n\} : p_i \cap M = \emptyset \wedge p_i \cap \overline{M} \neq \emptyset$.

▶ **Definition 1.2.** Lines $\{p_i\}_{i=1}^n$ are in *general position* if no two lines lie in the same plane.

▶ **Definition 1.3.** Lines $\{p_i\}_{i=1}^n$ are in *general position with respect to direction vectors* if every triple of the direction vectors of distinct lines are linearly independent.

▶ **Problem 1.4.** *What is the minimum number n for which there exist lines $\{p_i\}_{i=1}^n$ in general position such that they are not in convex position?*

2 Non-general position

Before discussing the general solution we will solve some examples in non-general position. A trivial example of this is when four lines are parallel to each other. Then the problem degenerates to the two-dimensional problem solved above. Therefore we will take a look at some more interesting non-trivial examples. The first case assumes that at most three lines can lie in one plane and at most three lines can be parallel to each other. Second case assumes that lines can lie in one plane but they are not parallel to each other.

Both solutions use the construction of a plane W which separates two open half-spaces. Then we add two lines p_1, p_2 that do not intersect the plane W and lie in the opposite half-spaces. Then each line segment with endpoints at p_1 and p_2 intersects W. If we take enough lines $q_1, \dots, q_n$ in W, then $\{p_1, p_2, q_1, \dots, q_n\}$ are not in convex position. The first construction takes three parallel lines in the plane W. The second one takes five lines in W.

▶ **Theorem 2.1.** *Let there be a plane $W \subset \mathbb{R}^3$, $\varrho_1, \varrho_2 \subset \mathbb{R}^3$ open half-spaces separated by W, lines $q_1, q_2, q_3 \subset W$ parallel to each other and lines p_1, p_2 such that $p_1 \subset \varrho_1$, $p_2 \subset \varrho_2$. Then $\{p_1, p_2, q_1, q_2, q_3\}$ are not in convex position.*

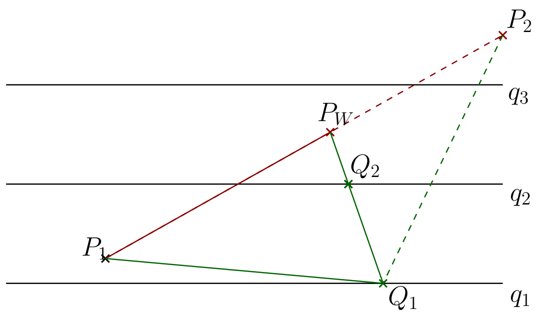


Figure 2 Three parallel lines from the construction of non-convex position of lines.

Proof. Suppose that lines are in convex position. Then there exists an open convex set $M \subset \mathbb{R}^3$ such that $\forall p \in \{p_1, p_2, q_1, q_2, q_3\} : p \cap M = \emptyset \wedge p \cap \overline{M} \neq \emptyset$.

Three parallel lines q_1, q_2, q_3 divide W into four cells. Let $P_1 \in p_1 \cap \partial M, P_2 \in p_2 \cap \partial M$. Then P_1P_2 intersects W in one point P_W, which is a convex combination of P_1, P_2. This point can lie either on some q_i or in some cell Q of W separated by our parallel lines.

The first case implies that $P_W \in M$, which derives a contradiction.

Second case states that P_W lies in Q. The boundary of Q is defined by at most two of the lines q_1, q_2, q_3. We take the one that does not belong to ∂Q, without loss of generality q_1. We choose any point $Q_1 \in q_1 \cap \partial M$. Then $P_W Q_1$ intersects at least one of the other lines q_2, q_3 at one point Q_2 (illustrated in figure 2). This implies that $Q_2 \in M$, which derives a contradiction too. Therefore they are not in convex position. ◀

▶ **Theorem 2.2.** *Let there be a plane $W \subset \mathbb{R}^3$, $\varrho_1, \varrho_2 \subset \mathbb{R}^3$ open half-spaces separated by W. Then there exist lines $q_1, q_2, q_3, q_4, q_5 \subset W$ pairwise non-parallel, $p_1 \subset \varrho_1$, $p_2 \subset \varrho_2$, such that lines $\{p_1, p_2, q_1, q_2, q_3, q_4, q_5\}$ are not in convex position.*

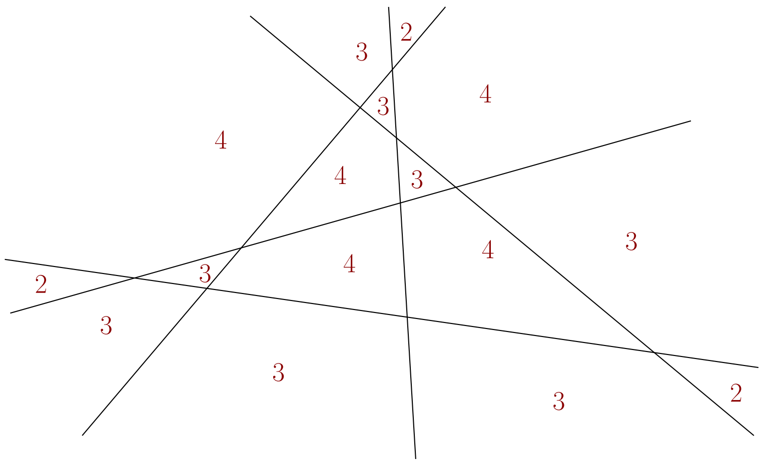


Figure 3 Five lines in a plane from the construction of non-convex position of lines.

Proof. Suppose that they are in convex position. Then there exists an open convex set $M \subset \mathbb{R}^3$ such that $\forall i \in \{1, \dots, n\} : p_i \cap M = \emptyset \wedge p_i \cap \overline{M} \neq \emptyset$.

We take an arrangement such that every open subset of W separated by lines $q_1, \dots, q_5$ is formed by at most four lines, for example as shown in figure 3. Let $P_1 \in p_1 \cap \partial M, P_2 \in p_2 \cap \partial M$. Then P_1P_2 intersects W in one point P_W, which is a convex combination of P_1, P_2. This point can lie either on some q_i or in some open set Q.

The first case implies that $P_W \in M$, which derives a contradiction.

In the second case, without loss of generality we take q_1 such that $q_1 \cap \partial Q = \emptyset$. We choose any point $Q_1 \in q_1$. Then P_WQ_1 intersects at least one of the other lines $q_2, \dots, q_5$ at one point Q_P. This implies that $Q_P \in M$, which derives a contradiction as well. Hence they are not in convex position. ◀

3 General position

One of the possible definitions of general position is that no two lines lie in the same hyperplane. In the next few paragraphs we will show a construction of 10 lines in such position which are in non-convex position.

▶ **Definition 3.1.** A *gadget* is a set of 8 lines in $\mathbb{R}^3$ arranged in the way illustrated in Figures 4 and 5. Line z is not included in the 8 lines of the construction, it is simply an axis intersected by all of the other lines. For the sake of simplicity we shall assume in the following lemmata that the line z is the z-axis. (The particular position of the lines is not important, only their ordering along the line z and the cyclic order of their direction vectors.)

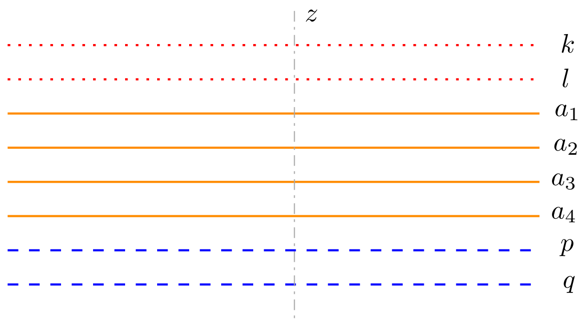


Figure 4 Side view of the gadget.

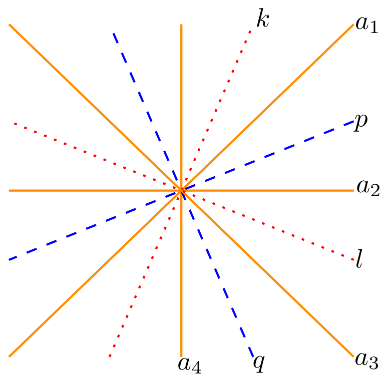


Figure 5 Top view of the gadget.

In the following paragraphs assume the lines in the gadget from Definition 3.1. are in convex position and let $\mathcal{C}$ be the convex set from the definiton of a convex position. Let K, L, P and Q be the points where $\mathcal{C}$ touches lines k, l, p and q.

▶ **Lemma 3.2.** *Let ϱ_1 be the plane determined by lines z and a_1. Let ϱ_1^+ and ϱ_1^- be the two closed half-spaces separated by ϱ_1.*

If points K and L lie in the opposite open half-spaces from ϱ_1 then P and Q must lie in the same half-space. (And vice versa for P and Q.)

Proof. If K lies in the opposite open half-space to L and P lies in the opposite open half-space to Q, then a_1 would intersect $\mathcal{C}$.

Denote X and Y the points where the line segments KL and PQ intersect the plane ϱ_1. Since K and L both have larger z-coordinate than a_1, X also has larger z-coordinate than a_1. Similarly Y has z-coordinate smaller than a_1. Therefore a_1 will intersect the segment XY and hence the whole convex set $\mathcal{C}$. ◀

We can rephrase Lemma 3.2 as the following condition: It holds for at least one of the pairs $\{K, L\}$ and $\{P, Q\}$ that either both of the points from the pair lie in ϱ_1^+ or both of them lie in ϱ_1^-.

▶ **Lemma 3.3.** *At least one point from the set $\{K, L\}$ and at least one point from the set $\{P, Q\}$ lies on z.*

Proof. Suppose that no point from the set $\{K, L, P, Q\}$ lies on z. Lemma 3.2 must hold for all the lines a_1, a_2, a_3, a_4. Consider the lines a_2 and a_4. One of the pairs $\{K, L\}$ or $\{P, Q\}$ has to satisfy the condition from Lemma 3.2 (that is, it lies in the same half-space) for a_2 and the other pair has to satisfy the condition for a_4. Note that no pair can satisfy both conditions at once.

However, if we choose K, L to satisfy the condition for a_2 and P, Q to satisfy the condition for a_4, then $\mathcal{C}$ is intersected by a_3. If we choose P, Q to satisfy the condition for a_2 and K, L to satisfy the condition for a_4, then $\mathcal{C}$ is intersected by a_1. This yields a contradiction. Therefore at least one of the points K, L, P, Q must lie on z.

Without loss of generality assume that K lies on z. If none of the points P, Q lie on z, then one of the lines a_1, a_2, a_3, a_4 would intersect the triangle K, P, Q, hence the whole convex set $\mathcal{C}$. This means at least one of the points P, Q has to also lie on z. ◀

We have shown that any convex set which touches all the lines in the gadget has to intersect line z (and contain the whole line segment between the two points of intersection).

This allows us to use the gadget as a substitute for lines in non-general position. For example in the construction with three parallel lines blocking the plane (see Figure 2) we can substitute the outer two parallel lines with two gadgets and get a construction in general position with 19 lines.

However, there exists a simpler way to obtain a set of lines in non-convex position from the gadget. It requires one more lemma to prove.

▶ **Lemma 3.4.** *Consider the cells determined by planes ϱ_1, ϱ_2, ϱ_3, ϱ_4 (where z and a_i determine ϱ_i). The convex set $\mathcal{C}$ cannot contain points from more than one of those cells.*

Proof. Without loss of generality suppose K and P are the two points lying on z. Suppose there exist two points $R, S \in \mathcal{C}$ such that each lies in a different cell. Since $\mathcal{C}$ must be three-dimensional, we can assume that R and S do not lie in the same hyperplane passing through the z axis.

Let ϱ_i be the plane which separates R and S and let X be the intersection of RS and ϱ_i. Either X has higher z-coordinate than a_i, therefore PX intersects a_i, or X has smaller z-coordinate than a_i, therefore KX intersects a_i. In both cases a_i intersects $\mathcal{C}$, which is a contradiction. ◀

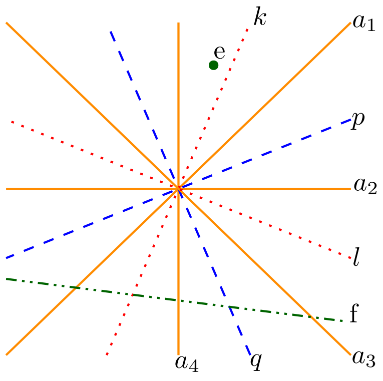


Figure 6 Example of 10 lines in non-convex position. (Line e, which is parallel to z, is drawn as a point.)

Thus we can simply add two lines, e and f, to the construction such that they do not traverse the same cell. (For example choose one line parallel with z and let the other one be any line that does not pass through the same cell.) As we have shown in the previous lemma it is impossible for $\mathcal{C}$ to touch them both at the same time. We have therefore proven the following theorem:

▶ **Theorem 3.5.** *There exist* 10 *lines in* $\mathbb{R}^3$ *in general position which are in non-convex position.*

4 Lower bound for general position with respect to direction vectors

▶ **Definition 4.1.** We say points $p_1, \ldots, p_n$ in $\mathbb{R}^3$ on a common plane T are *in a convex (weakly convex) position*, if for some distance preserving map f from T to $\mathbb{R}^2$ the points $f(p_1), \ldots, f(p_n)$ are in convex (weakly convex, respectively) position.

▶ Remark. Let $\{l_1, \ldots, l_n\}$ be a set of lines in $\mathbb{R}^3$ in convex position, then a set of lines we get by removing a line is still in convex position.

▶ **Lemma 4.2.** *Let $\{l_1, \dots, l_n\}$ be a set of lines and T a plane in $\mathbb{R}^3$. If for each $i \in [n]$ $T \cap l_i = \{p_i\}$ for some not necessarily different points $p_1, \dots, p_n$ in $\mathbb{R}^3$ that are in weakly convex position or are collinear, then the lines are in convex position.*

Proof. Assume first that the convex hull of the points $p_1, ..., p_n$ is a two dimensional polytope P. Then there exists a point q on the line that is orthogonal to the plane T and passing through the barycenter of P, such that the convex hull of the points $q, p_1, \dots, p_n$ is a three dimensional polytope H which has nonempty intersection with each of those lines but $\operatorname{int} H$ has empty intersection with each of those lines.
If the convex hull of the points $p_1, \dots, p_n$ is less than two dimensional we add additional lines so that the points in the intersection of the lines and the plane T are in convex position, then we use the previous part of the proof and the remark before this lemma. ◀

▶ **Lemma 4.3.** *Let $\{p_1, \dots, p_n\}$ be a set of points in $\mathbb{R}^2$ in convex position, then $\exists r > 0$, such that $\forall (a_1, \dots, a_n) \in B_r(p_1) \times \dots \times B_r(p_n)$ the points $a_1, \dots, a_n$ are in convex position.*

Proof. For the case $n \leq 3$ this statement is obvious. Now we will prove the case when $n = 4$. First we make an observation, for all triples of points x, y, z in $\mathbb{R}^2$ the following holds

$$\text{points } x, y, z \text{ are collinear} \iff \det \begin{pmatrix} x_1 & x_2 & 1 \\ y_1 & y_2 & 1 \\ z_1 & z_2 & 1 \end{pmatrix} = 0.$$

We define map $F : \mathbb{R}^2 \times \mathbb{R}^2 \times \mathbb{R}^2 \times \mathbb{R}^2 \to \mathbb{R}$ so that $F(x, y, z, v) = 0$ if and only if at least one triple of the four points x, y, z, v is collinear. So we define F as follows:

$$F(x, y, z, v) = \det \begin{pmatrix} x_1 & x_2 & 1 \\ y_1 & y_2 & 1 \\ z_1 & z_2 & 1 \end{pmatrix} \det \begin{pmatrix} x_1 & x_2 & 1 \\ y_1 & y_2 & 1 \\ v_1 & v_2 & 1 \end{pmatrix} \det \begin{pmatrix} x_1 & x_2 & 1 \\ v_1 & v_2 & 1 \\ z_1 & z_2 & 1 \end{pmatrix} \det \begin{pmatrix} v_1 & v_2 & 1 \\ y_1 & y_2 & 1 \\ z_1 & z_2 & 1 \end{pmatrix}.$$

Because F is continuous and $F(p_1, p_2, p_3, p_4) \neq 0$ then

$$\exists r \in \mathbb{R}, r > 0 \, \forall (a_1, \dots, a_4) \in B_r(p_1) \times \dots \times B_r(p_4) : F(a_1, a_2, a_3, a_4) \neq 0.$$

Now suppose there exist points $(a_1, \dots, a_4) \in B_r(p_1) \times \dots \times B_r(p_4)$ that are in non-convex position, then for each $i \in [4]$ there exists a point b_i on a line segment between a_i and p_i such that there is a triple of the points b_1, b_2, b_3, b_4 that is collinear. But then $F(b_1, b_2, b_3, b_4) = 0$.
The case when $n \geq 5$ follows from the following fact: a set of points is in non-convex position if and only if there is a quadruple of these points that is in non-convex position, this is a special case of Carathéodory's theorem. ◀

▶ **Lemma 4.4.** *Let $\{l_1, \dots, l_n\}$ be a set of lines in $\mathbb{R}^3$ passing through the origin and let T be a plane in $\mathbb{R}^3$ such that for each $i \in [n]$ it holds that $T \cap l_i = \{p_i\}$ for some points $p_1, \dots, p_n$ in $\mathbb{R}^3$ that are in convex position. If $s_1, \dots, s_n$ are lines in $\mathbb{R}^3$ and assume that for every $i \in [n]$ s_i is parallel to l_i, then they are in convex position.*

Proof. For any $i \in [n]$ the distance between the only point in $K \cap l_i$ and the only point in $K \cap s_i$ is the same for every plane K parallel to the plane T, we indicate this distance as d_i. We define $d = 2 \cdot \max_{i \in [n]} (d_i)$. Now we can shift the plane T further from the origin so that there exists $r \in \mathbb{R}$ from Lemma 4.3 that is greater than d. (We used the Lemma 4.3 on the set of points $T \cap \bigcup_{i \in [n]} l_i$ and the plane T.) Hence we showed that the set of points

$T \cap \bigcup_{i\in[n]} s_i$ is in convex position, from Lemma 4.2 follows that the lines $s_1, \dots, s_n$ are in convex position. ◀

▶ **Theorem 4.5.** *Let $4 \leq n \leq 5$ and $\{l_1, \dots, l_n\}$ be a set of lines in $\mathbb{R}^3$, if every triple of their direction vectors are linearly independent, then they are in convex position.*

Proof. We use the fact that there is only one order type of sets of points in general position with size less then 6 in the projective plane, that means those sets can be projectively transformed into sets of points in convex position (as shown in [1]). That means for any lines $l'_1, \dots, l'_n$ passing through the origin there exists a plane T so that the set of points $T \cap \bigcup_{i\in[n]} l'_i$ is in convex position. With the use of the Lemma 4.4 the theorem follows. ◀

5 Conclusion

We have shown constructions of lines in non-convex position for 5 and 7 lines in non-general position and 10 lines in general position. For the general position with respect to direction vectors, we have shown only the lower bound of 6 lines, the upper bound remains to be found.

Acknowledgments. We would like to thank the organizers of the Seminar on Combinatorial Problems, Jan Kynčl and Pavel Valtr, for their guidance through this problem.

The work on this paper was supported by grant no. 21-32817S of the Czech Science Foundation (GAČR).

References

1 Oswin Aichholzer, Franz Aurenhammer, and Hannes Krasser. Enumerating order types for small point sets with applications. *Order*, 19(3):265–281, 2002. URL: `https://doi-org.ezproxy.is.cuni.cz/10.1023/A:1021231927255`, `doi:10.1023/A:1021231927255`.

2 Imre Bárány, Gil Kalai, and Attila Pór. Erdős–Szekeres theorem for k-flats. *Discrete & Computational Geometry*, Oct 2022. `doi:10.1007/s00454-022-00450-4`.

Computing Minimum Complexity 1D Curve Simplifications under the Fréchet Distance

Thijs van der Horst[1,2] **and Tim Ophelders**[1,2]

1 **Department of Information and Computing Sciences, Utrecht University, the Netherlands**
`{t.w.j.vanderhorst|t.a.e.ophelders}@uu.nl`

2 **Department of Mathematics and Computer Science, TU Eindhoven, the Netherlands**

Abstract

We consider the problem of simplifying curves under the Fréchet distance. Let P be a curve and $\varepsilon \geq 0$ be a distance threshold. An ε-simplification is a curve within Fréchet distance ε of P. We consider ε-simplifications of minimum complexity (i.e. minimum number of vertices). Parameterized by ε, we define a continuous family of minimum complexity ε-simplifications P^ε of a curve P in one dimension. We present a data structure that after linear preprocessing time can report the ε-simplification in linear output-sensitive time. Moreover, for $k \geq 1$, we show how this data structure can be used to report a simplification P^ε with at most k vertices that is closest to P in $O(k)$ time.

1 Introduction

Curve simplification is widely studied in computational geometry. It has applications in areas such as geographic information systems [7] and visualisation [8]. Generally, the goal is to transform a curve into a different but similar curve that is easier to work with for the application at hand. For computational purposes, curves are often represented as polylines. Because the running times of algorithms depend mainly on the number of vertices, it is very natural to look for a simplification with a minimum number of vertices.

Consider a polyline P with n vertices. An ε-simplification of P is a curve within distance ε of P. Commonly used distance measures are the Hausdorff distance and Fréchet distance. Although it is slightly harder to compute, the Fréchet distance is more popular in computational geometry because it takes the connectivity of the curve into account. Bereg *et al.* [1] give algorithms for simplifying a polygonal curve in $\mathbb{R}^3$ to one with the minimum number of vertices, where the discrete Fréchet distance is used to measure the similarity between the original curve and its simplification. Depending on the application, it may be natural to require that the vertices of a simplification form a subsequence of those of the original curve. Under this constraint, the algorithm by Bereg *et al.* runs in $O(n^2)$ time. If there are no restrictions, their algorithm runs in $O(n \log n)$ time instead. Under the continuous Fréchet distance in general dimensions, Bringmann and Chaudhury [2] give an $O(n^3)$ time algorithm for the case where vertices must be a subsequence of those of P, and give a matching conditional lower bound. Van Kreveld *et al.* [10] show that under the Hausdorff distance the problem is NP-hard, if vertices are again restricted. The problem remains NP-hard in the unrestricted case, as shown by van de Kerkhof *et al.* [9].

Related work. In this work we consider curve simplification under the Fréchet distance, restricting vertices to lie on P while respecting their order along P, for curves in one dimension. We consider computing two types of simplifications: *min-#* simplifications and *closest k-curves*. A *min-#* ε-simplification of P is an ε-simplification with the minimum

39th European Workshop on Computational Geometry, Barcelona, Spain, March 29–31, 2023.

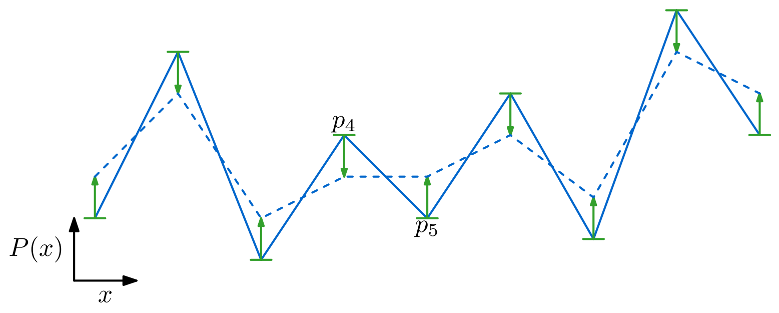


Figure 1 An illustration of a simplification (dashed) for a curve P (solid). Only the vertical axis carries geometric information, the horizontal axis is used only to illustrate the connectivity. The minimum edge length of P is realized by $\overline{p_4p_5}$. The vertices p_4 and p_5 have become degenerate and do not correspond to vertices in the simplification.

number of vertices. A *closest k-curve* of P is a curve P' with at most k vertices and the minimum Fréchet distance to P.

Van de Kerkhof *et al.* [9] give a linear time algorithm for min-# curve simplification in one dimension, with the same restrictions as in our setting. If vertices are restricted further, requiring them to be a subsequence of the vertices of P, then Driemel *et al.* [4] give a linear time algorithm for computing a simplification with at most two vertices more than the minimum number. Their simplification takes the form of a *signature*. The class of signatures also contains a 2-approximation for the closest k-curve, in that it contains a curve with k vertices that is at most twice as far as the closest k-curve. Driemel *et al.* [4] give an $O(k \log k)$ time algorithm for computing such a curve, after $O(n \log n)$ time preprocessing.

Results and organization. In Section 2 we present our family of simplifications for curves in one dimension. We show that our ε-simplification P^ε of a curve P is a min-# ε-simplification of P. We further show that for any positive integer k, there is a simplification in the family that is a closest k-curve for P. In Section 3 we give a data structure for computing P^ε for any $\varepsilon \geq 0$. After $O(n)$ time preprocessing, we can compute P^ε in $O(k)$ time, where k is the complexity of P^ε. This data structure is extended to support querying the minimum $\varepsilon \geq 0$ for which P^ε has at most k vertices in $O(k)$ time. By virtue of being a min-# simplification, P^ε is automatically a closest k-curve.

Preliminaries. A (polygonal) *n-curve* in one dimension is a continuous piecewise-linear function $P\colon [0,1] \to \mathbb{R}$ connecting a sequence $p_1, \ldots, p_n$ of values, which we refer to as *vertices*. A vertex p_i is *degenerate* if it is not a local minimum or maximum of P. An *edge* of P is an interval bounded by consecutive vertices p_i, p_{i+1}. We say that an edge is *increasing* if $p_i < p_{i+1}$ and *decreasing* if $p_i > p_{i+1}$.

A *reparameterization* is a non-decreasing, continuous surjection $f\colon [0,1] \to [0,1]$ where $f(0) = 0$ and $f(1) = 1$. Two reparameterizations f and g describe a *matching* (f, g) between two curves P and Q, where $P(f(t))$ is matched with $Q(g(t))$. A matching (f, g) between 1D curves P and Q is said to have *cost* $\max_t |P(f(t)) - Q(g(t))|$. The (continuous) Fréchet distance $d_F(P, Q)$ between P and Q is the minimum cost over all matchings.

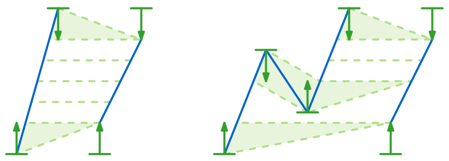


Figure 2 The matching induced by the simplification. (left) Simplifying (truncating) a single edge. Dashed segments indicate point to point matchings, dashed areas indicate subsegments matching to a single point. (right) Simplifying a more complex curve by half its minimum edge length.

2 Simplifications

Throughout this work we consider a polygonal 1D n-curve P without degenerate vertices. In this section we present our family of simplifications of P and show that this family consists of min-# ε-simplifications and contains the closest k-curves. The simplification is closely related to *truncated smoothings* for Reeb graphs [3] and the simplification of van de Kerkhof *et al.* [9].

Let ε' be the minimum edge length of P, and let $\varepsilon \in [0, \varepsilon'/2]$. The ε-simplification P^ε of P is the curve obtained by truncating the edges of P by ε on both sides and removing any degenerate vertex that is created, see Figure 1. We extend the definition to all $\varepsilon \geq 0$ by recursively defining the ε-simplification of P for ε greater than $\varepsilon'/2$ to be the $(\varepsilon - \varepsilon'/2)$-simplification of $P^{\varepsilon'/2}$ if $\varepsilon' > 0$ (i.e. if P has at least one edge), and as P otherwise.

▶ **Theorem 1.** *The Fréchet distance between P and its ε-simplification is at most ε.*

Proof. Let $\varepsilon \geq 0$ and let ε' be the minimum edge length of P. If $\varepsilon \leq \varepsilon'/2$ then there is a natural matching between P and P^ε induced by the truncating operation performed for the simplification. See Figure 2 for an illustration of this matching. This matching has cost at most ε, since points are moved by distance at most ε during truncation. For general ε, applying the triangle inequality to the recursive definition of the simplification yields that

$$d_F(P, P^\varepsilon) \leq \begin{cases} \varepsilon & \text{if } \varepsilon \leq \varepsilon'/2, \\ d_F(P, P^{\varepsilon - \varepsilon'/2}) + \varepsilon'/2 & \text{otherwise,} \end{cases}$$

which implies $d_F(P, P^\varepsilon) \leq \varepsilon$. ◀

We proceed to show that the ε-simplification of P is a min-# ε-simplification of P. An important consequence is that certain simplifications are also closest k-curves for P.

▶ **Theorem 2.** *Let P be a curve in one dimension and let $\varepsilon \geq 0$. The ε-simplification P^ε of P is a min-# ε-simplification of P.*

Proof. There is a matching between P and P^ε that has cost at most ε, where minima on P^ε correspond to minima on P that lie ε lower, and maxima on P^ε correspond to maxima on P that lie ε higher. Let $V^\varepsilon_{\text{orig}}$ be the corresponding sequence of k vertices on P. Note that consecutive vertices of $V^\varepsilon_{\text{orig}}$ are strictly more than 2ε apart, since P^ε alternates between minima and maxima.

Let Q be a curve within distance ε of P and consider a matching between P and Q that has cost at most ε. We show that Q has at least as many vertices as P^ε. For two consecutive vertices p_i and p_j of $V^\varepsilon_{\text{orig}}$, consider the subcurve P' of P starting at p_i and ending at p_j, and let Q' be the subcurve of Q matched to P'. Because p_i and p_j are more than 2ε apart,

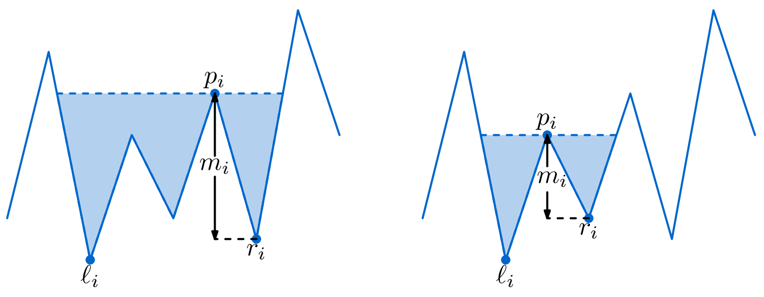


Figure 3 (left) The sublevel set component of p_i, below the dashed line segment. Points ℓ_i and r_i are the minima of the left and right parts of this component. (right) If p_i is incident to a shortest edge of P then m_i is the length of this edge.

Q' must contain an increasing edge if $p_i \leq p_j$ and a decreasing edge otherwise. For any three consecutive vertices p_i, p_j, p_h of $V^\varepsilon_{\text{orig}}$ we have that either $p_i \leq p_j$ or $p_j \leq p_h$, but not both, as P^ε alternates between minima and maxima. Hence the subcurves of Q matched to the subcurves of P between p_i and p_j, and between p_j and p_h, together contain at least two distinct edges, one for each orientation. There must therefore be at least $k-1$ distinct edges on Q, which together have at least k distinct vertices. ◀

▶ **Theorem 3.** *Let P be a curve in one dimension and let $k \geq 1$ be an integer. Let $\varepsilon \geq 0$ be the smallest value for which P^ε has at most k vertices. Then P^ε is a closest k-curve for P.*

Proof. Let Q be a curve with at most k vertices. Let $\varepsilon' = d_F(P,Q)$. By Theorem 2, the ε'-simplification of P has at most k vertices. Thus we obtain that $\varepsilon \leq \varepsilon' = d_F(P,Q)$. It follows from Theorem 1 that $d_F(P,P^\varepsilon) \leq \varepsilon \leq d_F(P,Q)$. ◀

3 Constructing simplifications in linear time

In this section we present a data structure for computing the simplifications of a curve. The data structure relies on computing the *death times* of the vertices of P. We define the death time of a vertex p_i of P to be the smallest value $\varepsilon \geq 0$ for which p_i is degenerate in P^ε.

We proceed to express the death time of a vertex in terms of the extreme values in its sub- or superlevel set component. The *sublevel set* of a point p on P is the set of points on P with value at most p. The *sublevel set component* of p is the connected component of its sublevel set that contains p, see Figure 3. The *superlevel set component* of p is defined symmetrically. For a local maximum p_i of P, let P^- be its sublevel set component. We define the points ℓ_i and r_i as (global) minima on the prefix and suffix curves of P^- that end and start at p_i, respectively. We let $m_i := \min\{|p_i - \ell_i|, |p_i - r_i|\}$, see Figure 3. We symmetrically define P^+ to be the superlevel set component of a local minimum p_i of P, and symmetrically define ℓ_i and r_i in terms of P^+. The definition of m_i is the same as for local maxima. We show that the death time of an interior vertex p_i is equal to $m_i/2$.

▶ **Lemma 4.** *For any $2 \leq i \leq n-1$, the death time of vertex p_i is equal to $m_i/2$.*

Proof. Let p_i be a vertex of P for some $2 \leq i \leq n-1$ and assume without loss of generality that p_i is a local maximum. We distinguish between the case where p_i is incident to a shortest edge of P and the case where no incident edge is a shortest edge.

First assume that p_i is incident to a shortest edge e of P and assume without loss of generality that $e = \overline{p_{i-1}p_i}$. The death time of p_i is equal to $\|e\|/2$, since truncating e by half

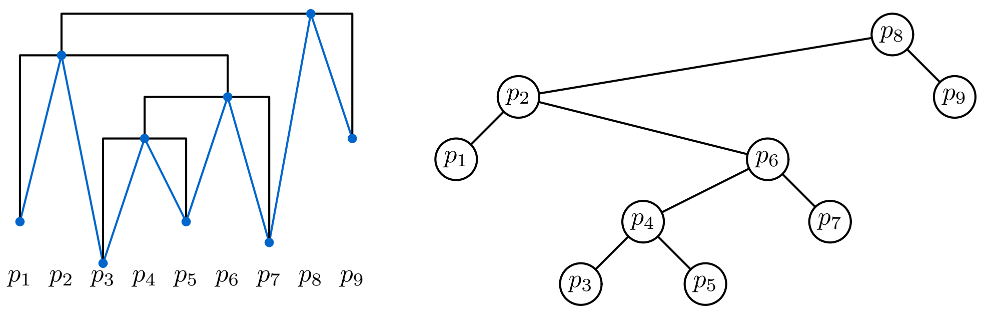


Figure 4 A max-Cartesian tree.

the minimum edge length truncates e into a point and p_i is not an endpoint of P. Observe that $m_i = \|e\|$. Indeed, because e is a shortest edge of P, we have that $p_{i-2} \geq p_i \geq p_{i-1}$ (if p_{i-2} exists) and $p_{i+1} \leq p_{i-1}$. Thus we obtain that $\ell_i = p_{i-1}$ and $r_i \leq \ell_i$, and hence $m_i = |p_i - \ell_i| = \|e\|$. See Figure 3. Hence the death time of p_i is $m_i/2$.

Next assume that p_i is not incident to a shortest edge of P. Let ε be equal to half the minimum edge length of P. Note that ℓ_i and r_i are both local minima of P and therefore vertices of P. As every local minimum of P gets increased by ε by the simplification, every local maximum gets decreased by ε, and the minimum edge length of P is 2ε, we obtain that the points $\ell_i^\varepsilon := \ell_i + \varepsilon$ and $r_i^\varepsilon := r_i + \varepsilon$ are the analogues of ℓ_i and r_i for the point p_i^ε, with respect to P^ε. It follows that m_i^ε, the analogue of m_i, is equal to $\min\{|p_i^\varepsilon - \ell_i^\varepsilon|, |p_i^\varepsilon - r_i^\varepsilon|\} = m_i - 2\varepsilon$. Applying the above recursively on the point p_i^ε, curve P^ε and value m_i^ε shows that the death time of p_i is $m_i/2$. ◀

With the expression m_i for the death times of vertices, we are able to compute the death time of every vertex in linear time. To this end we use *Cartesian trees*, introduced by Vuillemin [11]. A Cartesian tree is a tree with the heap property. We call a Cartesian tree a *max-Cartesian tree* if it has the max-heap property and a *min-Cartesian tree* if it has the min-heap property. A max-Cartesian tree T for a sequence of values $x_1, \dots, x_n$ is recursively defined as follows. The root of T contains the maximum value x_j in the sequence. The subtree left of the root node is a max-Cartesian tree for the sequence $x_1, \dots, x_{j-1}$, and the right subtree is a max-Cartesian tree for the sequence $x_{j+1}, \dots, x_n$ (see Figure 4). Min-Cartesian trees are defined symmetrically.

▶ **Lemma 5.** *We can compute the death time of every vertex of P in $O(n)$ time.*

Proof. To compute the death times of the vertices, we build two Cartesian trees; a max-Cartesian tree $T_{\max}$ and a min-Cartesian tree $T_{\min}$, both built on the sequence of vertices $p_1, \dots p_n$ of P. These trees can be constructed in $O(n)$ time [6].

For a given node v of $T_{\max}$ storing vertex p_i, the vertices stored in the subtree rooted at v are precisely those in the sublevel set component of p_i. Thus if p_i is a local maximum, the values ℓ_i and r_i are precisely the minimum values stored in the left and right subtrees of v, respectively. We can therefore compute the death times of the local maxima of P with a bottom-up traversal of $T_{\max}$, taking $O(n)$ time. Repeating the above process for $T_{\min}$, we compute the death times of the local minima of P in $O(n)$ time as well. ◀

Having computed the death times of the vertices, computing P^ε is merely a matter of removing vertices of P with a death time at most ε, decreasing the leftover local maxima by ε, and increasing the leftover local minima by ε. To identify the vertices present in the simplification, we store the vertices of P in another max-Cartesian tree, storing the vertices based on death time. The vertices with a death time greater than ε can be found in linear output-sensitive time by traversing the tree from the root. We obtain the following result.

▶ **Theorem 6.** *We can preprocess an n-curve P in $\mathbb{R}$ in $O(n)$ time, after which we can query it for the ε-simplification of P in $O(k)$ time for any $\varepsilon \geq 0$, where k is the output complexity.*

Using death times, we can extend this data structure to support queries for closest k-curves.

▶ **Theorem 7.** *We can preprocess an n-curve P in $\mathbb{R}$ in $O(n)$ time, after which we can query it for a closest k-curve for P in $O(k)$ time for any $k \geq 1$.*

Proof. We store the death times of P in a max-heap in $O(n)$ time. To compute a closest k-curve we proceed as follows. Let ε be the $(k+1)$-st greatest death time. We can compute ε in $O(k)$ time using the algorithm for selection in binary heaps by Frederickson [5]. Note that P^{ε} has at most k vertices and for any $\varepsilon' < \varepsilon$ that $P^{\varepsilon'}$ has more than k. Thus by Theorem 3, P^{ε} is a closest k-curve for P. We report P^{ε} in $O(k)$ time using Theorem 6. ◀

References

1 Sergey Bereg, Minghui Jiang, Wencheng Wang, Boting Yang, and Binhai Zhu. Simplifying 3d polygonal chains under the discrete Fréchet distance. In *Proc. 8th Latin American Symposium on Theoretical Informatics (LATIN)*, volume 4957, pages 630–641, 2008. doi:10.1007/978-3-540-78773-0_54.

2 Karl Bringmann and Bhaskar Ray Chaudhury. Polyline simplification has cubic complexity. *Journal of Computational Geometry*, 11(2):94–130, 2020. doi:10.20382/jocg.v11i2a5.

3 Erin Wolf Chambers, Elizabeth Munch, and Tim Ophelders. A family of metrics from the truncated smoothing of reeb graphs. In *Proc. 37th International Symposium on Computational Geometry (SoCG)*, volume 189, pages 22:1–22:17, 2021. doi:10.4230/LIPIcs.SoCG.2021.22.

4 Anne Driemel, Amer Krivosija, and Christian Sohler. Clustering time series under the Fréchet distance. In *Proc. 27th Annual Symposium on Discrete Algorithms (SODA)*, pages 766–785, 2016. doi:10.1137/1.9781611974331.ch55.

5 Greg N. Frederickson. An optimal algorithm for selection in a min-heap. *Information and Computation*, 104(2):197–214, 1993. doi:10.1006/inco.1993.1030.

6 Harold N. Gabow, Jon Louis Bentley, and Robert Endre Tarjan. Scaling and related techniques for geometry problems. In *Proc. 16th Annual ACM Symposium on Theory of Computing*, pages 135–143, 1984. doi:10.1145/800057.808675.

7 Zhilin Li. Digital map generalization at the age of enlightenment: a review of the first forty years. *The Cartographic Journal*, 44(1):80–93, 2007. doi:10.1179/000870407X173913.

8 Damian Merrick and Joachim Gudmundsson. Path simplification for metro map layout. In *Proc. Graph Drawing, 14th International Symposium*, volume 4372, pages 258–269, 2006. doi:10.1007/978-3-540-70904-6_26.

9 Mees van de Kerkhof, Irina Kostitsyna, Maarten Löffler, Majid Mirzanezhad, and Carola Wenk. Global curve simplification. In *Proc. 27th Annual European Symposium on Algorithms*, volume 144, pages 67:1–67:14, 2019. doi:10.4230/LIPIcs.ESA.2019.67.

10 Marc J. van Kreveld, Maarten Löffler, and Lionov Wiratma. On optimal polyline simplification using the Hausdorff and Fréchet distance. *Journal of Computational Geometry*, 11(1):1–25, 2020. doi:10.20382/jocg.v11i1a1.

11 Jean Vuillemin. A unifying look at data structures. *Communications of the ACM*, 23(4):229–239, 1980. doi:10.1145/358841.358852.

A Heuristic Algorithm for Maximal Contained Polyhedrons*

Vahideh Keikha

The Czech Academy of Sciences, Institute of Computer Science, Czech Republic
Department of Computer Science, University of Sistan and Baluchestan, Zahedan, Iran
keikha@cs.cas.cz

Abstract

Let $\mathcal{P}$ be a polyhedron with n vertices chosen from a $3D$ grid; our objective is to approximate the largest volume convex polyhedron that is inscribed in $\mathcal{P}$. We propose a heuristic algorithm for this problem that is extendable to higher dimensions and we analyze its efficiency in implementation. Our experimental studies hold promises of providing good efficiency in practice.

1 Introduction

In computational geometry, a simple polygon is a closed polygonal chain in the plane that does not intersect itself and has no holes, and a convex shape is the boundary of a convex set. Analogs to simple polygons, a simple polyhedron is a $3D$ shape with flat polygonal faces, straight edges, sharp corners, and no holes. A polyhedron with holes is a simple polyhedron minus the interiors of some other simple polyhedrons. First let $\mathcal{P}$ be a simple polygon of n vertices. The best-known algorithm for computing a convex polygon $Q \subseteq \mathcal{P}$ of the maximum area takes $O(n^7)$ time [2]. In $3D$ (indeed, any fixed dimension), computing the largest volume convex polytope[1] in a simple polyhedron is still solvable in polynomial time. A simple idea is determining the complete face-lattice of the polyhedron. After computing the triangulation of the face-lattice, one then can use the formula for the volume of a simplex; see Lemma 2 in [4]. However, computing the largest volume contained polytope in arbitrarily high dimensions becomes more complicated. Computing the volume of an arbitrary convex polyhedron itself is an NP-hard problem in arbitrary high dimensions [4], and also difficult to be arbitrarily approximated [1].

▶ Problem 1 (Largest Convex Polyhedron). Let $\mathcal{P}$ be a polyhedron (possibly with holes) with all vertices on the vertices of a $3D$ grid. Our objective is to approximate a largest-volume convex polyhedron Q inside $\mathcal{P}$.

See Fig. 1 for an illustration of a polyhedron constructed on a $3D$ grid, and Fig. 2(a,b) for an illustration of $\mathcal{P}$ and Q. Observe that the coordinates of the vertices of $\mathcal{P}$ cannot be arbitrary real-valued numbers, and are of the form $i \times a_k + c_k$, where i is an integer, a_1, a_2, a_3 are grid constants for the three dimensions, and c_1, c_2, c_3 are offset. The grid constants are the same for all three dimensions, that is, $a_1 = a_2 = a_3 = a$. The constants c_1, c_2, c_3 could be eliminated w.l.o.g. by appropriately translating the origin so that one can simplify the coordinates of all three dimensions to $i \times a$, where i is an integer. The orientation of the coordinate system is assumed to be given and fixed.

* Supported by the CAS PPPLZ grant L100302301 and the institutional support RVO: 67985807.
[1] We refer to [6] for the basic properties of convex polytopes.

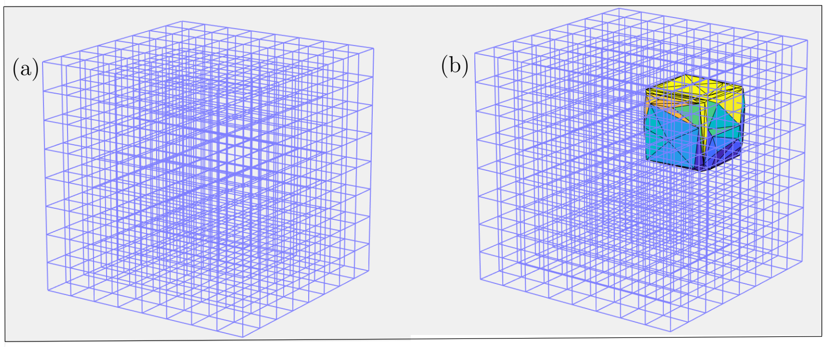


Figure 1 (a) A $3D$ grid. (b) A simple polyhedron with vertices chosen from a $3D$ grid that is decomposed into a set of convex pieces. Each of the convex pieces of the polyhedron is shown in a different color.

One of the main applications of our problem is high-clearance motion planning, in which the objective is computing a path for a convex robot contained in a non-convex environment from an initial position to a final position while remaining as far as possible from the boundaries of the environment. See, e.g., [3].

We introduce *Genetic Algorithm (GA)* as a heuristic method for solving our problem. For computing the volume of a polytope, we just need to find an interior point for decomposing the polytope into a set of pyramids. For the generalization of GA to input polyhedrons with holes, we use a point inclusion algorithm for approximating the volume of the polyhedrons. We simply count the number of grid points inside the polyhedron[2] as an approximation to the volume. This would relax us from having any restrictions on the number of holes and their complexity, which is a big challenge in the literature of our problem; see, e.g., [7]. We do not bound the error rate of this technique. Also, to the best of our knowledge, heuristic algorithms have not been studied in our problem. Hence, we did not find alternative approaches to make a comparison.

2 Genetic Algorithms (GA)

Genetic Algorithms (GA) are powerful stochastic search techniques that are applicable to a variety of non-convex optimization problems [9].

The basic idea of Genetic Algorithm [9] is that a *population* of candidate solutions (so-called *individuals*) to an optimization problem is evolved toward better solutions using successive *generations*, and with applying the basic operators *Crossover* and *Mutation*. Each candidate solution has a set of *genes* which can be mutated and altered.

The Genetic parameters are the population size and the number of generations. In a Genetic Algorithm, the candidate solutions (population) improve over the evolution of each generation. The evolution usually starts from a population of randomly generated individuals (also referred to as offsprings), and is an iterative process, with the population in each iteration called a generation. In each generation, the fitness of every individual in the population is evaluated; the fitness is usually the value of the objective function in

[2] But we do not count the grid points on an edge or a face for preventing some potential inaccuracies caused in these cases.

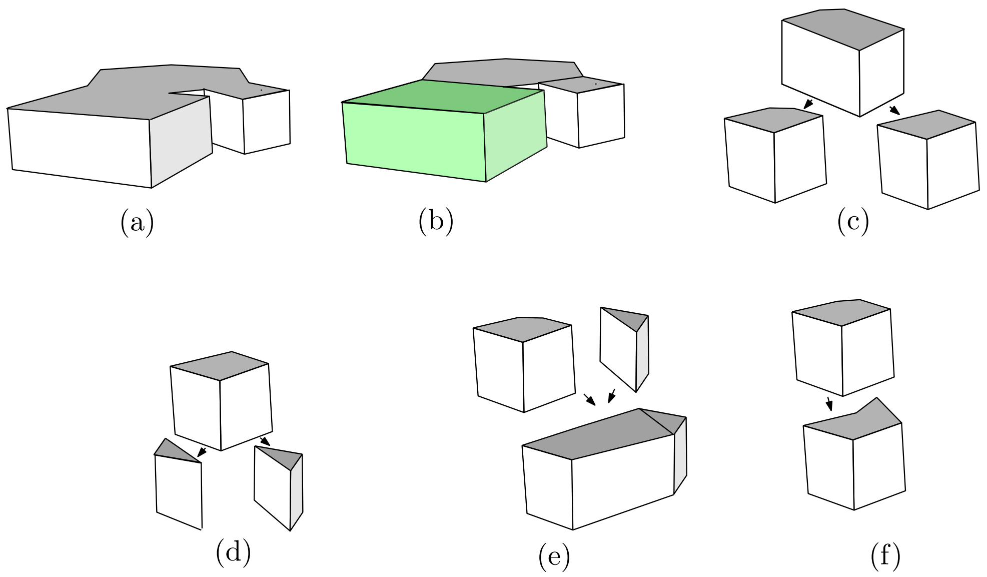


Figure 2 (a,b) Problem definition and optimal solution. The largest convex polyhedron of the input polyhedron is shown in green. (c,d) Cutting two polyhedrons for the crossover operation along z-axis. (e) Concatenating the cut polyhedrons for making a new offspring of a possibly larger volume (here a modification is done for convexity). Note that the result is the convex hull of the union of the polyhedrons. (f) Mutation of a vertex of polyhedron may result in the appearance of a non-convex vertex. Note that the resulting shape is not convex yet.

the optimization problem being solved. GAs operate on a population, which is a set of proposed solutions to the problem. Each proposed solution is called an individual (or a chromosome/offspring). Each chromosome is composed of a set of genes. GA's employ a set of biological evolutions to improve each proposed solution among a number of generations. The number of generations could be fixed, but usually would be determined by the desired value of the accuracy, based on a predefined threshold for the error.

3 Preparation for Experimental Studies

In this section, we introduce a formulation for a Genetic Algorithm for Problem 1 that is useful for higher dimensions and all population-based heuristic algorithms in similar problems.

We use the output of a convex decomposition of a polyhedron as the population in the first generation. During the evolution process, we keep it invariant that each individual lies entirely within $\mathcal{P}$. The individual of the largest volume is called *elite* of the generation.

We have implemented our algorithm in C++ with Visual Studio 2013. The algorithm is performed on a Core (TM) i9CPU and 8GB RAM computer with Windows 10 operating system. In some of the computations, we have used CGAL-5.1.

Initializing the first population As said before, before the initialization step, the input non-convex polyhedron is already generated uniformly at random from the vertices of a $3D$ grid. We then computed a convex decomposition of the input of a fixed number of m pieces (as the individuals). We use the output of a convex decomposition of a polyhedron from CGAL as the candidates of the population in the first generation. Note that if we have fewer convex pieces than the desired number, a convex decomposition can always be made to have a desired number of pieces, by simply splitting convex pieces. The genes of each individual are the set of its vertices. In the following, we describe the GA operators.

Crossover We select two polyhedrons and exchange half of their vertices from either the upper or lower hull. See Fig. 2(c,d) for an illustration of the decomposition process and Fig. 2(e) for the concatenation of two polyhedrons. Note that the resulting polyhedrons are intersecting, and each polyhedron is the convex hull of the new point sets. We exchange half of the top-most vertices of selected polyhedrons along the z-axis (could be also another arbitrary axis). To store and update a polyhedron and its set of vertices, we use existing data structures for adjacency lists for graph traversal.

Mutation Similar to the usual technique for mutation operator, our suggestion is to sample d positive real numbers uniformly at random in the domain of the points of the grid, where d is the number of the dimension of the problem, and find the closest gene (which is a vertex here) in the corresponding individual and substitute it with the selected random point. In other words, a vertex with the smallest Euclidean distance to the sampled point will be replaced. This would change some vertex of the mutated polyhedron by a random vertex. See Fig. 2(f) for an illustration.

Selection We keep the size of the population at a fixed number m throughout the algorithm. After performing the GA operators in each generation, the m polyhedrons which have the highest volume will determine the new population. From this number, the polyhedron of the highest volume (elite) is directly inserted into the new generation.

Modification of the individuals For each resulting individual from the GA operators, we may need to modify the resulting convex polyhedron to lie within $\mathcal{P}$. First, we compute the intersection of the offspring computed from the mutation or crossover and $\mathcal{P}$. If the intersection is empty, the offspring is valid, otherwise, we cut the offspring to fit it inside $\mathcal{P}$.

Evaluating fitness value The volume of each individual determines its fitness value. If the input polyhedron does not have any holes, the volume of each polytope can easily be computed using the decomposition of the polytope. Here we find the mean of all of its vertices, connect this point to all vertices of the polyhedron, and split the polyhedron into a set of pyramids. For polyhedrons with holes, we use a point inclusion algorithm for approximating the volume. For this purpose, we count the number of grid points within the polyhedron for computing the volume of a polyhedron, however, the efficiency of this algorithm highly depends on the resolution of the original grid. Also, we consider an identifier for each chromosome that would give a zero value if the polyhedron is only in $2D$. This can be done by storing the maximum difference between the height of the vertices of the polyhedron in different dimensions. In the following, we elaborate on the details.

For computing the number of the grid points inside each polyhedron Q, we simply perform a point-inclusion test for the grid points within all the cubes which lie within the bounding cube of Q. The convex polyhedrons can be preprocessed using the Dobkin-Kirkpatrick hierarchy technique so that the point-in-convex-polyhedron test takes $O(n)$ space and $O(\log n)$ query time [10] (Sec. 38.3). We then simply multiply the number of grid points (strictly) inside Q with an identifier that is either 0 or 1, where 0 is indicating Q is restricted in $2D$, and 1 is indicating Q is a proper polyhedron. So, we determine a 0 volume for the case where a polyhedron has its faces in the plane.

4 Experimental Results

Because of the exponential time complexity of our method (see [8]), the exact result could not be obtained on a large set of points in a reasonable computational time. Thus, we ran the GA on synthetic data of a reasonable size.

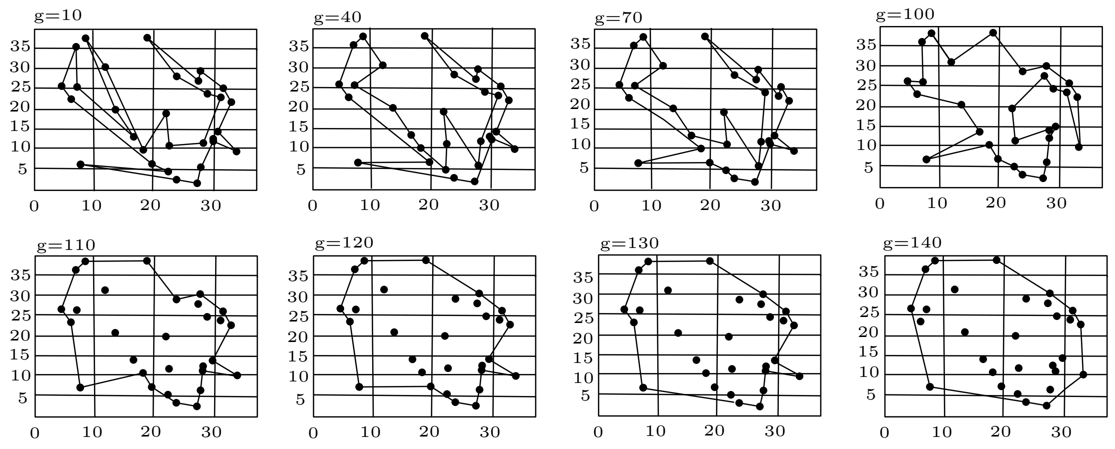


Figure 3 An illustration of the progress of GA over time: We have illustrated the changes on the vertices of an elite offspring of 40 vertices mapped in $2D$ by ignoring the third axis (note that some vertices might be missing during the mapping if they lie on the boundary), in different generations. The coordinates of the grid points are divided by 100. Observe that there is an inverse relationship between the number of reported vertices and the number of generations.

The mutation rate is the number of offspring produced in each generation over the population size, using the mutation operator. The crossover rate is defined analogously.

Here the rate of the mutation operation is chosen 0.2, and the rate of the crossover operation is chosen 0.6. We observed that our selected rate of mutation helps to escape from local minima at the beginning of the execution of the algorithm, and also prevents premature convergence, and handle the diversity of the population throughout the algorithm. For evaluating the efficiency of our algorithm in practice, we have implemented the GA and tested it on a $3D$ grid G in the range $[0, 4000]$. The simple polyhedron P has $40, 70$ or 100 vertices selected uniformly at random from G. Constructing each polyhedron is done by first sampling vertices of convex polyhedrons from the grid and then merging the resulting convex polyhedrons to achieve a certain number of vertices (using any union algorithm [5]). It may need several steps but still takes a constant time for a fixed number of vertices. We run each test to at most 160 generations. Also, we stop the algorithm whenever we find an offspring convex polyhedron of volume 70% of the volume of $\mathcal{P}$. These bounds are chosen arbitrarily and can be increased if more accurate approximations are required. One always can allow the GA to run for a longer time, to see the progress of the algorithm with more generations. We observed that in 160 generations, we usually have found some convex polyhedron that has at least 70% of the volume of the input polyhedron if such a polyhedron exists. Experimental results of the Genetic Algorithm on synthetic point sets of sizes 40, 70, and 100 are shown in Table 1. The polyhedrons of the dataset of size 100 have one hole of s vertices chosen uniformly at random with $d \leq s \leq 30$, where d is the dimension of the problem.

The reported time in the GA time column of each dataset determines the so far elapsed time. See Fig. 3 for an illustration of the progress of GA over time.

The third column of Table 1 demonstrates the number of vertices of the current elite polyhedron in the generation demonstrated in the second column. The fifth column demonstrates the ratio of the volume of the elite polyhedron (in the current generation) and the volume of the largest contained ball in $\mathcal{P}$, and the last column demonstrates the ratio of the volume of the current elite polyhedron and the volume of $\mathcal{P}$. Although the time required to solve an optimization problem with meta-heuristic techniques might be long, our reported running

Input size	# of generation	# of vertices of elite polyhedron	GA Time (s)	Vol(elite polyhedron)/ Vol(largest contained ball)	Vol(elite polyhedron)/ Vol($\mathcal{P}$)
40	40	29	2.3867	1.3685	0.2107
	100	28	4.8967	1.6557	0.4238
	120	17	6.0163	1.7657	0.6214
	143	11	6.4162	1.8514	0.7123
70	40	56	3.5223	1.5114	0.1125
	100	42	5.9185	2.3352	0.5689
	120	36	6.1945	2.4732	0.6351
	151	31	6.9790	2.5647	0.7418
100	40	79	4.5421	2.4236	0.2895
	100	68	6.9928	2.5930	0.5872
	120	61	8.2568	2.6638	0.6253
	158	57	9.1644	2.7458	0.6942

Table 1 Summary of the experiment results.

times are reasonable concerning the number of vertices, even on an average hardware setup. However, the time required to solve the problem may increase with the scale of the problem.

5 Discussion

Considering the theoretical aspects of our problem remained open. One open question is finding some quick and efficient approximation algorithms for computing the biggest convex polyhedron in arbitrary polyhedrons in our problem formulation in arbitrarily high dimensions. This would give a baseline to evaluate the fitness of the individuals properly. One can use this value to stop further generations when the individuals approach the optimal value. We believe our results besides the introduced dataset give a baseline for future studies.

References

1 Imre Bárány and Zoltán Füredi. Computing the volume is difficult. *Discrete & Computational Geometry*, 2(4):319–326, 1987.

2 Jyun-Sheng Chang and Chee-Keng Yap. A polynomial solution for the potato-peeling problem. *Discrete & Computational Geometry*, 1(2):155–182, 1986.

3 Paul Chew and Klara Kedem. A convex polygon among polygonal obstacles: Placement and high-clearance motion. *Computational Geometry*, 3(2):59–89, 1993.

4 Martin E. Dyer and Alan M. Frieze. On the complexity of computing the volume of a polyhedron. *SIAM Journal on Computing*, 17(5):967–974, 1988.

5 Herbert Edelsbrunner. The union of balls and its dual shape. In *Proc. ninth annual Symposium on Computational geometry*, pages 218–231, 1993.

6 Martin Henk, Jürgen Richter-Gebert, and Günter M Ziegler. *Basic properties of convex polytopes*. Chapman and Hall/CRC, 2017.

7 Nilanjana Karmakar and Arindam Biswas. Construction of an approximate 3D orthogonal convex skull. In *Proc. International Workshop on Computational Topology in Image Context*, pages 180–192, 2016.

8 Fernando G Lobo, David E Goldberg, and Martin Pelikan. Time complexity of genetic algorithms on exponentially scaled problems. In *Proceedings of the 2nd annual conference on genetic and evolutionary computation*, pages 151–158, 2000.

9 Zhang Qing-feng. The application of genetic algorithm in optimization problems. *Journal of Shanxi Normal University*, pages 1–8, 2014.

10 Jack Snoeyink. Point location. In *Handbook of discrete and computational geometry*, pages 1005–1028. Chapman and Hall/CRC, 2017.

Crossing Optimization in Neighborhood Drawings

Patricia Bachmann[1] and Ignaz Rutter[1]

1 **University of Passau**
`{bachmanp, rutter}@fim.uni-passau.de`

Abstract

We consider the problem of modifying a given straight-line drawing Γ of a graph G so that it emphasizes readability of the neighborhood of a given vertex v while preserving the user's mental map. To model the mental map preservation, we seek a *neighborhood drawing*, i.e., a straight-line drawing Γ' of the graph $G[v]$ induced by the neighborhood of v that has v at the same position as in Γ and all other vertices of $G[v]$ may only be moved on the ray from v to their position in Γ. Our objective is to find a neighborhood drawing with few crossings.

We give an algorithm for testing whether a vertex v has a neighborhood drawing without crossings. We further show that neighborhood drawings with minimum skewness can be computed efficiently. Finally, we evaluate the crossing reduction in practice and showcase some examples.

1 Introduction

Whether a particular graph layout is informative or useful strongly depends on the specific task that it is used for [4, 5]. While there exist approaches that focus on visualizing the neighborhood of specific vertices, e.g., Da Lozzo et al. [2] consider the task of visualizing the neighborhood of a given vertex using limited screen space, most general-purpose layout algorithms such as force-directed methods [1] are designed to give a good overview of the overall structure of a graph. This focus on the global structure leads to suboptimal results in the visualization of more local structures such as the neighborhood of a particular vertex.

We study the problem of visualizing the neighborhoods of specific vertices in a way that allows the user to understand the neighborhood of a vertex with the context of a drawing of the entire graph. To this end, we assume that our input consists of a graph $G = (V, E)$ together with a geometric drawing Γ of G and a particular vertex v of G. The *neighborhood graph* of v in G, denoted by $G[v]$, is the subgraph of G induced by the neighborhood of v. The drawing Γ induces a drawing $\Gamma[v]$ of $G[v]$. We seek a new drawing Δ of $G[v]$ such that Δ is more readable than $\Gamma[v]$ and yet sufficiently similar to it. This motivates the following problem. Given a graph G with a straight-line drawing Γ of G and a *center vertex* v, we call a drawing Δ of $G[v]$ a *neighborhood drawing* if $\Gamma(v) = \Delta(v)$ and for all neighbors u of v there exists $c > 0$ such that $\Gamma(v) - \Gamma(u) = c \cdot (\Delta(u) - \Delta(v))$, i.e. the position of u in Δ lies on the ray from v through u in Γ. We may assume w.l.o.g. that Γ maps v to the origin. In particular, the order of the edges incident to v is the same in $\Gamma[v]$ and in Δ, and the angles between consecutive edges incident to v are preserved, see Figure 1 for an example.

In this paper we initiate the study of the problem of optimizing crossings in neighborhood drawings. In particular, we characterize the neighborhoods that admit planar drawings, we give an efficient algorithm for computing a neighborhood drawing with minimum skewness, and we experimentally evaluate our approach in terms of the crossing reduction.

2 Planar Neighborhood Drawings

Let $G[v]$ be a neighborhood graph. We call the edges of G that are incident to v *spine edges*. An edge of $G[v]$ not incident to v is called *peripheral edge*. Note that, since $G[v]$ is

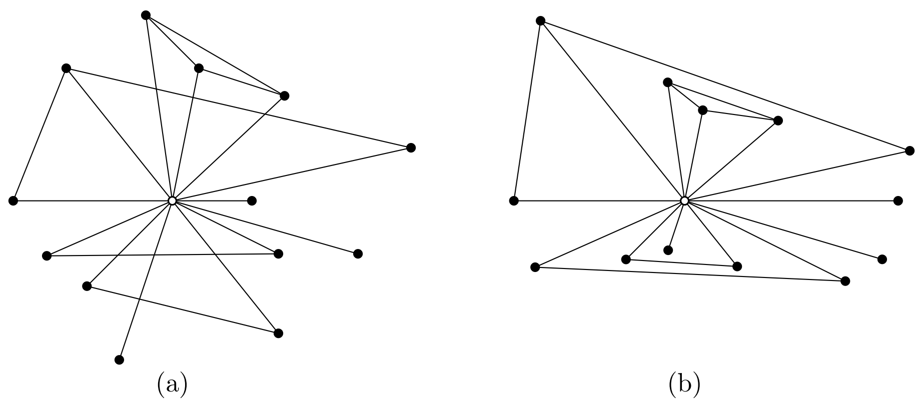


Figure 1 (a) Neighborhood graph $G[v]$. (b) Crossing-free neighborhood drawing Δ of $G[v]$.

a neighborhood graph, each peripheral edge $e = xy$ determines a 3-cycle $C_e = xyv$ in $G[v]$, which is represented as a (geometric) triangle in $\Gamma[v]$. Two peripheral edges uw and xy *alternate* if their endpoints alternate in the circular order around v determined by Γ. The goal of this section is to prove that a neighborhood graph admits a planar neighborhood drawing if and only if there are no alternating 3-cycles. We begin with a preparatory lemma.

▶ **Lemma 2.1.** *Let $G[v]$ be neighborhood graph that admits a planar neighborhood drawing Δ. Then, for any vertex u of the outer face of $G[v]$, the drawing Δ' obtained by moving u along its ray away from v is a planar neighborhood drawing.*

Proof. Let Δ' be the drawing obtained by moving u along its ray away from v. We can only cause crossings on edges incident to u. Assume for the sake of contradiction that uw crosses an edge xz in Δ'. Then w.l.o.g. uw passes the vertex x such that x now lies between uw and v, see Figure 2. But then the edge xv would have crossed uw in Δ, a contradiction. ◀

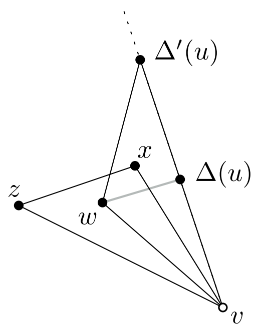


Figure 2 If moving u creates a crossing in Δ' then Δ is not planar.

▶ **Theorem 2.2.** *A neighborhood graph $G[v]$ admits a planar neighborhood drawing if and only if no two of its peripheral edges alternate.*

Proof. Assume that uw and xy are alternating peripheral edges and, for the sake of contradiction, that Δ is a planar neighborhood drawing of $G[v]$. We derive a contradiction by showing that Δ contains a crossing. Consider the triangle $T_{uw} = \Delta[C_{uw}]$. Since uw and xy alternate and since neither vx nor vy may cross uw, it follows that the two vertices x and y

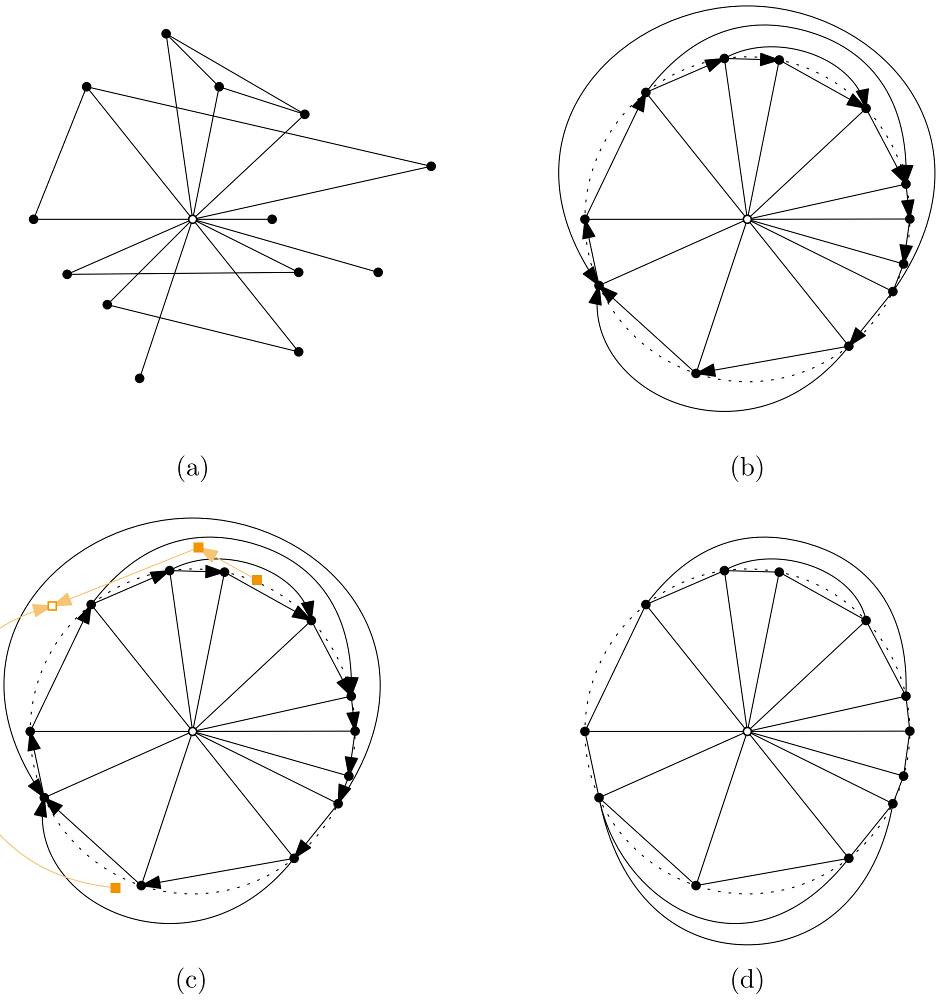


Figure 3 Constructing $\mathcal{E}$ by Theorem 2.2. (a) Neighborhood graph $G[v]$. (b) Clockwise oriented drawing of $G[v]$. (c) Auxiliary graph H, vertex corresponding to f_0 highlighted. (d) Drawing of $\mathcal{E}$.

lies on different sides of T_{uw} in Δ. But then the edge xy crosses one of the edges that form its boundary; a contradiction.

Conversely, assume that $G[v]$ has no alternating peripheral edges. Note that we can add the peripheral edges between any two spine edges that are consecutive around v, if they are not already in $G[v]$, without creating alternating peripheral edges. In the following we assume that $G[v]$ has been augmented in this way. Then $G[v]$ contains a wheel with center v whose outer edges are the peripheral edges between consecutive edges around v; see Figure 3(b). Since this wheel is triconnected, it has a unique planar embedding where the cycle of the peripheral edges lies on the outer face. We now add the remaining peripheral edges of $G[v]$ in a planar way in the outer face. Note that if this is not possible, then two of these peripheral edges cross, which implies that they alternate. Hence $G[v]$ is planar, and since it contains the 3-connected wheel as a spanning subgraph, its planar embedding is unique up to reversal and the choice of the outer face. We choose its embedding $\mathcal{E}$ such that the rotation around v coincides with the order of the rays emanating from v in Γ.

Each peripheral edge $e = uw$ defines two angles at v that sum to 2π. Assuming that no two rays are exactly opposite of each other, one of them is less than π, and we direct e in such a way that the smaller angle lies to its right. We call this orientation a *clockwise orientation*. Note that each face f of $\mathcal{E}$ that is bounded only by peripheral edges can be incident to at most one edge e such that f lies right of e; otherwise the angles around v would sum to more than π. Moreover, there exists a unique face f_0 that is bounded only by peripheral edges such that f_0 lies to the left of all its incident edges. This can be seen as follows. Consider the auxiliary graph H obtained by creating a vertex for each face that is bounded only by peripheral edges and so that a face f has a directed edge to a face g if and only if there is an edge that has f to its right and g to its left. Observe that, since G is outerplanar after removing v with all its incident edges, H is a tree. By the above observation, every vertex has out-degree at most 1. It then follows that H has a single sink, which corresponds to the desired face f_0. We take f_0 as the outer face of $\mathcal{E}$.

We now show that $G[v]$ admits a planar neighborhood drawing with embedding $\mathcal{E}$. We first gradually delete the peripheral edges that lie on the outer face. We do this until we get a drawing of $G[v]$ containing only the wheel which admits a straight line drawing Δ. We now draw the peripheral edges in Δ in the reverse order in which we deleted them. This way a peripheral edge e is added to Δ after another peripheral edge e' if and only if e' is nested in e. To draw a peripheral edge uw in Δ we move the respective endpoints away from v such that uw can be drawn as a straight line without creating any crossings. By Lemma 2.1 we know that moving u, resp. w, away from v does not create any crossings in Δ. Further, to determine how far to move the two endpoints we consider vertices on the outer face that form a path from u to w following the angle $< \pi$ formed at v, see Figure 4(a). From those, we find a vertex x furthest away from v. Since these vertices are not on the

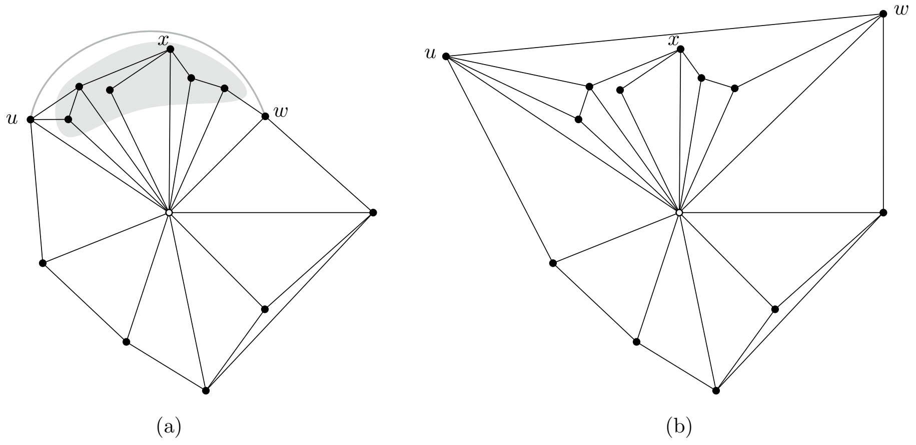


Figure 4 The vertices used to determine the new positions of u and w are highlighted in (a).

outer face after drawing the edge uw, they will never be visited again when drawing the remaining peripheral edges. Hence, we visit each vertex of $G[v]$ at most once while drawing peripheral edges, resulting in a linear runtime.

◀

3 Crossing-Minimization Heuristic

There exists a class of graphs that show that the number of crossings can grow arbitrarily large for any number of alternating 3-cycles. These graphs can be constructed as follows: Let abc and def be two alternating 3-cycles with $a = d = v$. We now add vertices x_i, y_j,

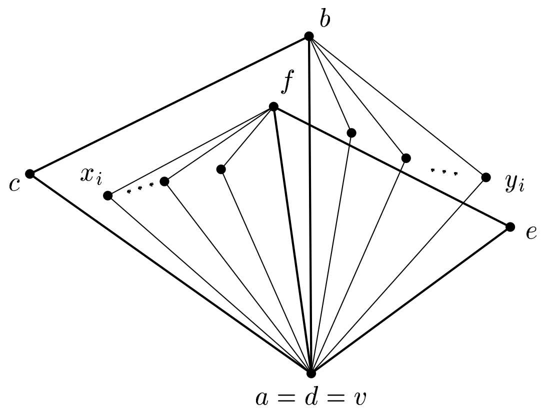


Figure 5 A graph with arbitrarily many crossings and only two alternating 3-cycles.

$i, j \in \mathbb{N}$ as follows: vertices x_i are added to G between c and f in the ordering around v, vertices y_j are inserted between b and e. All x_i and y_j are adjacent to v. Additionally, all x_i are adjacent to f, whereas all y_j are adjacent to b. By construction, either edge bc crosses $i + 1$ edges (and ef none) or ef crosses $j + 1$ edges (and bc none). Since we can add arbitrarily many such triangles vx_if (resp. vy_jb), the number of crossings can be arbitrarily high, even for only two alternating 3-cylces. An example of such a graph can be seen in Fig. 5. This construction proves the following theorem.

▶ **Theorem 3.1.** *There exists no function $f : \mathbb{N} \to \mathbb{N}$ such that for a neighborhood graph G with k alternating 3-cycles the number of crossings is $f(k)$.*

Since the number of alternating 3-cycles does not necessarily influence the number of crossings in a neighborhood drawing, we instead present a heuristic focusing on the *skewness* of the neighborhood graph. That is, we want to find the minimum number of peripheral edges such that their removal results in a planar neighborhood graph. For this planar neighborhood graph we can compute the crossing free drawing in linear time. We show that computing this minimum number of peripheral edges can be done in polynomial time by solving VERTEX COVER on the circle graph formed by alternating peripheral edges.

▶ **Theorem 3.2.** *Let $G = (V, E)$ be a neighborhood graph with center v. A straight-line neighborhood drawing Δ of G with minimal skewness can be computed in polynomial time.*

Proof. Consider a straight-line drawing Δ of peripheral edges of G such that each endpoint lies on the correct ray at unit-distance from v; see Fig. 6 (a). Then two peripheral edges intersect in their interior if and only if they alternate. Hence, this drawing Δ is an intersection representation of the conflict graph G_C describing the alternations, where each vertex of G_C is represented by a chord in Δ (we can move the endpoint in such a way that chords sharing an endpoint do not cross, and no new crossings are produced). Thus, the conflict graph G_C is a circle graph. A minimum set of peripheral edges that resolves all alternating peripheral edges is a vertex cover in this graph. For circle graphs there exists an algorithm that computes a maximum independent set in $O(n \min\{d, \alpha\})$ where α is the independence number of the circle graph and d its density [3]. By slightly modifying this algorithm we can compute a vertex cover in polynomial time for our conflict graph. ◀

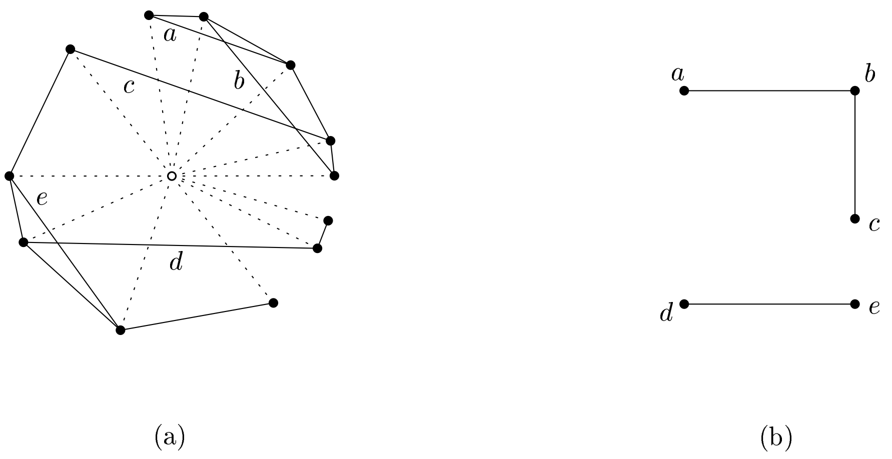


Figure 6 (a) Peripheral edges in Δ. (b) Conflict graph G_C (without isolated vertices).

3.1 Experimental Evaluation

To evaluate the effectiveness of the heuristic from Theorem 3.2 for the purpose of crossing reduction in neighborhood drawings, we implemented the algorithm and evaluate it in terms of the crossing reduction and the skewness of the produced neighborhood drawings.

The implementation uses mostly JavaScript with the library D3.js, and Python 3.9 as well as Sagemath, which was used to compute the vertex cover of the conflict graph. For the evaluation we considered 237 neighborhoods of 15 different graphs: 11 randomly generated graphs with a fixed number of vertices and edges, where each edge is present with the same probability p and four graphs published in the visone wiki[1].

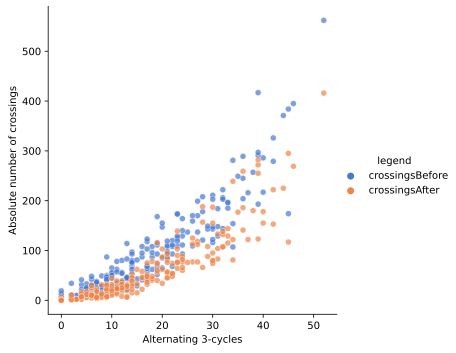
Crossing Minimization. To evaluate the performance in terms of crossing minimization, we compared for each neighborhood the number of crossings in the initial drawing with the number of crossings after applying our heuristic. Figure 7(a) shows that in almost all cases our algorithm reduces the number of crossings and that overall, despite Theorem 3.1, the number of crossings seem to be correlated with the number of alternating triangles. As Figure 7(b) shows, with an increasing number of alternating 3-cycles, our heuristic becomes less effective at reducing crossings. The crossing reduction is around 45% on average, although for neighborhood drawings with up to 10 alternating 3-cycles this number is around 56% and a little over 50% for neighborhood drawings with up to 20 alternating 3-cycles; see Table 1. The distribution of percentages of reduced crossings is illustrated in Figure 8(a).

# alternating 3-cycles	Avg. crossing reduction	Avg. # edges contributing to skewness
≤ 10	56.41%	10.41%
≤ 20	50.56%	14.68%
≤ 52	44.97%	19.79%

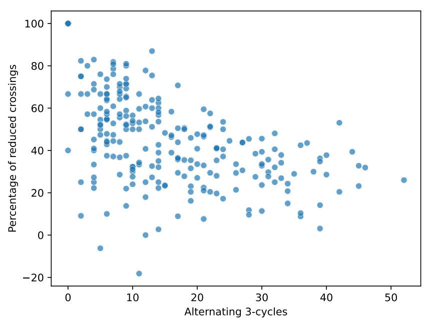
Table 1 Percentage of reduced crossings on average for different numbers of alternating 3-cycles.

Notably, for a small number of instances, namely two, our heuristic increases the number of crossings, albeit by a very small number, i.e. by six and one crossing, respectively.

[1] https://visone.ethz.ch/wiki/index.php/Knecht_Classroom_(data)

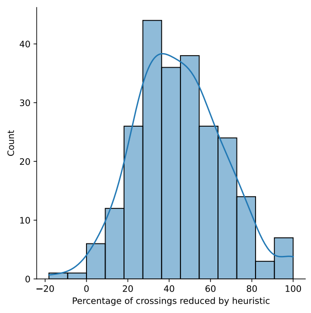


(a)

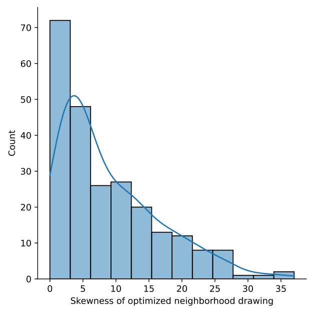


(b)

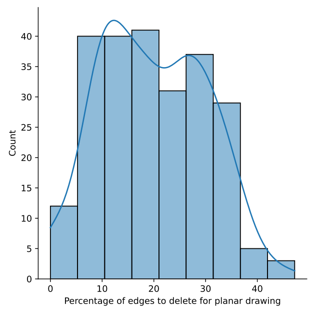
Figure 7 (a) Crossing number in neighborhood drawings before and after optimizing. (b) Percentages of reduced crossings tend to decrease as the number of alternating 3-cycles increase.



(a)



(b)



(c)

Figure 8 Distribution of reduced crossings (%), skewness and edges adding to skewness (%).

Skewness. We also investigated the skewness of our optimized neighborhood drawings. Figure 8(b) and (c) show the distributions of both the skewness and the percentage of edges that contribute to said skewness. Our evaluation shows that on average around 20% of edges in an optimized neighborhood drawing contribute to the skewness of the neighborhood graph. This percentage decreases with fewer alternating 3-cycles in the neighborhood drawing, see Table 1. For 20 or less alternating 3-cycles, around 14% of edges contribute the the skewness and for 10 or less alternating 3-cycles the number is around 10%.

3.2 Discussion

Our evaluation shows that our heuristic for crossing minimization works well for neighborhood graphs with few alternating 3-cycles, namely up to and including 20 alternating 3-cycles, in that it reduces the number of crossings by at least half and only around 14% of edges contribute to the skewness of the neighborhood graph. However, for more alternating 3-cycles the effectiveness of crossing minimization decreases to less than half on average while almost 20% of the edges of the neighborhood graph on average contribute to its skewness.

In practice, displaying neighborhood drawings also becomes problematic for some cases. That is, the vertices of a triangle uvw are moved further away from the center v the larger the angle at v in order to eliminate the crossings caused by the vertices that lie inside of $\Delta(C_{uw})$. As a result, vertices are often moved beyond the edge of the screen, see Figure 9.

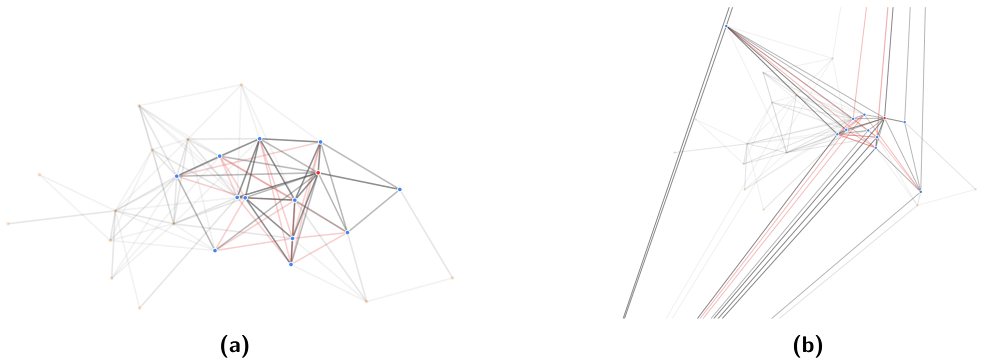


Figure 9 Neighborhood drawing (a) before and (b) after transformation.

4 Conclusion

We have shown that a neighborhood graph admits a crossing-free drawing Δ if and only if it contains no alternating peripheral edges. We can compute Δ in linear time. For general graphs, we give a heuristic to compute a neighborhood drawing with minimal skewness. Our application complements the theoretical results and, on average, reduces the crossings of neighborhood drawings by 45% and even 50% for graphs with few alternating 3-cycles. It also shows that, on average, removing 20% of the edges from optimized neighborhood drawings results in a crossing-free drawing. This result also improves for neighborhood graphs with fewer alternating 3-cycles.

The next step is to investigate so-called *open* neighborhood drawings. In such drawings, we omit spine edges and only keep peripheral edges. This ensures that alternating peripheral edges no longer create crossings on spine edges. An interesting result would be to show if open neighborhood drawings always omit a crossing-free straight-line drawing.

References

1 Thomas M. J. Fruchterman and Edward M. Reingold. Graph drawing by force-directed placement. *Software: Practice and Experience*, 21(11):1129–1164, nov 1991. `doi:10.1002/spe.4380211102`.

2 Giordano Da Lozzo, Giuseppe Di Battista, and Francesco Ingrassia. Drawing graphs on a smartphone. *J. Graph Algorithms Appl.*, 16(1):109–126, 2012. `doi:10.7155/jgaa.00252`.

3 Nicholas Nash and David Gregg. An output sensitive algorithm for computing a maximum independent set of a circle graph. *Information Processing Letters*, 110(16):630–634, jul 2010. `doi:10.1016/j.ipl.2010.05.016`.

4 Helen C. Purchase, David Carrington, and Jo-Anne Allder. *Empirical Software Engineering*, 7(3):233–255, 2002. `doi:10.1023/a:1016344215610`.

5 Helen C. Purchase, Robert F. Cohen, and Murray James. Validating graph drawing aesthetics. In *Graph Drawing*, pages 435–446. Springer Berlin Heidelberg, 1996. `doi:10.1007/bfb0021827`.