CGWeek Young Researchers Forum 2019

Booklet of Abstracts

2019

This volume contains the abstracts of papers at "Computational Geometry: Young Researchers Forum" (CG:YRF), a satellite event of the 35th International Symposium on Computational Geometry, held in Portland, Oregon on June 18-21, 2019.

The CG:YRF program committee consisted of the following people:

There were 20 papers submitted to CG:YRF. Of these, 16 were accepted with revisions.

Ellipsoidal Voronoi Diagrams

Ahmed Abdelkader
Department of Computer Science
University of Maryland, College Park MD, USA

David M. Mount
Department of Computer Science and Institute of Advanced Computer Studies
University of Maryland, College Park MD, USA

1 Introduction

Following a long series of papers, Arya, da Fonseca, and Mount [2] recently presented a breakthrough result by showing that it is possible to answer approximate nearest-neighbor searching (ε-ANN) queries in time $O_d(\log(n/\varepsilon))$ with storage of only $O_d(n/\varepsilon^{d/2})$, where d is assumed to be constant and O_d hides multiplicative factors exponential in d. This roughly halves the exponent in the storage bound compared to the approximate Voronoi diagram (AVD) first introduced by Har-Peled. This result was enabled by an entirely new approach to polytope approximation based on a classical concept from convexity theory called *Macbeath regions*. In this abstract, we present an intrinsic approach based on covering space by hierarchies of ellipsoids which are sensitive to the distance function, successfully bypassing the explicit reduction to polytope approximation in $\mathbb{R}^{d+1}$ through the lifting transform. Our approach applies to Bregman divergences defined by well-conditioned generator functions (which generalize the (squared) Euclidean distance), matching state-of-the-art results for this class of distances, and provides space-time trade-offs matching and extending state-of-the-art results for the Euclidean distance.

▶ **Theorem 1.** *Given a set P of n points in $\mathbb{R}^d$, an approximation parameter $0 < \varepsilon \leq 1$, and m such that $\log \frac{1}{\varepsilon} \leq m \leq 1/(\varepsilon^{d/2} \log \frac{1}{\varepsilon})$, there is a data structure that can answer ε-approximate nearest neighbor queries under well-conditioned Bregman divergences with*

$$\textit{Query time: } O_d\left(\log n + \frac{1}{m \cdot \varepsilon^{d/2}}\right) \quad \textit{and} \quad \textit{Space: } O_d(nm)\,.$$

2 Distance-based Macbeath regions

Given a σ-smooth μ-convex function F, the associated Bregman divergence is defined as $D_F(q,p) = F(q) - (F(p) + \langle \nabla F(p), q-p \rangle)$. We generalize the Delone sets approach for approximating convex bodies [1] (see Fig. 1(a)) to the approximation of (Bregman) Voronoi diagrams as follows. Define the *δ-expanded (Bregman) Voronoi cell* of p as $V_\delta(p) = \{x \in \mathbb{R}^d : D_F(x,p) \leq D_F(x,p') + \delta^2, \ \forall p' \in P\}$, where $\delta \geq 0$ is the *expansion factor* (see Fig. 1(b)).

Letting $\mathrm{nn}(x)$ denote the nearest neighbor of x in P, define the *distance-based Macbeath region* $M_\delta(x) = x + (K_\delta - x) \cap (x - K_\delta)$ where K_δ is taken as the expanded Voronoi cell $V_\delta(\mathrm{nn}(x))$ (see Fig. 1(c)), and define $M_\delta^\lambda(x)$ to be a central scaling of $M_\delta(x)$ by a factor of $\lambda \geq 0$. As in [1], define the *distance-based Macbeath ellipsoid*, denoted $E_\delta(x)$, to be the maximum volume ellipsoid contained within $M_\delta(x)$, and define its scaling $E_\delta^\lambda(x)$ analogously.

Given a query region w, we select a maximal set of points whose suitably shrunken distance-based Macbeath ellipsoids are pairwise disjoint, and then show that a suitable constant-factor expansion of these ellipsoids cover w. To ensure accuracy for ε-ANN queries, the ratio between the expansion value δ and the nearest-neighbor distance must be sufficiently small relative to $\sqrt{\varepsilon}$. We obtain the following bound on the number of such ellipsoids.

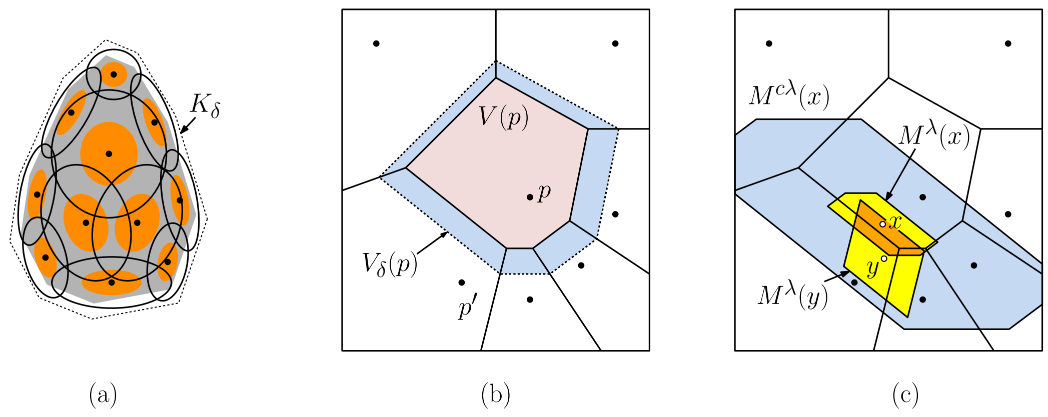


Figure 1 (a) Cover K_δ by Macbeath ellipsoids. (b) $V_\delta(p)$. (c) Distance-based Macbeath regions.

▶ **Lemma 2.** *Consider a point set P, a query region w, and $\gamma = O(1/\varepsilon)$ for some $\varepsilon > 0$ such that P and w are concentrically γ-separated, i.e., there exists a Euclidean ball $B(c, r)$ of radius r centered at some $c \in \mathbb{R}^d$ such that either:*

(a) $w \subseteq B(c, r)$, and $P \cap B(c, \gamma r) = \emptyset$, or *(b) $P \subseteq B(c, r)$ and $w \cap B(c, \gamma r) = \emptyset$.*

For any positive constant λ, let X be a maximal set of points lying within w such that the ellipsoids $E_\delta^\lambda(x)$ are pairwise disjoint, where $\delta = \gamma\sqrt{\varepsilon} \cdot r_b\sqrt{\mu/8}$. Then $|X| = O_d\big((\frac{1}{\gamma\varepsilon})^{d/2}\big)$.

3 The Ellipsoidal Voronoi Diagram (EVD)

The top level of our data structure is a balanced quadtree subdivision as in the AVD of [2]. Each leaf cell w of this structure is associated with a data structure that answers ε-ANN queries for any point within the cell. We apply Lemma 2 to design a generic data structure, which we call the EVD, that can be applied whenever the query region w and data points P are concentrically 2-separated, as can be ensured for the leaf cells of the AVD.

Intuitively, each successive level of the EVD from the root down involves an ellipsoid cover based on exponentially smaller expansions δ_i, implying that the representative of the ellipsoid containing the query point is a successively better approximation to its nearest neighbor. When the search procedure terminates, the representative of the last node visited will be an ε-ANN of the query point. The query time depends on the product of the number of levels ℓ (which is $O(\log(1/\varepsilon))$) and the maximum out-degree of each node (which is $O_d(1)$). The storage requirements are proportional to the total number of ellipsoids in all levels (which is $O_d(1/\varepsilon^{d/2})$). It follows that for each leaf of the AVD, the associated EVD data structure has space $O_d(1/\varepsilon^{d/2})$ and can answer ε-ANN queries in $O(\log 1/\varepsilon)$ time through a simple descent of the structure. By attaching one of these data structures to each of the $O_d(n)$ leaves of the AVD, we can answer ε-ANN queries for Bregman divergences generated by well-conditioned functions in time $O_d(\log(n/\varepsilon))$ and space $O_d(n/\varepsilon^{d/2})$.

References

1 A. Abdelkader and D. M. Mount. Economical Delone sets for approximating convex bodies. In *Proc. 16th Scand. Workshop Algorithm Theory*, pages 4:1–4:12, 2018.

2 S. Arya, G. D. da Fonseca, and D. M. Mount. Optimal approximate polytope membership. In *Proc. 28th Annu. ACM-SIAM Sympos. Discrete Algorithms*, pages 270–288, 2017.

On tree-like abstract Voronoi diagrams in expected linear time

Kolja Junginger
Faculty of Informatics, USI Università della Svizzera italiana,
Lugano, Switzerland
kolja.junginger@usi.ch

Evanthia Papadopoulou
Faculty of Informatics, USI Università della Svizzera italiana,
Lugano, Switzerland
evanthia.papadopoulou@usi.ch

Abstract

We present expected linear-time constructions for certain tree-like abstract Voronoi diagrams, building upon the concept of a *Voronoi-like diagram* [7] for a *boundary curve* in an *admissible* bisector system $\mathcal{J}$. We prove that the Voronoi-like diagram of a boundary curve on $\mathcal{J}$ always exists. Further, we compute the order-(k+1) subdivision within an order-k abstract Voronoi region in expected time linear in the complexity of the region's boundary.

2012 ACM Subject Classification Theory of computation → Computational geometry

Keywords and phrases Abstract Voronoi diagram, linear-time algorithm, higher order Voronoi diagram, randomized incremental algorithm

Digital Object Identifier 10.4230/LIPIcs.CVIT.2016.23

Funding Supported in part by the Swiss National Science Foundation, project SNF-200021E-154387

Extended abstract

For certain Voronoi diagrams with a tree structure linear-time algorithms have been well known to exist, e.g., [1, 5, 10, 6]. The basic framework was designed by Aggarwal et al. [1] for the Voronoi diagram of points in convex position, given their convex hull. It can be used to derive linear-time algorithms for various problems such as: (1) updating a Voronoi diagram of points after deletion of one site; (2) computing the order-(k+1) subdivision within an order-k Voronoi region of points; (3) computing the farthest Voronoi diagram of point-sites given their convex hull. A much simpler randomized approach for the same problems has been introduced by Chew [5]. The medial axis of a simple polygon is also well known to admit a linear-time construction [6] and uses the framework of [1].

Abstract Voronoi diagrams were introduced by Klein [9] as a unifying framework to various concrete Voronoi instances. Instead of sites and distance measures, they are defined in terms of bisecting curves that satisfy some simple combinatorial properties so that the resulting bisector system is *admissible* [9, 2, 4]. In the abstract setting, Klein and Lingas [10] adapted the linear-time framework, to compute a *Hamiltonian abstract Voronoi diagram* in linear time, given the order of Voronoi regions along an unbounded simple curve (of constant complexity), which visits each region *exactly once.*

A deterministic linear-time approach for problems (1)-(3) for generalized sites other than points and for abstract Voronoi diagrams has been a long-standing open problem. A major difficulty is that the diagrams involved in problems (1)-(3) in this case contain disconnected Voronoi regions. Recently, we derived *expected* linear-time algorithms for problems (1) and (3) in the framework of abstract Voronoi diagrams [7]. Our approach is based on a relaxed version of a Voronoi construct, called the *Voronoi-like diagram*, which provides

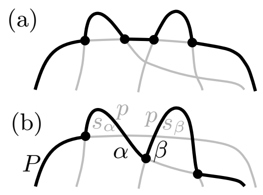


Figure 1 [7] (a) The envelope and (b) a p-monotone path P in an arrangement of bisectors.

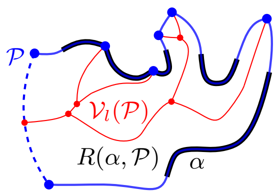


Figure 2 [7] The Voronoi-like diagram $\mathcal{V}_l(\mathcal{P})$ (red) of a boundary curve $\mathcal{P}$ (blue) for $\mathcal{S}'$ (black).

simplified intermediate structures. Related is also an expected linear-time algorithm for the farthest-segment Voronoi diagram [8].

Let $\mathcal{V}(S)$ denote the abstract Voronoi diagram of a set of abstract sites S that define an admissible bisector system $\mathcal{J}$. Let $\mathrm{VR}(s, S)$ denote the Voronoi region of site $s \in S$.

To define a Voronoi-like diagram we first define a *boundary curve* $\mathcal{P}$ related to $\mathrm{VR}(s, S)$. The definition is based on the notion of a *p-monotone path*, which is a path in the arrangement of bisectors $\mathcal{J}_p$ that involve site p, such that any two consecutive edges α, β along this path are pieces of bisectors $J(p, s_\alpha)$ and $J(p, s_\beta)$, respectively, corresponding to Voronoi edges of $\partial \mathrm{VR}(p, \{p, s_\alpha, s_\beta\})$. In contrast, the *p-envelope* is $\partial \mathrm{VR}(p, S)$. Refer to Figure 1.

Let $\mathcal{S} = \partial \mathrm{VR}(s, S)$ be the sequence of Voronoi edges bounding the region $\mathrm{VR}(s, S)$ and let $\mathcal{S}' \subseteq \mathcal{S}$ be a subset of these edges. For simplicity we consider a big closed Jordan curve Γ that contains all intersections of $\mathcal{J}$ and that intersects all bisectors exactly twice. A *boundary curve* $\mathcal{P}$ for $\mathcal{S}'$ is a closed s-monotone path in the arrangement of $\mathcal{J}_s \cup \Gamma$ that contains all Voronoi edges in $\mathcal{S}'$. The boundary curve $\mathcal{P}$ encloses a *domain* $D_\mathcal{P}$ and consists of *boundary arcs*, which represent the related sites (solid arcs in Figure 2), and of Γ-*arcs*, which represent infinity (dashed arc). The original sites in S may appear multiple times along $\mathcal{P}$.

The *Voronoi-like diagram* of a boundary curve $\mathcal{P}$, $\mathcal{V}_l(\mathcal{P})$, is a subdivision of $D_\mathcal{P}$ into regions such that each *boundary arc* α of $\mathcal{P}$ ($\alpha \subseteq J(s, s_\alpha)$) has exactly one region $R(\alpha, \mathcal{P})$ whose boundary is an s_α-*monotone path* in the arrangement of $\mathcal{J}_{s_\alpha} \cup \Gamma$ (instead of an envelope as in a real Voronoi diagram). The Voronoi-like diagram of $\mathcal{S}$, $\mathcal{V}_l(\mathcal{S})$, equals the real Voronoi diagram $\mathcal{V}(S \setminus \{s\}) \cap \mathrm{VR}(s, S)$, which is the diagram computed to solve problem (1) [7].

In this paper we extend the Voronoi-like framework of [7] in various directions and give an expected linear-time algorithm to solve problem (2) in abstract Voronoi diagrams. We first establish that the Voronoi-like diagram of a boundary curve is always well defined by proving the following theorem.

▶ **Theorem 1.** *For any boundary curve $\mathcal{P}$, its Voronoi-like diagram $\mathcal{V}_l(\mathcal{P})$ always exists.*

We then extend the applicability of the randomized algorithm to construct $\mathcal{V}_l(\mathcal{S})$, beyond problem (1), to any subset $\mathcal{S}'$ of the edges on $\partial \mathrm{VR}(s, S)$. Given $\mathcal{S}' \subseteq \mathcal{S}$, in expected linear time we can compute $\mathcal{V}_l(\mathcal{P}_o)$, for some boundary curve $\mathcal{P}_o$ of $\mathcal{S}'$, which still reveals $\mathcal{V}_l(\mathcal{S}) \cap D$ in a subdomain $D \subseteq \mathrm{VR}(s, S)$. In particular, $D = \mathrm{VR}(s, S) \setminus \bigcup_{\alpha \in \mathcal{S} \setminus \mathcal{S}'} \overline{R(\alpha)}$.

Let $\mathcal{V}_k(S)$ denote the *order-k Voronoi diagram* of S and $\mathrm{VR}_k(H, S)$ denote the *order-k Voronoi region* of $H \subseteq S, |H| = k$. We use the above extension to derive the following.

▶ **Theorem 2.** *Given a face f of $VR_k(H, S)$, we can compute $\mathcal{V}_{k+1}(S) \cap f$ in expected $O(m)$ time, where m is the complexity of ∂f.*

Finally, we extend the linear randomized approach to an *admissible* domain D enclosing a *tree-like* Voronoi diagram, relaxing upon the restrictions on D of previous literature [10, 3].

In future research, we expect that Voronoi-like diagrams may lead to a deterministic linear-time construction for problems (1)-(3).

References

1 Alok Aggarwal, Leonidas J. Guibas, James B. Saxe, and Peter W. Shor. A linear-time algorithm for computing the Voronoi diagram of a convex polygon. *Discrete & Computational Geometry*, 4:591–604, 1989. doi:10.1007/BF02187749.

2 Cecilia Bohler, Panagiotis Cheilaris, Rolf Klein, Chih-Hung Liu, Evanthia Papadopoulou, and Maksym Zavershynskyi. On the complexity of higher order abstract Voronoi diagrams. *Computational Geometry: Theory and Applications*, 48(8):539–551, 2015. doi:10.1016/j.comgeo.2015.04.008.

3 Cecilia Bohler, Rolf Klein, and Chih-Hung Liu. Forest-like abstract Voronoi diagrams in linear time. In *Proc. 26th Canadian Conference on Computational Geometry (CCCG)*, 2014.

4 Cecilia Bohler, Chih-Hung Liu, Evanthia Papadopoulou, and Maksym Zavershynskyi. A randomized divide and conquer algorithm for higher-order abstract Voronoi diagrams. *Computational Geometry: Theory and Applications*, 59(C):26–38, 2016. doi:10.1016/j.comgeo.2016.08.004.

5 L. Paul Chew. Building Voronoi diagrams for convex polygons in linear expected time. Technical report, Dartmouth College, Hanover, USA, 1990.

6 Francis Chin, Jack Snoeyink, and Cao An Wang. Finding the medial axis of a simple polygon in linear time. *Discrete & Computational Geometry*, 21(3):405–420, 1999.

7 Kolja Junginger and Evanthia Papadopoulou. Deletion in Abstract Voronoi Diagrams in Expected Linear Time. In *34th International Symposium on Computational Geometry (SoCG 2018)*, volume 99 of *LIPIcs*, pages 50:1–50:14, Dagstuhl, Germany, 2018. URL: http://drops.dagstuhl.de/opus/volltexte/2018/8763.

8 Elena Khramtcova and Evanthia Papadopoulou. An expected linear-time algorithm for the farthest-segment Voronoi diagram. arXiv:1411.2816v3 [cs.CG], 2017. Preliminary version in *Proc. 26th Int. Symp. on Algorithms and Computation (ISAAC), LNCS* 9472, 404-414, 2015.

9 Rolf Klein. *Concrete and Abstract Voronoi Diagrams*, volume 400 of *Lecture Notes in Computer Science*. Springer-Verlag, 1989.

10 Rolf Klein and Andrzej Lingas. Hamiltonian abstract Voronoi diagrams in linear time. In *Algorithms and Computation, 5th International Symposium, (ISAAC)*, volume 834 of *Lecture Notes in Computer Science*, pages 11–19, 1994.

On Mergable Coresets for Polytope Distance

Benwei Shi, Aditya Bhaskara, Wai Ming Tai, and Jeff M. Phillips
School of Computing, University of Utah
b.shi@utah.edu, bhaskara@cs.utah.edu, u1008421@utah.edu, jeffp@cs.utah.edu

Introduction. The max-margin linear separator is a classic problem in machine learning [3], defined as follows. Given a point set $P \subset \mathbb{R}^d$ with labels $\{-1, +1\}$ find a hyperplane h that separates the labels, which maximizes the *margin* $\gamma = \min_{p \in P} \|p - \pi_h(p)\|$, where $\pi_h(p)$ projects p onto h. This is equivalent to the two-polytope min-distance problem, and can be reduced to the one-polytope min-distance (polytope distance for short) problem [4]. Further, a $(1-\varepsilon)$-approximation of polytope distance can be used to obtain a $(1-\varepsilon)$-approximation for max-margin separating hyperplane. The former can be solved by finding an ε-coreset—the objective of this paper, defined formally below.

In this paper we ask if these ε-coresets can be merged [1]. That is, given two ε-coresets S_1 and S_2, can they be combined into a single ε'-coreset while not increasing the space (hopefully with $\varepsilon' = \varepsilon$). By creating coresets on batches of points in a streaming setting, if we can iteratively merge these coresets, this easily leads to streaming algorithms. This framework also implies small space and communication complexity in other big data settings.

Concepts and Definitions. We follow the definition of ε-coreset for polytope distance problem used in Gärtner and Jaggi's paper [4]. Formally, we are given a point set $P \in \mathbb{R}^d$, we want approximate $x^* = \arg\min_{v \in \mathrm{conv}(P)} \|v\|$, the point in conv$(P)$ closest to the origin. Define $p|_x := \frac{\langle p,x \rangle}{\|x\|}$ as the signed length of the *projection* of p onto the direction of the vector x. For any $\varepsilon > 0$, $x \in \mathrm{conv}(P)$ is called an ε-*approximation*, iff $(1-\varepsilon)\|x\| \le p|_x$, $\forall p \in P$; see Figure 1. This approximation is stronger than just requiring the distance $\|x\|$ to be close to the optimal value, $(1-\varepsilon)\|x\| \le \|x^*\|$. In particular, if x is an $(1-\varepsilon)$-approximation, it implies that $(1-\varepsilon)\|x\| \le \min_{p \in P} p|_x = \min_{v \in \mathrm{conv}(P)} v|_x \le \|x^*\| \le \|x\|$. A subset $S \subseteq P$ is an ε-*coreset* of P iff conv(S) contains an $(1-\varepsilon)$-approximation to the distance of conv(P).

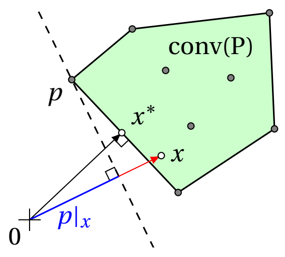
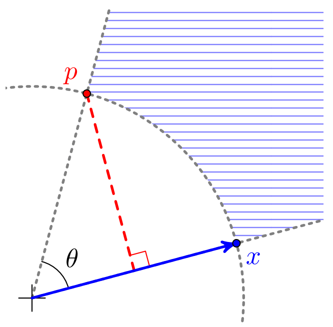


Figure 1 P in gray, and conv(P) in green. Point $x \in \mathrm{conv}(P)$ is an $(1-\varepsilon)$-approximation: the red part has length $\varepsilon\|x\|$.

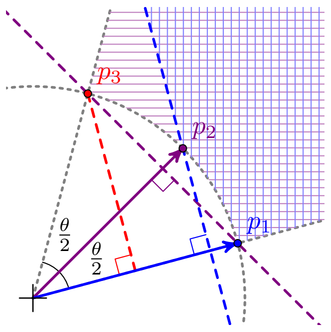
To bound the ε-coreset size, we need some bound on the width of the data P. Gärtner and Jaggi [4] use the *excentricity* of a point set P, defined $E = \frac{\mathsf{diam}(\mathrm{conv}(P))^2}{\|x^*\|^2}$. An ε-coreset with size no more than $2\lceil 2\frac{E}{\varepsilon} \rceil$ always exists [4, 3], and can be found with a simple greedy (Frank-Wolfe) algorithm. In this paper we use the *angular diameter* θ instead of the excentricity E; it is defined as the maximum angle between any two vectors (points) from P. While incomparable to excentricity, this property allows us to provide upper and lower bounds on the mergeability of polytope distance coresets.

Our results. We announce mainly negative results. First we show a constant-size $(1-\cos\theta)$-coreset for polytope distance is simple to find and maintain under merges (Theorem 1). However, increasing the size of the coreset cannot significantly improve the error bound (Theorems 2 and 3); we cannot maintain ε-coresets with arbitrarily small $\varepsilon > 0$ under merges.

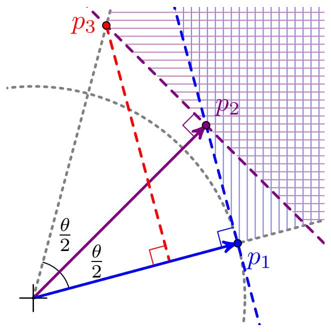
This hardness is not totally unexpected given the known hardness of streaming $(1-\varepsilon)$-approximate minimum enclosing ball [2], which would also imply a streaming coreset for max-margin linear separators. We show that even if we restrict P to have angular diameter at most $\pi/2$ (the margin is relatively large), polytope distance ε-coresets cannot be used to derive a $(1-\varepsilon)$-approximate margin algorithm for max-margin linear separators.



■ **Figure 2** Using shortest point x as coreset



■ **Figure 3** merging two $(1-\cos\frac{\theta}{2})$-coresets



■ **Figure 4** merging two 0-coresets

Maintaining a simple coreset. We first show that the closest point is an ε-coreset with $\varepsilon = 1-\cos\theta$. This is trivial to maintain under merges.

▶ **Theorem 1.** *Consider a point set P with angular diameter $\theta \le \frac{\pi}{2}$. Let $x = \arg\min_{p\in P}\|p\|$, then x is a $(1-\cos\theta)$-coreset of P.*

Proof. This follows almost directly from the definition. The assumption $\theta \le \frac{\pi}{2}$ implies $x \ne 0$ and $\cos\theta \le p|_x/\|p\|, \forall p \in P$. Then $\cos\theta\|x\| \le p|_x$ is immediate since $\|x\| \le \|p\|$; see Figure 2. Thus x is a $(\cos\theta)$-approximation and also is a $(1-\cos\theta)$-coreset of P. ◀

Hardness of merging. We next show this simple $\varepsilon = 1-\cos\theta$ bound cannot be significantly improved. In particular, merging coresets with smaller error may obtain this error (Theorem 2) and even merging 0-error coresets may result in nearly this much error (Theorem 3).

▶ **Theorem 2.** *Consider a point set P of angular diameter $\theta \le \frac{\pi}{2}$. Decompose P into P_1 and P_2. There exists such a setting where (1) S_1 is a $(1-\cos\frac{\theta}{2})$-coreset of P_1, (2) S_2 is a $(1-\cos\frac{\theta}{2})$-coreset of P_2, (3) S is a $(1-\cos\frac{\theta}{2})$-coreset of $S_1 \cup S_2$, but (4) S is no better than a $(1-\cos\theta)$-coreset of P.*

Proof. We prove this existence by an example. Let P include 3 points, p_1, p_2, and p_3. Such that $\|p_1\| = \|p_2\| = \|p_3\|$, $\angle(p_1,p_2) = \angle(p_2,p_3) = \theta/2$, and $\angle(p_1,p_3) = \theta$; see Figure 3. Then for $P_1 = \{p_2,p_3\}$, $S_1 = \{p_2\}$ is a valid $\left(1-\cos\frac{\theta}{2}\right)$-coreset. For $P_2 = \{p_1\}$ then clearly $S_1 = \{p_1\}$ is a valid $\left(1-\cos\frac{\theta}{2}\right)$-coreset. Now let $S = \{p_1\}$, so that S is a valid $\left(1-\cos\frac{\theta}{2}\right)$-coreset of $S_1 \cup S_2 = \{p_2,p_1\}$. However, $\frac{p_3|_{p_1}}{\|p_1\|} = \cos\theta$. Therefore S is not better than $(1-\cos\theta)$-coreset of P. ◀

▶ **Theorem 3.** *Consider a point set P of angular diameter $\theta \le \frac{\pi}{2}$. Decompose P into P_1 and P_2. There exists a setting where (1) S_1 is a 0-coreset of P_1, (2) S_2 is a 0-coreset of P_2, (3) S is a 0-coreset of $S_1 \cup S_2$, but (4) S is no better than a $\left(\frac{1-\cos\theta}{1+\cos\theta}\right)$-coreset of P.*

Proof. The proof is similar with the one of Theorem 2. Let P include 3 points, p_1, p_2, and p_3. Such that $p_2|_{p_1} = \|p_1\|$ and $p_3|_{p_2} = \|p_2\|$, also $\angle(p_1,p_2) = \angle(p_2,p_3) = \theta/2$, and $\angle(p_1,p_3) = \theta$; see Figure 4. Then $S_1 = \{p_2\}$ is a 0-coreset of $P_1 = \{p_2,p_3\}$, and $S_2 = P_2 = \{p_1\}$ is a 0-coreset for P_2. Then $S = \{p_1\}$ is a 0-coreset for $S_1 \cup S_2$. However $\frac{p_3|_{p_1}}{\|p_1\|} = 1 - \frac{1-\cos\theta}{1+\cos\theta}$. Therefore S can be no better than a $\left(\frac{1-\cos\theta}{1+\cos\theta}\right)$-coreset of P. ◀

References

1 Pankaj K. Agarwal, Graham Cormode, Zengfeng Huang, Jeff M. Phillips, Zhewei Wei, and Ke Yi. Mergeable summaries. In *PODS*, 2012.

2 Pankaj K. Agarwal and R Sharathkumar. Streaming Algorithms for Extent Problems in High Dimensions. In *SODA*, 2010.

3 Christopher J C Burges. A Tutorial on Support Vector Machines for Pattern Recognition. *Data mining and knowledge discovery*, 2(2):121–167, 1998.

4 Bernd Gärtner and Martin Jaggi. Coresets for polytope distance. In *SOCG*, 2009.

Art Gallery Problem for Indoor Localization

Haotian Wang[1]**, Jie Gao**
Department of Computer Science, Stony Brook University, NY, USA
haotwang@cs.stonybrook.edu, jgao.cs.stonybrook.edu

Niranjini Rajagopal, Anthony Rowe, Bruno Sinopoli
Department of Electrical and Computer Engineering, Carnegie Mellon University, PA, USA
niranjir@andrew.cmu.edu, agr@ece.cmu.edu, brunos@andrew.cmu.edu

1 Problem Statement

The *Art Gallery Problem* is to find the minimum number of guards such that every point of the domain P can be seen by at least one guard b. In our work, for a point $p \in P$, we are given the distance between p and the guards visible to p (obtained through measurements of wireless signals using Time-Of-Flight (TOF) or Time-Difference-Of-Arrival (TDOA) ranging), with which we hope to find out the location of p.

If a point is only seen by one guard, we cannot decide its location uniquely. With three or more visible guards, a point can be uniquely localized. In between, with exactly two visible guards g_1, g_2, things are tricky. The two measurements provide two candidate locations p_1 and p_2. But we could eliminate the candidate location whose visible guards differ from $\{g_1, g_2\}$. Figure 1 provides a few examples when two guards do provide a unique solution.

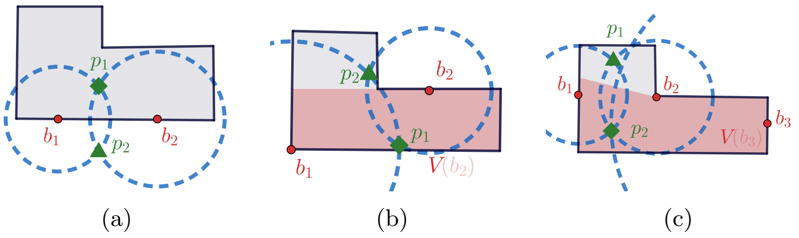


Figure 1 Examples of unique localization by two guards: (a) The candidate location p_2 is outside the domain and is not feasible; (b) The candidate location p_2 is not visible to b_2 and is not feasible; (c) p_2 is visible to b_1, b_2, b_3 and therefore is not feasible.

We now formulate the Art Gallery problem for Indoor Localization. The domain is represented by P, a polygon with possibly holes. B is a set of m *candidate guard locations* inside P. The problem (continuous version) is to find a minimum set of guards $D \subseteq B$ such that the entire domain P can be uniquely localized with guards located at D. The problem also has a discrete form, in which n target points in P need to be uniquely localized.

Our problem is different and more complex than the Art Gallery problem, as we have a stronger requirement on coverage by collaboration. Our problem is also related yet different from the *k-coverage* problem, which ensures every point of P to be covered by at least k guards. In our problem, the points in P have to be uniquely localized, which can be achieved by 2 guards or 3 guards, depending on the geometry.

2 Contribution

We provide two approximation algorithms to solve the art gallery problem for indoor localization.

Greedy Algorithm. The standard Art Gallery problem is a special case of the set cover problem, in which the visible regions of the guards are selected to collectively cover P. For

[*] A longer version is available at http://www3.cs.stonybrook.edu/~haotwang/paper/Localization.pdf

set cover type of problem, maximizing the marginal increase of coverage in each step gives an approximation algorithm when the coverage function is a submodular one [3].

▶ **Definition 1.** ***(Submodular)*** *Let N be a finite set and z be a real-valued defined on the set of subsets of N that satisfies $z(S) + z(T) \geq z(S \cup T) + z(S \cap T)$ for all $S, T \subseteq N$. Such a function is called submodular.*

For our problem unfortunately the set of uniquely localizable points given a set of guards is not submodular. Hence, we design a new objective function which is submodular, which improves over previous work [4]. We use $|V_1(D)|$ and $|V_2(D)|$ to represent the area seen by at least one and two guards in D respectively. $|U(D)|$ is the uniquely localized area by D. We are able to prove that the new objective $F(D) = 3|V_1(D)| + 2|V_2(D)| + |U(D)|$ is a submodular function. The proof is highly non-trivial and is in the full version. For each iteration, we choose the guard that increases this function the most, until all points of P are uniquely localizable. With the results in [3], we can derive

▶ **Theorem 2.** *The number of guards selected by the improved greedy algorithm is an $O(\min\{\ln \frac{|P|}{\Delta}, m\})$-approximation of the optimal solution, where $|P|$ is the area of domain, Δ is the minimum increased area for the objective function F with a new guard, and m is the number of candidate guard locations. In the discrete problem setting, the area is replaced by the number of target points and the approximation ratio is $O(\ln n)$.*

Random Sampling Algorithm. We present another approximation algorithm using the random sampling technique, motivated by the ϵ-net based algorithm for geometric set cover [1, 2]. We first define some terms for our setting.

▶ **Definition 3.** *Every guard b_i is given a weight $w(b_i)$. For a set of guards D, its weight is $w(D) = \sum_{b \in D} w(b)$. The weight of a point $p \in P$ is defined by the weight of the guards that can see p, $w(p) = w(V(p))$, where $V(p) = \{b \in D | b$ can see the point $p\}$.*

▶ **Definition 4.** ***(ϵ-oracle):*** *Given a domain P, a candidate guard location set B and the weight function w, a subset $D \subseteq B$ is an ϵ-oracle for (P, B, w) if p is uniquely localized by D, for each $p \in P$ with $w(p) \geq \epsilon \cdot w(B)$.*

When D is an ϵ-oracle, one can uniquely localize all the locations with weight higher than $\epsilon \cdot w(B)$. Such an oracle can be obtained with probability $1 - \delta$ through random sampling – specifically, if we select a subset of $\max(\frac{2}{\epsilon} \log_2 \frac{1}{2\delta}, \frac{4d+16}{\epsilon} \log_2 \frac{4d+16}{\epsilon})$ guards, where each guard is selected with probability proportional to its weight. Here d is bounded by the VC-dimension of the art gallery problem, which is known to be a constant [2].

In our algorithm, we gradually increment k from 1. For each k, we initially set all weights of the guards to be 1. Perform the (weighted) random sampling procedure above to find k random guards. Then we check if the obtained guard set can uniquely localize all points in P. If yes, we stop and output the guard set. If not, we double the weights of the guards that see points that are not yet uniquely localizable and continue. This iterative procedure stops after $\frac{2k}{\delta} \log_2 \frac{m}{k}$ iterations. Then we increment k and start over.

▶ **Theorem 5.** *If the optimal solution is k^*, then the above algorithm produces $O(k^* \log k^*)$ guards.*

In simulation, we observe that our guard placement algorithm performs better than manual placement (obtained through crowdsourcing experiments). The proposed algorithm also places 5% fewer guards in real-world domain and 12% fewer guards on random polygon compared to prior work based on heuristics [4]. In addition, we have enhanced the random sampling algorithms by introducing a heuristic, based on Geometric Dilution of Precision to improve the accuracy performance.

References

1. Alon Efrat, Sariel Har-Peled, and Joseph SB Mitchell. Approximation algorithms for two optimal location problems in sensor networks. In *2nd International Conference on Broadband Networks, 2005.*, pages 714–723. IEEE, 2005.
2. Giordano Fusco and Himanshu Gupta. ε-net approach to sensor k-coverage. In *International Conference on Wireless Algorithms, Systems, and Applications*, pages 104–114. Springer, 2009.
3. George L Nemhauser, Laurence A Wolsey, and Marshall L Fisher. An analysis of approximations for maximizing submodular set functions—i. *Mathematical programming*, 14(1):265–294, 1978.
4. Niranjini Rajagopal, Sindhura Chayapathy, Bruno Sinopoli, and Anthony Rowe. Beacon placement for range-based indoor localization. In *Indoor Positioning and Indoor Navigation (IPIN), 2016 International Conference on*, pages 1–8. IEEE, 2016.

Trajectory Visibility in a Simple Polygon

Patrick Eades
University of Sydney
patrick.eades@sydney.edu.au

Ivor van der Hoog
Utrecht University
i.d.vanderhoog@uu.nl

Maarten Löffler
Utrecht University
m.loffler@uu.nl

Frank Staals
Utrecht University
f.staals@uu.nl

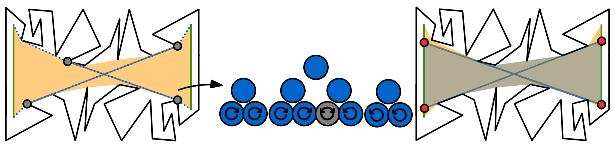


Figure 1 (left) $H(e_1, e_2)$ in orange with the shortest paths between endpoints in dotted blue. The bitangents are underlined in grey. (middle) We obtain each dotted path as a collection of binary search trees, where only the bitangent starts in clockwise and ends in counterclockwise rotation. (right) Using the bitangents, we identify the endpoints of the respective subsegments and obtain $L(e_1, e_2)$ in grey.

1 Introduction

Visibility is one of the most-studied topics in computational geometry. It has many applications, also in adjacent fields such as computer graphics, geographic information science (GIS), and robotics. The visibility-blocking environment is typically modelled as a polygon. Often one is interested in preprocessing a polygon to retrieve visibility information during query time. A natural question to ask (with numerous applications) is the following: suppose you have two entities following different trajectories in a polygonal domain that blocks visibility. Can, at any time, the two entities see one another? Despite the amount of research on both trajectories and visibility almost no previous work in this direction exists.

Problem Statement. Given a simple polygon P with n vertices, can we build a data structure that can answer queries of the type: for any two trajectories T_1, T_2 with τ vertices within P that represent the motion of two entities $q(t)$ and $r(t)$ for $t \in [0, 1]$, is there a time t^* at which q and r can see each other? We propose a near-linear size data structure that can solve the problem in sub-linear time when T_1 and T_2 are line segments (of different length) denoted by e_1 and e_2. The original problem can therefore be solved in $o(\tau n)$ time.

2 Our approach

Guibas and Hershberger [3] study shortest paths in a simple polygon P. They define the *hourglass* $H(e_1, e_2)$ as the union of all shortest paths between points on two line segments e_1 and e_2. The hourglass $H(e_1, e_2)$ is a polygon whose boundary consists of two semi-convex chains and e_1 and e_2 itself. They devise a linear-size data structure $\mathcal{D}$ that can return $H(e_1, e_2)$ in an *implicit representation* (as a collection of at most $\log n$ trees where each node is a vertex of the bounding path). We define the *visibility glass* $L(e_1, e_2)$ as the restricted hourglass in which all paths are segments. Chazelle and Guibas [2] show that $L(e_1, e_2)$ is the hourglass of two segments $e_1' \subset e_1$ and $e_2' \subset e_2$. Using $\mathcal{D}$, we obtain $L(e_1, e_2)$ in logarithmic time: e_1' and e_2' end in the extension of the bitangents of $H(e_1, e_2)$ (Figure 1).

The visibility glass $L(e_1, e_2)$ represents all non-obstructed line segments between q and r. Thus, to test if q and r see one another we can check if there is a time t^* such that the segment $\overline{q(t^*)r(t^*)}$ is contained within $L(e_1, e_2)$. We define the dual $\Lambda(e_1, e_2)$ of $L(e_1, e_2)$ as the dual of the lines in $L(e_1, e_2)$, which forms a convex polygon of linear complexity [2]. Similarly, for any time t we can dualize the line through $q(t)$ and $r(t)$ to a point. This continuous dualization traces a hyperbolic curve $\gamma(t)$ with eight degrees of freedom which we denote by $\vec{a} = (a_1, a_2, \ldots, a_8)$. There is a time t^* where $q(t^*)$ can see $r(t^*)$ if and only if $\gamma(t)$ intersects an edge of or is contained in $\Lambda(e_1, e_2)$. At this point, we employ the linearization technique from Agarwal *et al.* [1]. Suppose you have n objects, each parametrised by a vector $\vec{x}$ and a query parametrised by a vector $\vec{a}$. Suppose that for any combination of $\vec{x}$ and $\vec{a}$, you can express the intersection between the object and the query as a predicate function $F(\vec{x}, \vec{a}) \leq C$ with $F(\vec{x}, \vec{a}) = \sum_{i=0}^{k} f_i(\vec{x}) g_i(\vec{a})$ where f_i and g_i are polynomial functions. Then the intersection query can be transformed into consecutive halfspace emptyness queries in $\mathbb{R}^k$. Our objects are the edges of $L(e_1, e_2)$ parametrized by the coordinates of their endpoints $\vec{x} = (x_1, x_2, x_3, x_4)$ and our query is the curve $\gamma(t)$ parametrized by $\vec{a}$. We give a linearization which yields four halfspace emptyness queries in $\mathbb{R}^k$ with $k \leq 16000$. These four queries can be solved with multi-level partition trees which use near-linear space and construction time and have $O(n^{1-\frac{1}{16000}+\varepsilon})$ query time (where ε is an arbitrarily small positive constant).

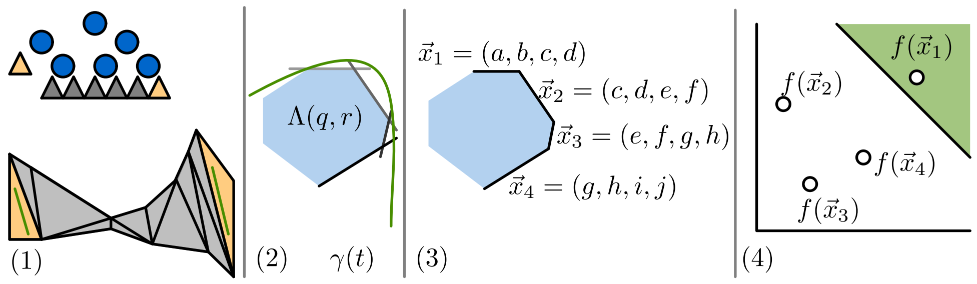

(1)
$\Lambda(q, r)$
(2)
$\gamma(t)$
$\vec{x}_1 = (a, b, c, d)$
$\vec{x}_2 = (c, d, e, f)$
$\vec{x}_3 = (e, f, g, h)$
$\vec{x}_4 = (g, h, i, j)$
(3)
$f(\vec{x}_1)$
$f(\vec{x}_2)$
$f(\vec{x}_4)$
$f(\vec{x}_3)$
(4)

Figure 2 (1) The base level of our data structure is a hierarchical triangulation. (2) Given q and r, we compute the dualized visibility glass and the degree-2 query curve γ. (3) We store the parameters of each edge. (4) Each parameter vector gets mapped to a point in $\mathbb{R}^4$ and the query curve segment gets mapped to a 4-dimensional halfspace which is empty only if γ intersects no edge from the dualized visibility glass.

Our final data structure (Figure 2) consists of two levels. The first level is a slight variation of the two-point shortest-path query data structure of Guibas and Hershberger [3]. The data structure essentially stores a collection of hourglasses explicitly (unlike in the original data structure). For each pre-stored hourglass, its boundary vertices are in leaves of a binary search tree and internal nodes of these trees correspond to subchains. Each internal node v corresponds to a subchain C_v and stores an associated data structure Δ_v. We dualize the supporting-lines of the edges in C_v to points. This essentially dualizes C_v into another polygonal chain in the dual. The associated structure Δ_v stores not only the edges of this dual chain, but also for each edge a specific point in $\mathbb{R}^k$. This allows Δ_v to answer the intersection query using halfspace emptyness queries.

When we get a query consisting of the line-segments e_1, e_2 representing the trajectories of q and r, we have to decide if there exists a time t^* at which q and r can see each other. The main idea is to query our data structure for the visibility-glass $L(e_1, e_2)$ and we obtain $L(e_1, e_2)$ as a collection of $\mathcal{O}(\log^2 n)$ nodes. These nodes together store the dual visibility glass $\Lambda(e_1, e_2)$. Given e_1, e_2, we can compute the dual query hyperbola $\gamma(t)$ and its degrees of freedom $\vec{a}$ in constant time. We then for each node v, query its associate data structure Δ_v to detect an intersection between $\gamma(t)$ and a part of $\Lambda(e_1, e_2)$. It follows from our formulation of the predicate function (which specifies if there is an intersection between $\gamma(t)$ and $\Lambda(e_1, e_2)$) that the total query time is $O(n^{1-\frac{1}{16000}+\varepsilon})$.

References

1 Agarwal, Matousek, and Sharir. On range searching with semialgebraic sets. *JoC*, 2013.

2 Chazelle and Guibas. Visibility and intersection problems in plane geometry. *DCG*, 1989.

3 Guibas and Hershberger. Optimal shortest path queries in a simple polygon. *JoCSS*, 1989.

Computing feasible trajectories for an articulated probe in three dimensions

Ovidiu Daescu
University of Texas at Dallas, Richardson TX 75080, USA
ovidiu.daescu@utdallas.edu

Ka Yaw Teo
University of Texas at Dallas, Richardson TX 75080, USA
ka.teo@utdallas.edu

Abstract

Problem statement. An articulated probe is modeled as two line segments ab and bc connected at point b (Figure 1(A)). Line segment ab can be infinitely long, while bc is of a given length r. The input consists of a set P of n interior-disjoint triangular obstacles and a target point t in the free space, all enclosed by a large sphere S centered at t. Initially, the probe is located outside S and assumes an unarticulated configuration, in which line segments ab and bc are collinear and $b \in ac$. The goal is to find a feasible (obstacle avoiding) probe trajectory to reach t, with the condition that the probe is constrained by the following sequence of moves – a straight line insertion of the unarticulated probe into S, possibly followed by a rotation of bc at b by at most 90°, so that c coincides with t.

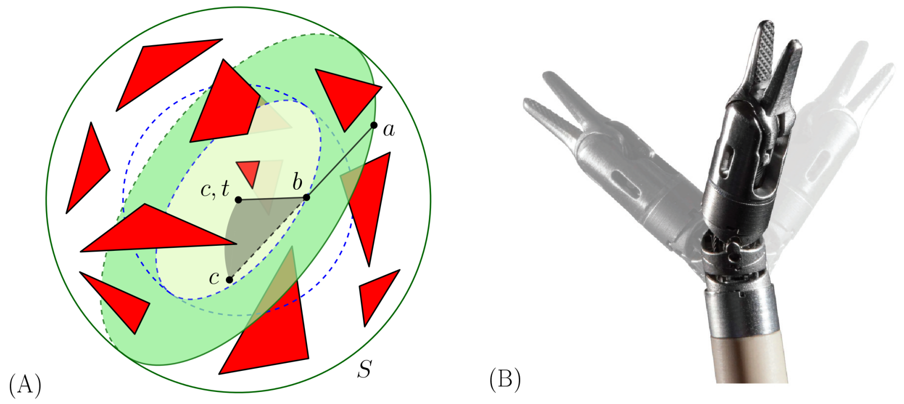


Figure 1 (A) After inserting line segment abc into sphere S, in order to reach target point t, line segment bc may be rotated from its intermediate position (dashed line) to its final position (solid line). (B) Example of an articulated probe (da Vinci EndoWrist by Intuitive Surgical).

Prior work. The motion of a linkage – that is, a sequence of fixed-length edges connected consecutively through joints – has been formerly studied from various perspectives, ranging from basic properties and questions (e.g., reachability, reconfiguration, and locked decision) with strong geometric and topological aspects [2, 6] to application-driven problems related to linkage design and motion planning [1, 7]. In contrast to those previous studies on polygonal linkages, which are generally allowed to rotate unrestrictedly at their joints while moving from a start to a final configuration, our work is concerned with finding a collision-free path of motion for a two-bar linkage constrained to an ordered sequence of moves – namely, a straight insertion of the linkage followed by a rotation at its joint. Furthermore, one of the links is considered to be unbounded in length.

Daescu, Fox, and Teo [4] originally proposed the aforementioned trajectory planning problem in two dimensions, and they presented an $O(n^2 \log n)$-time, $O(n \log n)$-space algorithm for finding a feasible trajectory amidst n line segment obstacles. The algorithm was based on computing extremal trajectories that are tangent to one or two obstacle vertices. This algorithmic approach was later extended to finding a feasible trajectory of a given clearance δ from the obstacles, for any $\delta > 0$, in $O(n^2 \log n)$ time using $O(n^2)$ space [3]. In addition, Daescu and Teo [5] showed that the feasible solution space for the two-dimensional trajectory planning problem can be characterized by a simple-curve arrangement of complexity $O(k)$, and the arrangement can be constructed in $O(n \log n + k)$ time using $O(n + k)$ space, where $k = O(n^2)$ is the number of vertices in the arrangement.

Motivation. Besides its apparent relevance in robotics, the outlined problem arises particularly from planning for minimally invasive surgeries. In fact, surgical instruments that can be modeled by our simple articulated probe are already in clinical use (Figure 1(B)), given their enhanced capability in reaching targets while circumventing surrounding critical structures. In our problem setting, a human body cavity can be viewed as (a subset of) workspace S, and any critical organ/tissue can be represented by using a triangle mesh. Despite its importance and relevance, the problem has never been investigated in three dimensions from a theoretical viewpoint, and only a handful of results in two dimensions have been reported [3, 4, 5]. Due to practicality, it may seem inevitable to use heuristics and approximation (through voxelization) in real-life applications; nevertheless, analyzing the problem using an exact solution approach allows us to fully explore its rich combinatorial and geometric properties, which have often proven useful in seeking algorithmic improvement.

Our results. We prove that if there exists a feasible probe trajectory, then some *extremal* feasible trajectories must be present. An extremal trajectory is characterized by its tangencies to a combination of obstacle edges and/or vertices. Through careful case analysis, we show that these extremal trajectories can be represented by $O(n^4)$ combinatorial events. We present a solution approach that enumerates and verifies these combinatorial events for feasibility, and our main result is summarized in the following theorem.

▶ **Theorem 1.** *One can determine if a feasible probe trajectory exists, and if so, report (at least) one such trajectory by computing and checking $O(n^4)$ extremal trajectories for feasibility in $O(n^{4+\epsilon})$ time using $O(n^{4+\epsilon})$ space, for any $\epsilon > 0$.*

As an alternative, an $O(n^5)$-time algorithm with linear space usage is achievable by performing a simple $O(n)$-check on each of the $O(n^4)$ events. The proposed enumeration algorithm is highly parallel, considering that each combinatorial event can be generated and verified for feasibility independent of the others. In the process of deriving our solution, we address a special instance of the circular sector emptiness query problem in three dimensions, which we consider to be of independent interest. Specifically, we obtain the following result.

▶ **Theorem 2.** *For any $\epsilon > 0$, a set P of n triangles in $\mathbb{R}^3$ can be preprocessed in $O(n^{3+\epsilon})$ time into a data structure of size $O(n^{3+\epsilon})$ so that, for a query circular sector σ with a fixed radius r and an endpoint of its arc located at fixed point t, one can determine if σ intersects P in $O(\log^2 n)$ time.*

The result above also implies a new data structure for the corresponding emptiness query problem in two dimensions.

▶ **Theorem 3.** *A set P of n line segments in $\mathbb{R}^2$ can be preprocessed in $O(n \log n)$ time into a data structure of size $O(n\alpha(n))$, where $\alpha(n)$ is the inverse Ackermann function, so that, for a query circular sector σ with a fixed radius r and a fixed arc endpoint t, one can determine if σ intersects P in $O(\log n)$ time.*

It is worth mentioning that our new $\mathbb{R}^2$ query data structure simplifies the two-part approach formerly proposed in [4] while maintaining the same time and space complexity.

2012 ACM Subject Classification Theory of computation → Computational geometry; Computing methodologies → Motion path planning; Computing methodologies → Robotic planning

Keywords and phrases Articulated probe trajectory, motion planning, circular sector emptiness

Acknowledgements We would like to thank Pankaj K. Agarwal for helpful discussions and comments.

References

1 Howie M Choset, Seth Hutchinson, Kevin M Lynch, George Kantor, Wolfram Burgard, Lydia E Kavraki, and Sebastian Thrun. *Principles of robot motion: theory, algorithms, and implementation.* MIT Press, 2005.

2 Robert Connelly and Erik D Demaine. Geometry and topology of polygonal linkages. *Handbook of Discrete and Computational Geometry*, pages 233–256, 2017.

3 Ovidiu Daescu, Kyle Fox, and Ka Yaw Teo. Computing trajectory with clearance for an articulated probe. In *28th Annual Fall Workshop on Computational Geometry*, 2018.

4 Ovidiu Daescu, Kyle Fox, and Ka Yaw Teo. Trajectory planning for an articulated probe. In *30th Annual Canadian Conference on Computational Geometry*, pages 296–303, 2018.

5 Ovidiu Daescu and Ka Yaw Teo. Characterization and computation of the feasible space of an articulated probe. In *34th Annual European Workshop on Computational Geometry*, 2019.

6 John Hopcroft, Deborah Joseph, and Sue Whitesides. Movement problems for 2-dimensional linkages. *SIAM Journal on Computing*, 13(3):610–629, 1984.

7 Steven M LaValle. *Planning algorithms.* Cambridge University Press, 2006.

New Applications of Nearest-Neighbor Chains

Nil Mamano[1]
Department of Computer Science, University of California Irvine, US
nmamano@uci.edu

Alon Efrat
Departnment of Computer Science, University of Arizona, US
alon@cs.arizona.edu

David Eppstein
Department of Computer Science, University of California Irvine, US
eppstein@uci.edu

Daniel Frishberg
Department of Computer Science, University of California Irvine, US
dfrishbe@uci.edu

Michael T. Goodrich
Department of Computer Science, University of California Irvine, US
goodrich@uci.edu

Stephen Kobourov
Departnment of Computer Science, University of Arizona, US
kobourov@cs.arizona.edu

Pedro Matias
Department of Computer Science, University of California Irvine, US
pmatias@uci.edu

Valentin Polishchuk
Communications and Transport Systems, ITN, Linköping University, Sweden
valentin.polishchuk@liu.se

Abstract

We propose a general technique for speeding up closest-pair-based algorithms. In particular, this allows us to speed up the multi-fragment algorithm for Euclidean TSP from $O(n^2)$ to $O(n \log n)$ in any fixed dimension.

2012 ACM Subject Classification Theory of computation → Design and analysis of algorithms; Theory of computation → Computational geometry

Keywords and phrases Nearest-neighbor chain, multi-fragment algorithm, Euclidean TSP

Related Version A full version of the paper is available on `https://arxiv.org/abs/1902.06875`.

Funding *David Eppstein*: Supported in part by NSF grants CCF-1618301 and CCF-1616248.
Michael T. Goodrich: Supported in part by NSF grant 1815073.
Stephen Kobourov: Supported in part by NSF grants CCF-1740858, CCF-1712119, DMS-1839274, and DMS-1839307.

1 New Applications of Nearest-Neighbor Chains

Consider *closest-pair-based* algorithms, which work by repeatedly finding the closest pair among a set of geometric objects. Some prominent examples include:

[1] Corresponding author.

- Greedy Euclidean Matching: given a set of $2n$ points, repeatedly match and remove the closest pair.
- Agglomerative hierarchical clustering: given a set of points, initialize a singleton cluster for each point. Then, repeatedly merge the two closest clusters into a super-cluster until there is a single cluster left. The distance between clusters is defined as the minimum (or maximum) distance between points inside each cluster.
- Multi-fragment Euclidean TSP: Given a set of points, initialize a single-node path for each point. Then, repeatedly connect the two closest paths into a bigger path, where the distance between paths is defined as the minimum distance between their endpoints. Once there is a single path left, connect its endpoints to form a closed cycle.
- Geometric stable matching: given a set of n red points and a set of n blue points, the goal is to find a stable matching between the sets, where each point ranks the points in the other set by proximity (with closer being preferred). A stable matching can be found by repeatedly matching and removing the closest pair of different colors.

A property of closest-pair-based algorithms. In a set of objects in space (points, clusters, ...), we say two objects are *mutual nearest neighbors* (MNN) if they are the nearest neighbor of each other. Note that the closest pair in the set is a pair of MNN, but the converse is not necessarily true. Now, consider a variation of a closest-pair-based algorithm where, instead of finding and processing the closest pair at each step, it finds MNN and processes them in the same way. In the four mentioned algorithms, this modification *does not affect the result.* This is not the case for every closest-pair-based algorithm, but, as the examples above show, it is a common feature. This is interesting because MNN are defined on local information about the two objects, while the closest pair is a global property of the entire set.

An algorithmic technique to exploit it. In many settings, we can find MNN faster than closest pairs using a technique called *nearest-neighbor chain* (NNC). We illustrate it for Euclidean matching. We maintain a stack (called *chain*) of points. The first point is arbitrary. We repeatedly extend the chain with the nearest neighbor of the current point at the top of the chain. Note that the distance between points in the chain keeps decreasing, so, assuming there are no ties, no repeated points occur, and the chain inevitably reaches a pair of MNN. Then, the MNN are matched and removed from the chain. Crucially, after a match happens, the rest of the chain is not discarded. Every point in the chain still points to its nearest neighbor, so the chain is still valid. The process continues from the new top of the chain.

The algorithm is efficient because each point is added to the chain only once, since it stays there until it is matched with another point. This bounds the number of iterations to be linear on the input size. The runtime is $O(nT(n))$, where $T(n)$ is the time per operation of a dynamic nearest-neighbor structure.

Related work and contributions. The NNC algorithm was invented to speed up agglomerative hierarchical clustering in the 70's [3]. Very recently, it found its first use outside of clustering in geometric stable matching [1]. In the full preprint of this abstract [2], we extend this technique to a number of problems: we speed up the multi-fragment algorithm for Euclidean TSP from $O(n^2)$ to $O(n \log n)$ in any fixed dimension. This requires additional techniques to use approximate near neighbors rather than exact nearest neighbors in the NNC algorithm. We also use NNC to speed up algorithms for constructing straight skeletons, a new stable matching problem, and a geometric coverage problem. The hope is to find new problems where these ideas apply at SoCG'19.

References

1 David Eppstein, Michael T. Goodrich, and Nil Mamano. Algorithms for stable matching and clustering in a grid. In *International Workshop on Combinatorial Image Analysis*, pages 117–131. Springer, 2017.

2 Nil Mamano, Alon Efrat, David Eppstein, Daniel Frishberg, Michael Goodrich, Stephen Kobourov, Pedro Matias, and Valentin Polishchuk. Euclidean tsp, motorcycle graphs, and other new applications of nearest-neighbor chains, 2019. `arXiv:1902.06875`.

3 Fionn Murtagh. A survey of recent advances in hierarchical clustering algorithms. *The Computer Journal*, 26(4):354–359, 1983.

Active Learning a Convex Body in Low Dimensions

Sariel Har-Peled, Mitchell Jones, and Saladi Rahul
Dept. of Computer Science, University of Illinois at Urbana-Champaign, USA
{sariel, mfjones2}@illinois.edu, saladi.rahul@gmail.com

Related version. The full version of this paper is available on arXiv [4].

1. Introduction

Active learning is a subfield of machine learning, in which at any time, the learning algorithm is able to query an oracle for the label of a particular data point. A popular model in active learning is the membership query synthesis model [1]. Here, the learner wants to minimize the number of oracle queries, as such queries are expensive—they usually correspond to either consulting with a specialist, or performing an expensive computation. In this setting, the learning algorithm is allowed to query the oracle for the label of any data point in the instance space. See [7] for a more in-depth survey on active learning.

The problem. In this paper we consider a variation on the active learning problem in the membership query synthesis model. The goal of the learner is to learn a convex body C in $\mathbb{R}^d$ with access to a *separation* oracle. For a query $q \in \mathbb{R}^d$, the oracle either reports that $q \in C$, or returns a hyperplane separating q and C (as a proof that $q \notin C$). Note that if the query is outside the body, the oracle answer is significantly more informative than just the label of the point. The learner is provided with a set P of n unlabelled points in $\mathbb{R}^d$ and access to a separation oracle for an unknown convex body C in $\mathbb{R}^d$. The task is to label each point as either inside or outside C while minimizing the number of separation oracle queries.

Hard and easy instances. We show that in the worst case, an algorithm may have to query the oracle for all input points (see Lemma 1). As such, the purpose here is to develop algorithms that are *instance sensitive*—if the given instance is easy, they work well. If the given instance is hard, they might deteriorate to the naive algorithm that queries all points.

Additional motivation & some previous work.

(A) *Separation oracles.* The use of separation oracles is a common tool in optimization (e.g., solving exponentially large linear programs) and operations research. It is natural to ask what other problems can be solved efficiently with access to this specific type of oracle.

(B) *Other types of oracles.* Various models of computation utilizing oracles has been previously studied within the community. Examples of other models include nearest-neighbor oracles (i.e., black-box access to nearest neighbor queries over a point set P) [5], and proximity probes (which given a convex polygon C and a query q, returns the distance from q to C) [6]. Furthermore, other types of oracles (rather than just membership oracles) have also been studied within the learning community, see [1].

(C) *Active learning.* As discussed, the problem at hand can be interpreted as active learning a convex body in relation to a set of points P that need to be classified (as either inside or outside the body), where the queries are via a separation oracle. We are unaware of any work directly on this problem in the theory community, while there is some work in the machine learning community that studies related active learning classification problems [2, 3, 7]. However we emphasize that our model differs from most of the other membership based query models in the machine learning community. Specifically, each oracle query provides more information than just the label of a data point.

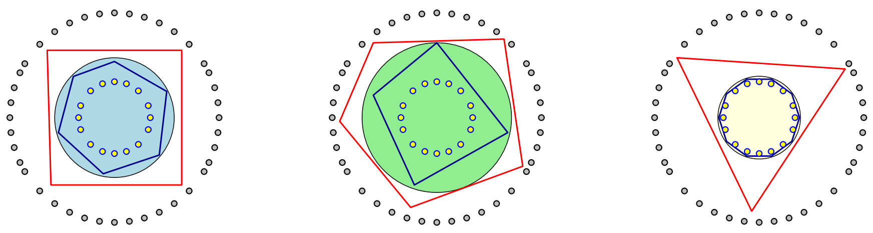


Figure 1.1 The separation price, for the same point set, is different depending on how "tight" the body is in relation to the inner and outer point set.

Note that if some error in classification is allowed, then at first it appears that PAC learning could work. However for learning arbitrary convex ranges, the PAC model fails since the VC dimension of such ranges is infinite.

2. Our results

A lowerbound. Given a set P of points in the plane, and a convex body C, the *outer fence* of P is a closed convex polygon F_{out} with minimum number of vertices, such that $C \subseteq F_{\text{out}}$ and $C \cap P = F_{\text{out}} \cap P$. Similarly, the *inner fence* is a closed convex polygon F_{in} with minimum number of vertices, such that $F_{\text{in}} \subseteq C$ and $C \cap P = F_{\text{in}} \cap P$. Intuitively, the outer fence separates $P \setminus C$ from the boundary of C, while the inner fence separates $P \cap C$ from the boundary of C. The *separation price* of P and C is $\circledcirc(P, C) = |F_{\text{in}}| + |F_{\text{out}}|$, where $|F|$ denotes the number of vertices of a polygon F. We prove the following result.

▶ **Lemma 1.** *Given a point set P and a convex body C in the plane, any algorithm that classifies the points of P in relation to C, must perform at least $\circledcirc(P, C)$ oracle queries.*

See Figure 1.1 for example instances, where the minimum number of queries required changes depending on the position and size of the convex body in relation to the point set.

Algorithms. Each of the algorithms focuses on maintaining a current approximation of the unknown convex body.

(A) We develop a greedy algorithm, for points in the plane, which solves the problem using $O(\hexagon_P \log n)$ oracle queries, where $\hexagon_P$ is the largest subset of points of P in convex position. Note that $\hexagon_P$ can be larger than $\circledcirc(P, C)$, thus this result can be far from optimal.

The algorithm works by maintaining an approximation B of the body C, with $B \subseteq C$. In each iteration, select the halfspace tangent to B containing the largest number of unclassified points $U \subseteq P$. The algorithm computes the centerpoint c of U and queries the oracle with c. The intuition is that by using a centerpoint c as the query, we can either improve the approximation B (if $c \in C$) or classify many points (if $c \notin C$).

(B) The above algorithm naturally extends to three dimensions, also using $O(\hexagon_P \log n)$ oracle queries. While the proof idea is similar to that of the algorithm in 2D, we believe the analysis in three dimensions is also technically interesting.

(C) We present an improved algorithm for the 2D case using $O(\circledcirc(P, C) \log^2 n)$ queries.

(D) We consider the extreme scenarios of the problem: Verifying that all points are either inside or outside of C. For each problem we present a $O(\log n)$ approximation algorithm to the optimal strategy.

It is currently open to improve any of the logarithmic factors in the above algorithms. Finally, obtaining any results in dimensions greater than three is also an open problem.

References

1 D. Angluin. Queries and concept learning. *Machine Learning*, 2(4):319–342, 1987. URL: https://doi.org/10.1007/BF00116828, doi:10.1007/BF00116828.

2 D. A. Cohn, L. E. Atlas, and R. E. Ladner. Improving generalization with active learning. *Machine Learning*, 15(2):201–221, 1994. URL: https://doi.org/10.1007/BF00993277, doi:10.1007/BF00993277.

3 Y. Guo and R. Greiner. Optimistic active-learning using mutual information. In *Proc. 20th Int. Joint Conf. on AI* (IJCAI), pages 823–829, 2007. URL: http://ijcai.org/Proceedings/07/Papers/132.pdf.

4 S. Har-Peled, M. Jones, and S. Rahul. Active learning a convex body in low dimensions. *CoRR*, abs/1712.02949, 2019. URL: https://arxiv.org/abs/1903.03693.

5 S. Har-Peled, N. Kumar, D. M. Mount, and B. Raichel. Space exploration via proximity search. *Discrete Comput. Geom.*, 56(2):357–376, 2016. URL: https://doi.org/10.1007/s00454-016-9801-7, doi:10.1007/s00454-016-9801-7.

6 F. Panahi, A. Adler, A. F. van der Stappen, and K. Goldberg. An efficient proximity probing algorithm for metrology. In *Int. Conf. on Automation Science and Engineering, CASE 2013*, pages 342–349, 2013. URL: https://doi.org/10.1109/CoASE.2013.6653995, doi:10.1109/CoASE.2013.6653995.

7 B. Settles. Active learning literature survey. Technical Report #1648, Computer Science, Univ. Wisconsin, Madison, January 2009. URL: https://minds.wisconsin.edu/bitstream/handle/1793/60660/TR1648.pdf?sequence=1&isAllowed=y.

Skeletonisation Algorithms for Unorganised Point Clouds with Theoretical Guarantees

Philip Smith[1]
Department of Computer Science, University of Liverpool, UK
Philip.Smith@liverpool.ac.uk

Vitaliy Kurlin
Department of Computer Science, University of Liverpool, UK
Vitaliy.Kurlin@liverpool.ac.uk

Abstract

Data often comes in the form of an unstructured point cloud, that is a finite set of points equipped with a pairwise distance function. It is the task of data skeletonisation algorithms to appropriately represent such point clouds by a skeleton. This is an important step in interpreting the data by allowing the visualisation of topological and geometric features that are implicitly present. We compare three skeletonisation algorithms: the established Mapper and α-Reeb algorithms, and the more recent HoPeS algorithm, introduced by V. Kurlin [3]. We compare their abilities to capture topological and geometric features implicitly present in both synthetic and real data.

1 The Problem and the Algorithms

The problem skeletonisation algorithms attempt to solve is, when given a noisy point cloud C sampled from a graph G in a metric space, to produce a reconstruction G' of G that is both topologically and geometrically similar to G. Topological similarity requires that the reconstructed graph G' has the same first Betti number (i.e. the number of independent cycles) as G, and can be continuously deformed to the original graph G. Geometric similarity means that G and G' are close to each other with respect to a distance. For example, G' should be in a small offset of G and vice versa.

The three skeletonisation algorithms chosen to be compared were selected because, not only do they share broadly similar input and output, but they also all have theoretical guarantees. For example, HoPeS's Reconstruction Theorem [4, 6] gives conditions for a noisy sample of a graph such that HoPeS provides a reconstructed graph with the correct first Betti number and within a small offset of the sample.

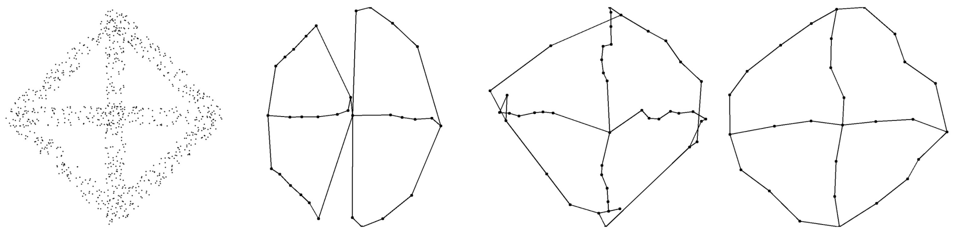


Figure 1 Left: A cloud generated from the wheel four pattern with uniform noise of 0.1. The remaining three figures are the outputs of each algorithm: Middle-left: Mapper; Middle-right: The α-Reeb algorithm; Right: A simplified HoPeS output.

The Mapper algorithm [5] uses a method of partial clustering to convert a point cloud into a network of interlinked clusters. The α-Reeb algorithm [1] discretises the classical

[1] The author thanks the Leverhulme Trust for funding this research via the Leverhulme Research Centre for Functional Materials Design.

Reeb graph and can be applied to discrete clouds at different scales α. Lastly, HoPeS [2, 3] – a Homologically Persistent Skeleton – uses persistent homology to extend the minimum spanning tree relative to a filtration of a point cloud by adding critical edges to form cycles.

2 Comparing the Algorithms

Having implemented the three algorithms, we compared them using both synthetic and real data [6]. The **synthetic dataset** consists of noisy point clouds that were generated from known graphs (which we refer to as patterns) by uniformly sampling points from a pattern (the number of points being proportional to the length of the pattern), and then perturbing each point according to a given type and magnitude of noise. A wide range of types of patterns were used, along with variations in magnitudes of two types of noise – uniform and Gaussian – to produce a large dataset (of more than 50000 clouds), which could be used by new algorithms in order to compare their capabilities too.

When a cloud C produced from a pattern P is run through one of the algorithms, we analyse the output according to four criteria. Namely, does the output have the same first Betti number as P? Are the output and P homeomorphic? What is the output's RMS error – the root-mean-square deviation of the cloud from the output? What is the runtime of the algorithm?

For each type of cloud (same pattern, noise type and noise magnitude), we randomly produced 200 such clouds and asked the above questions for each of the 200 outputs, either taking the mean result or obtaining a success rate – the percentage of the 200 outputs that meet the criteria. A small sample of the results presented in [6] can be seen in Figure 2.

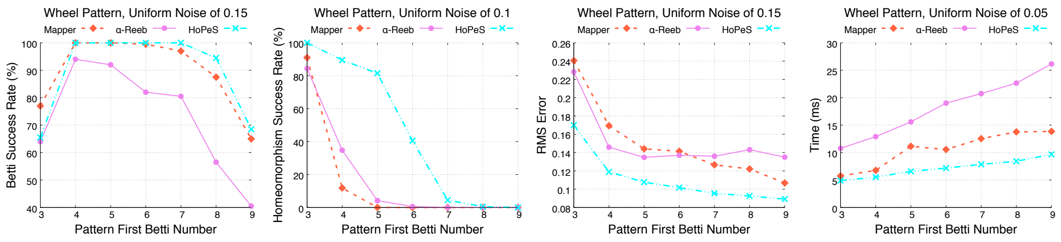


Figure 2 Along the x-axis of these graphs we have the first Betti number of the pattern that produced the cloud, so here we are going from the wheel three to wheel nine pattern. Left: Betti success rate; Middle-left: Homeomorphism success rate; Middle-right: RMS error; Right: Runtime.

3 Conclusion

We conclude from the results that HoPeS generally outperforms the other two algorithms. HoPeS and Mapper are more likely to produce outputs with the correct first Betti number, with the HoPeS output more likely to be homeomorphic to the pattern than the other two. In addition, the HoPeS output is usually geometrically closer to the point cloud, as it has a lower RMS error, and it is also the faster algorithm. A key drawback of Mapper and the α-Reeb algorithm is that they require additional parameters, whereas HoPeS does not. We optimised the choice of parameters for these two algorithms by performing the experiments multiple times with different parameter configurations, only taking the best results.

We also carried out experiments on real data, using images from the BSDS500 dataset. On this data it was again HoPeS that outperformed the other two algorithms. The synthetic dataset and C++ code are available on request from the authors.

References

1 F. Chazal, R. Huang, and J. Sun. Gromov-hausdorff approximation of filament structure using reeb-type graph. *Discrete Computational Geometry*, 53:621–649, 2015.

2 S. Kališnik, V. Kurlin, and D. Lešnik. A higher-dimensional homologically persistent skeleton. *Advances in Applied Mathematics*, 102:113–142, 2019.

3 V. Kurlin. A one-dimensional homologically persistent skeleton of an unstructured point cloud in any metric space. *Computer Graphics Forum*, 34(5):253–262, 2015.

4 V. Kurlin. A fast persistence-based segmentation of noisy 2d clouds with provable guarantees. *Pattern Recognition Letters*, 83:3–12, 2016.

5 G. Singh, F. Mémoli, and G. Carlsson. Topological methods for the analysis of high dimensional data sets and 3d object recognition. *Eurographics Symposium of Point-Based Graphics*, pages 91–100, 2007.

6 P. Smith and V. Kurlin. Skeletonisation algorithms for unstructured point clouds with theoretical guarantees. 2019. `arXiv:1901.03319`.

Jaccard Filtration and stable paths for Mapper

Dustin L. Arendt[1]
Visual Analytics Group, Pacific Northwest National Laboratory, USA

Bala Krishnamoorthy[2]
Department of Mathematics and Statistics, Washington State University, USA

Nathaniel Saul[1,2]
Department of Mathematics and Statistics, Washington State University, USA
nathaniel.saul@wsu.edu

Mapper is a popular new method of exploratory data analysis that leverages ideas of Algebraic Topology to construct a skeletonization of high dimensional data [5]. Given topological spaces X, Y, a function $f : X \to Y$, and a cover U of Y, **Mapper** is defined as the nerve of the refined pullback cover $f^*(U)$. The pullback cover is a cover constructed as the preimage of cover elements U, i.e. $f^{-1}(U_i) = \{x \in X \mid f(x) \in U_i\}$. Refinement of this pullback cover is done by splitting each cover element into new cover elements representing path-connected components or clusters.

When analyzing a Mapper construction, it is common to explore vertex memberships and relationships between vertices [4]. In this context, the size of intersections becomes important and can affect the robustness of an analysis. The standard Mapper construction computes the nerve of a cover which is blind to the size of the intersection, drawing an edge for both large and small overlap of cover elements.

The multi-scale mapper implicitly quantifies the intersection sizes using a tower of covers to build sequence of mappers connected by simplicial mappers [3], but can be difficult to compute and interpret. The multi-nerve mapper finds a stable mapper with respect to the Reeb graph, but is restricted to 1-dimensional functions [1]. To facilitate meaningful interpretation of Mapper, we import ideas from Persistent Homology. We define a new nerve operator that incorporates intersection size, which in turn provides a filtration on any Mapper. Using this filtration, we explore stable features for Mapper analysis, specifically the stability of paths.

Jaccard Nerve

We begin with an extension of the Jaccard distance from an operator on a pair of elements to an operator on a set of elements of a cover. Let $\mu(\cdot)$ be a measure on a set, using cardinality in the common context of discrete data, and $\bigcap$ and $\bigcup$ representing intersection and union of sets respectively.

▶ **Definition 1** (Generalized Jaccard Distance)**.** Given a subsets Ω of X, Define the *generalized Jaccard distance* on $\{U_i\} \subset \Omega$ as $d_J(\{U_i\}) = 1 - {}^{\mu(\bigcap U_i)}/_{\mu(\bigcup U_i)}$.

Using this generalized distance, we extend the definition of the *nerve of a cover* to a weighted nerve that includes information about intersection size. Recall that the nerve of a cover $\mathcal{U}$ is a simplicial complex with an n-simplex defined for each nonempty n-way intersection of elements of $\mathcal{U}$.

[1] Funding: Laboratory Directed Research and Development Program at PNNL
[2] Funding: NSF DBI-1661348 and NSF DMS-1819229

▶ **Definition 2** (Jaccard Nerve). The *Jaccard Nerve* of a cover $\mathcal{U}$, denoted $\mathrm{Nrv}_J(\mathcal{U})$, is defined as the nerve of $\mathcal{U}$ with each simplex assigned their generalized Jaccard distance as weight: $w_\sigma = d_J(\{U_i \mid i \in \sigma\}) \ \ \forall \sigma \in \mathrm{Nrv}(\mathcal{U})$.

The Jaccard Nerve can be thought of as a weighted nerve, but conveniently, the weighting scheme satisfies the conditions of a monotonic filtration, i.e. $K_i \subset K_{i+1}$ for each i in the sequence of simplicial complexes. See Appendix A.1 for a proof.

▶ **Theorem 3.** *The Jaccard Nerve of a cover $\mathcal{U}$ is a filtered simplicial complex.*

The Jaccard Nerve could be applied to a cover of continuous elements, such as intervals. We conjecture that the Čech filtration on a finite set of points (i.e. the nerve of balls with radius r around each point and over a sequence of r) and the Jaccard Nerve constructed from the terminal cover of the Čech filtration are isomorphic, i.e. insertion order of simplices is equivalent and there exists a continuous bijection between insertion times of the Jaccard Nerve and insertion times of the Čech filtration. We prove the case when $n = 1$, i.e. X is drawn from the real line,and provide experimental results for the 1-skeleton equivalence in Appendix A.2.

▶ Conjecture 4 (Čech equivalence). Given a finite data set $X \subset \mathbb{R}^n$ and some radius $R > \mathtt{diam}(X)$ the Čech filtration constructed from X is isomorphic to the the cover filtration on X constructed from the Čech complex of X constructed with radius R.

Jaccard Mapper

We now define a filtration on Mapper using the Jaccard Nerve construction rather than the traditional nerve. This construction allows us to explore persistent features within Mapper. We found in practice that using the persistence diagram to tune an intersection threshold of an over-connected Mapper is considerably easier than tuning cover parameters directly. Additionally, we define stable paths that provide a way of quantifying confidence that edges in a path do exist and are not due to noise in the data or an artifact of Mapper hyperparameter selection.

▶ **Definition 5** (Jaccard Mapper). Given data X, a function $f : X \to Y$, and a cover U of Y, define the *Jaccard Mapper* as the Jaccard Nerve of the refined pullback cover of $f(U)$: $\mathrm{Nrv}_J(f^*(U))$.

▶ **Definition 6** (ρ-Stable Path). Given a Jaccard distance ρ, a path P is defined to be *ρ-stable* if $\max\{d_J(e) \mid e \in P\} \leq \rho$.

The *most stable path* between a pair of vertices is defined as a ρ-stable path with the smallest value of ρ. Computing the Pareto frontier between the most stable and shortest path provides a complete spectrum of paths for path analysis. The problem of finding the most stable *s*-*t* path can be efficiently solved as a minimax path problem on an undirected graph using range minimum queries [2]. An algorithm for computing the Pareto frontier of stable paths is given in Figure 2 of Appendix A.3.

Discussion

In future work, we intend to show that the persistence diagrams of Jaccard Mappers are stable with respect to changes in filter function, data, and cover parameters. An implementation of the Jaccard Mapper can be found in the Scikit-TDA packages `kepler-mapper` and `cechmate`.[3]

[3] Found at `scikit-tda.org`, `kepler-mapper.scikit-tda.org`, and `cechmate.scikit-tda.org`

References

1 Mathieu Carrière, Bertrand Michel, and Steve Oudot. Statistical analysis and parameter selection for mapper. *Journal of Machine Learning Research*, 19(12):1–39, 2018.

2 E. D. Demaine, G. M. Landau, and O. Weimann. On cartesian trees and range minimum queries. In *Automata, Languages and Programming*, pages 341–353, 2009.

3 T. K. Dey, F. Mémoli, and Y. Wang. Multiscale mapper: Topological summarization via codomain covers. In *Proceedings of Symposium on Discrete Algorithms*, 2016.

4 N. Saul and D. L. Arendt. Machine learning explanations with topological data analysis. *Demo at the Workshop on Visualization for AI explainability (VISxAI)*, 2018.

5 G. Singh, F. Memoli, and G. Carlsson. Topological methods for the analysis of high dimensional data sets and 3D object recognition. *Symposium on Point-Based Graphics*, 2007.

A Proofs

A.1 Monotonic Filtration of Jaccard Nerve

The following is a proof for Theorem 3.

Proof. This proof makes use of standard set theory results. Let $\mathcal{U}$ be an arbitrary cover of some set X and let Nrv_J be its Jaccard nerve. We consider Nrv_J as a filtration by assigning as the birth time of simplex $\sigma \in \mathrm{Nrv}_J$ its weight w_σ. To show this is indeed a filtration, we focus on a single simplex σ and a face $\tau \preceq \sigma$ to show that the face always appears in the filtration before the simplex.

Suppose σ is generated from cover elements $\{U_i\}_{i\in I}$ over some index set I. Let a face $\tau \preceq \sigma$ be generated by cover elements indexed by a subset $J \subset I$. The birth time of τ is

$$d_J(\{U_i\}_{i\in J}) = 1 - \frac{|\cap_{i\in J} U_i|}{|\cup_{i\in J} U_i|}$$

and the birth time of σ is

$$d_J\left(\{U_i\}_{i\in I}\right) = 1 - \frac{|\cap_{i\in I} U_i|}{|\cup_{i\in I} U_i|}.$$

Clearly, with $\{U_i\}_{i\in J} \subset \{U_i\}_{i\in I}$, we have that $|\cap_{i\in J} U_i| \geq |\cap_{i\in I} U_i|$ and $|\cup_{i\in J} U_i| \leq |\cup_{i\in I} U_i|$. It follows then that $d_J(\tau) \leq d_J(\sigma)$. With K_α denoting the subcomplex that includes all simplices in Nrv_J with birth time at most $\alpha \in [0,1)$, for any $\alpha, \beta \in [0,1)$ with $\alpha < \beta$, we have $K_\alpha \subseteq K_\beta$. Hence $\mathrm{Nrv}_J(\mathcal{U})$ is a monotonic filtration. ◀

A.2 Čech equivalence

The following is a proof for Theorem 4 in the case of X sampled from the real-line.

Proof. Let X be a set of points in $\mathbb{R}$. For some subset $\{v_i\} \subset X$, let $\check{C}(\{v_i\})$ be the birth radius of the simplex σ defined by the subset of points. In $\mathbb{R}$, this can be computed as

$$\check{C}(\{v_i\}) = \frac{\max_i(v_i) - \min_i(v_i)}{2}.$$

The Jaccard distance between intervals centered on $\{v_i\}$ with some large radius R is defined as

$$\begin{aligned} d_J(\{v_i\}) &= 1 - \frac{\min(v_i + R) - \max(v_i - R)}{\max(v_i + R) - \min(v_i - R)} \\ &= 1 - \frac{\min(v_i) - \max(v_i) + 2R}{\max(v_i) - \min(v_i) + 2R}. \end{aligned}$$

We find the equivalence as

$$\check{C}(\{v_i\}) = \frac{d_J(\{v_i\}) \cdot R}{(2 - d_J(\{v_i\}))}$$

and

$$d_J(\{v_i\}) = 1 - \frac{R - \check{C}(\{v_i\})}{R + \check{C}(\{v_i\})}.$$

◀

We now detail experimental results suggesting that the 1-skeleton of the Jaccard filtration and the 1-skeleton the Čech filtration are isomorphic (i.e. the Vietoris-Rips filtration). To estimate the area of intersection of 1-spheres, we use Monte Carlo integration with uniform sampling. The first plot shows the 50 landmark points along with 20,000 points uniformly sampled around the landmarks. The middle plot shows the persistence diagrams of dimension 0 and 1 for the Vietoris-Rips filtration on the landmarks. Finally, we show an approximated Jaccard filtration on the landmarks, using the balls with radii 0.5 as the covers. We approximate the Jaccard filtration similarly as the Vietoris-Rips approximates Čech filtration, i.e. by only computing the 1-skeleton of the nerve, and including any higher order simplices for which all faces are already contained in the filtration, taking the maximum birth time of all faces.

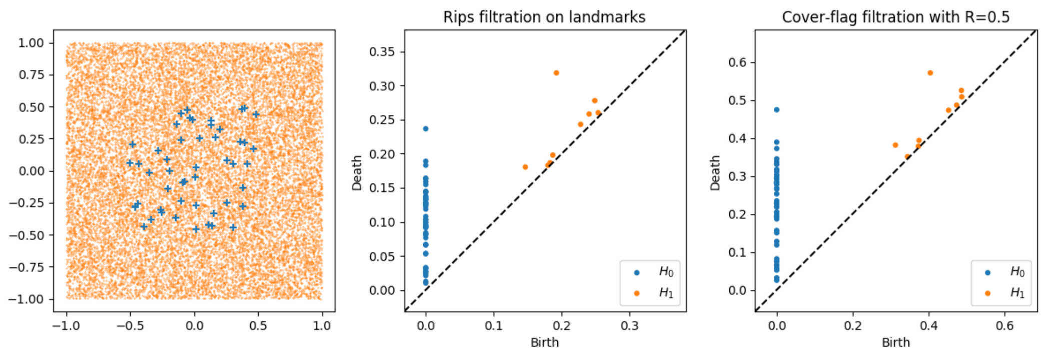


Figure 1 Persistence diagrams for the Vietrois-Rips Filtration and the approximate Jaccard filtration

The resulting diagrams are remarkably similar, hinting at some interesting relationship between the two constructions.

A.3 Algorithm to compute Pareto frontier of stable paths

In this algorithm, we repeatedly compute the shortest path while sweeping over the Jaccard Distance, akin to the process of computing persistent homology. This process results in a Pareto frontier, which balances the shortest paths with the stability of those paths. Figure 3 shows each path along the Pareto frontier between two vertices of a graph. This graph is a triangulation of the plane with random weights drawn from an inverted exponential distribution.

Figure 2 Algorithm to identify the Pareto frontier between shortest and most stable paths.

```
Input: 1-skeleton $G$ of Jaccard filtration and vertices $s,t$

set LIST $=[\emptyset,\emptyset]$ // stores $[P,\rho]$ pairs
while $s,t$ are connected in $G$
    compute shortest path $P$ between $s$ and $t$
    find $\rho=\max\{d_J(e)\,|\,e\in P\}$

    if LIST has no pair $[P',\rho']$ with $|P|=|P'|$
       add $[P,\rho]$ to LIST
    else if $\rho<\rho'$ for $[P',\rho']\in$LIST with $|P|=|P'|$
       replace $[P',\rho']$ with $[P,\rho]$ in LIST

    remove all edges $e$ from $G$ with $d_J(e)\geq\rho$

Return: LIST
```

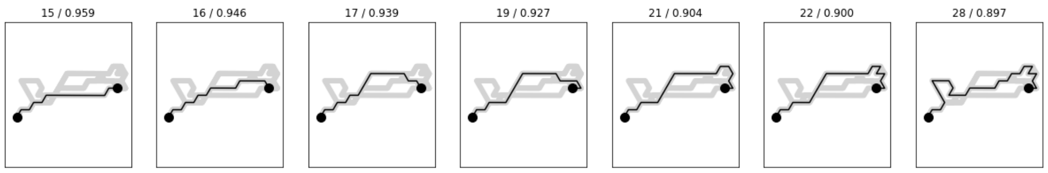


Figure 3 Visualization of each path on the Pareto frontier computed from a triangulation of the plane with random weights drawn from an inverted exponential distribution. Above each path depiction shows the length of the path and the stability score.

First Steps Towards Lower-Bounding the Number of Topological Descriptors for Reconstruction

Samuel Micka[1]
School of Computing, Montana State University, Bozeman, MT, USA
samuelmicka@montana.edu
https://orcid.org/0000-0003-3814-9727

David L. Millman[2]
School of Computing, Montana State University, Bozeman, MT, USA
david.millman@montana.edu
https://orcid.org/0000-0003-4112-3026

1 Introduction

Topological descriptors are used to represent complex data by multiple fields [7–10]. Turner, Mukherjee, and Boyer [11] showed that an uncountably infinite set of persistence diagrams (PDs) (or Euler characteristic curves (ECCs)) of height filtrations yields a unique representation of a geometric simplicial complex. This unique representation is capable of *reconstructing* the shape leveraging only information from the topological descriptors. As a result, researchers utilized finite subsets of the descriptors to represent shapes [2, 11]. Only particular (non-unique) subsets of descriptors can reconstruct complexes leading to the development of multiple algorithms to identifying sufficient descriptor sets [1,3,6]. We analyze the size of the sets by the number of directions from which they are generated.

Determining whether various descriptors require fewer directions than others remains an open question. Fasy et al. [5] observed that finding directions to reconstruct degree two vertices presents more difficulties with ECCs than PDs. In this work, we discuss differences between PDs, ECCs, and the effects of storing additional information in these descriptors. We offer first steps towards understanding the differences between the reconstructive ability of these descriptors and their effectiveness on particular classes of simplicial complexes.

1.1 Descriptors and Background

We assume the reader is knowledgeable on the PD and ECC, otherwise, we refer the reader to [11] for a description of the ECC and [4] for information on persistent homology. Additionally, we assume that descriptors are generated using the lower-star filtration from a direction in $\mathbb{S}^d$, see [4] for background on the lower-star filtration. In this work, the PD refers to the computed persistence diagram, where points (b, d) in which $b = d$ are stored on the diagonal, yielding geometric information about vertex locations and simplex births/deaths. We also consider an augmented version of the ECC which records the Betti number at the heights which vertices are encountered. We refer to this curve as the Betti Curve (BC).

2 Counting Descriptors for Reconstruction

In [3, 11] the ECC and PD are used interchangeably. We demonstrate that reconstructing particular shapes requires strictly fewer PDs than BCs and ECCs.

[1] SM is supported by NSF CCF 1618605.
[2] DLM is supported by NSF ABI 1661530.

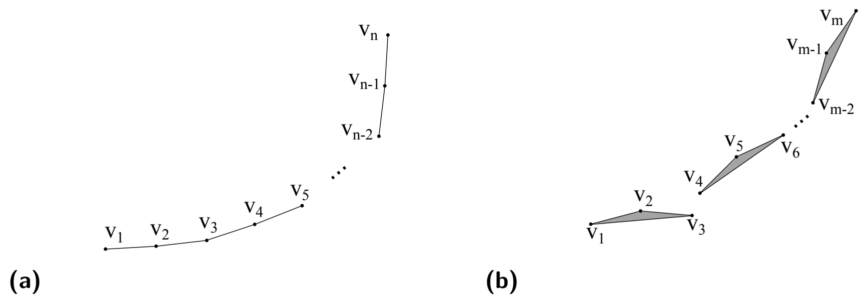


Figure 1 Intuition behinds proof of Theorem 1. On the left, $\Omega(n)$ directions are necessary to observe every vertex using the ECC. On the right, $\Omega(n)$ directions are necessary to observe every simplicial complex using the BC. Only four directions are necessary for the PD to reconstruct both examples.

▶ **Theorem 1.** *Let C_1 be the simplicial complex with n vertices in Figure 1a and C_2 be the simplicial complex with m vertices in Figure 1b embedded in $\mathbb{R}^2$. The embedding of C_1 and C_2 can be reconstructed with $O(1)$ PDs. However, C_1 requires $\Omega(n)$ ECCs and C_2 requires $\Omega(m)$ BCs or ECCs for reconstruction.*

First, we explain how C_1 and C_2 can be reconstructed with $O(1)$ PDs. The vertices in C_1 and C_2 meet the general position assumptions for Theorem 5 in [1] which provides a method for finding three PDs for determining vertex locations. For the edges in C_1, we generate constraints from direction $(-1, 0)$ to infer that the shape is chain of degree two vertices and eliminate edges that can not exist, leaving us with only the edges in the complex. For C_2, we can also infer connected components with the three directions for vertex reconstruction and $(-1, 0)$, the vertices in each connected component, and the number of faces. We find that there are $O(m)$ two-simplices and "fill in" the edges and faces. Thus, we can reconstruct C_1 and C_2 with four PDs. When reconstructing C_1 with the ECC, each degree two vertex can only be observed from particular regions of $\mathbb{S}^1$ which can grow arbitrarily small [5]. In the example found in Figure 1a, there are $O(n)$ distinct non-overlapping regions on $\mathbb{S}^1$ which must be sampled to observe each degree two vertex and, as a result, $\Omega(n)$ ECCs. Finally, when reconstructing C_2 using the BC, there exist multiple simplicial complexes that can generate the same BC from the same direction. Specifically, the BC requires that a direction observing each edge is sampled from $\mathbb{S}^1$ to determine if each connected component is a two-simplex or a chain of two edges. As such, $\Omega(m)$ BCs are required to reconstruct the complex. Since the BC contains strictly more information than the ECC, $\Omega(m)$ ECCs are necessary to reconstruct C_2 as well.

Theorem 1 is the first step towards identifying the differences in comparing the number of various descriptors necessary for simplicial complex reconstruction. We state the claim that $\Omega(n)$ ECCs are necesary for C_1 even though showing that $\Omega(m)$ ECCs are necessary for C_2 is sufficient for the theorem. However, we provide both constructions because we conjecture that C_1 can be reconstructed using $O(1)$ BCs. Future work includes: determining examples in which fewer BCs are necessary than ECCs, finding other types of subcomplexes (similar to the degree two vertex) that limit the ability of particular descriptors to reconstruct complexes, and experimentally comparing the distributions of regions that must be sampled from the sphere when using various descriptors for reconstruction.

References

1 Robin Lynne Belton, Brittany Terese Fasy, Rostik Mertz, Samuel Micka, David L. Millman, Daniel Salinas, Anna Schenfisch, Jordan Schupbach, and Lucia Williams. Learning simplicial complexes from persistence diagrams. In *Canadian Conference on Computational Geometry*, August 2018. Also available at arXiv:1805.10716.

2 Lorin Crawford, Anthea Monod, Andrew X. Chen, Sayan Mukherjee, and Raúl Rabadán. Functional data analysis using a topological summary statistic: The smooth Euler characteristic transform. Also available at arXiv:1611.06818, 2016.

3 Justin Curry, Sayan Mukherjee, and Katharine Turner. How many directions determine a shape and other sufficiency results for two topological transforms. Preprint available at arXiv:1805.09782, 2018.

4 Herbert Edelsbrunner and John Harer. *Computational Topology: An Introduction*. American Mathematical Society, 2010.

5 Brittany Terese Fasy, Samuel Micka, David L. Millman, Anna Schenfisch, and Lucia Williams. Challenges in reconstructing shapes from euler characteristic curves. October 2018. Also available at arXiv:1811.11337.

6 Robert Ghrist, Rachel Levanger, and Huy Mai. Persistent homology and Euler integral transforms. *Journal of Applied and Computational Topology*, 04 2018. doi:10.1007/s41468-018-0017-1.

7 Chad Giusti, Eva Pastalkova, Carina Curto, and Vladimir Itskov. Clique topology reveals intrinsic geometric structure in neural correlations. *Proceedings of the National Academy of Sciences*, 112(44):13455–13460, 2015.

8 Yongjin Lee, Senja D. Barthel, Paweł Dłotko, S. Mohamad Moosavi, Kathryn Hess, and Berend Smit. Quantifying similarity of pore-geometry in nanoporous materials. *Nature Communications*, 8:15396, 2017.

9 Monica Nicolau, Arnold J. Levine, and Gunnar Carlsson. Topology based data analysis identifies a subgroup of breast cancers with a unique mutational profile and excellent survival. *Proceedings of the National Academy of Sciences*, 108(17):7265–7270, 2011.

10 Abbas H. Rizvi, Pablo G. Camara, Elena K. Kandror, Thomas J. Roberts, Ira Schieren, Tom Maniatis, and Raul Rabadan. Single-cell topological RNA-seq analysis reveals insights into cellular differentiation and development. *Nature Biotechnology*, 35(6):551, 2017.

11 Katharine Turner, Sayan Mukherjee, and Doug M. Boyer. Persistent homology transform for modeling shapes and surfaces. *Information and Inference: A Journal of the IMA*, 3(4):310–344, 2014.

On the Average Time Complexity of the Reduction Algorithm for Persistent Homology

Michael Kerber
Graz University of Technology, Austria
kerber@tugraz.at

Hannah Schreiber
Graz University of Technology, Austria
hschreiber@tugraz.at

Abstract

All practically efficient algorithms for persistent homology are based on matrix reduction. While the worst case complexity is cubic, the experimental behavior tends to be linear. Our goal is to analyze the average time complexity of matrix reduction for standard models of random filtrations.

2012 ACM Subject Classification Theory of computation → Complexity classes; Theory of computation → Computational geometry

Keywords and phrases Persistent Homology, Matrix Reduction, Complexity analysis

Digital Object Identifier 10.4230/LIPIcs.CVIT.2016.23

Funding Austrian Science Fund (FWF) grant number P 29984-N35.

Persistent homology enables the analysis of evolving topological properties of general data sets through different scales. It found many applications (see, e.g. [2, 3]) and therefore developed a need for efficient computation. The most common and well-studied case is persistent homology of *filtrations*. A filtration $\mathcal{F} : \mathbb{K}_0 \to \dots \to \mathbb{K}_m$ is a sequence of nested complexes $\mathbb{K}_0 \subseteq \dots \subseteq \mathbb{K}_m$ and its persistent homology is represented by a barcode consisting of the birth and death times of the different cycle classes evolving through the growing complex. Despite good improvements (e.g. [1]), all efficient algorithms used in practice have a cubic worst case complexity. The reason is that they are based on matrix reduction. However, practical experiments tend more to a linear behavior, which contributes to the popularity of the method. The aim of this ongoing work is to analyze the average complexity of the matrix reduction algorithm for particular but common randomized filtration types such as Erdős-Rényi or Rips filtrations over random point sets in low dimension.

Reduction Algorithm. For a simplicial complex $\mathbb{K}$, let n_d be its number of d-simplices. The *boundary matrix* in dimension d of $\mathbb{K}$ is a $n_d \times n_{d+1}$-matrix whose columns are the boundaries of $\mathbb{K}$'s $(d+1)$-simplices. Then, the algorithm to compute the barcode in dimension d of a filtration $\mathcal{F} : \mathbb{K}_0 \to \dots \to \mathbb{K}_m$ consists of reducing the d^{th} boundary matrix of $\mathbb{K}_m$ with $\mathbb{Z}_2$-coefficients, whose columns and rows are ordered with respect to the order of appearance in $\mathcal{F}$ (see Algorithm 1). The barcode can then be read off the resulting reduced matrix. Therefore, in the worst case, the algorithm performs $n_{d+1} \cdot O(n_d) \cdot O(n_d)$ bit additions, which leads to cubical complexity.

Experimental results. We counted the number of matrix operations on different random filtrations of the 2-skeleton of a simplex. The instances went up to approximately half a million simplices. We focused on four models. For each, we created random instances, with n the number of simplices, increasing, and averaged the running time over ten repetitions for each n. We used linear regression to infer the empirical asymptotic behavior:

Algorithm 1: Reduction algorithm for the boundary matrix $\mathcal{M}$.

```
1 The function pivot returns the index of the lowest non-zero entry of the given column.
  for $i = 1 \ldots n_{d+1}$ do
2     while $\exists\, j \in \{1, ..., i-1\}$ pivot($\mathcal{M}[j]$) == pivot($\mathcal{M}[i]$) do
3         $\mathcal{M}[i] \leftarrow \mathcal{M}[i] + \mathcal{M}[j]$;
```

- Lower star filtrations: the vertices are added in random order and a simplex is added as soon as all its facets are included. The number of bit operations are independent of the vertex order and can be expressed by a deterministic formula. It is in the order of n.
- Rips filtrations: for a fixed number of random points in the unit square in $\mathbb{R}^2$, the edges are inserted in the order of their length. The triangles are inserted as soon as their boundary is inserted. Experimentally, the complexity tends to approximately $O(n^{1.3})$.
- Erdős-Rényi filtrations: for a fixed number of vertices, the edges are chosen in a random order. The triangles are inserted as soon as their boundary edges are inserted. Experimentally, the complexity tends to approximately $O(n^{1.6})$.
- Shuffled filtrations: for a fixed number of vertices, both for edges and triangles, the order is randomly chosen. Experimentally, the complexity tends to approximately $O(n^2)$.

A first theoretical result. We intend to analyze those precedent cases to verify the empirical observations. As an initial step, we look at a variant of the last case above.

Define a 3-*column* in a matrix as a column with exactly three non-zero entries. We define $\widetilde{\mathcal{M}}$ as a $m \times \binom{m}{3}$-matrix over $\mathbb{Z}_2$ with m rows and all possible 3-columns arranged in a random order (every column order is equiprobable). $\widetilde{\mathcal{M}}$ can be interpreted as the boundary matrix of a cell complex consisting of one vertex v, m distinct self-loops attached to v, and $\binom{m}{3}$ 2-cells bounded by three of the self-loops. Note that $\widetilde{\mathcal{M}}$ contains significantly more columns than the matrix of a shuffled filtration with the same number of edges.

Algorithm 1 applied to $\widetilde{\mathcal{M}}$ takes $O(m^3) \cdot O(m) \cdot O(m) = O(m^5)$ time in the worst case. We now slightly modify the algorithm as follows: If during the reduction, we encounter an input column c with the same pivot as a previously reduced column c', we first reduce c and then replace c' with the unreduced version of c. The idea is that c has initially only 3 non-zero entries, so subsequent reduction steps are less expensive. When enough reduced columns have been replaced, the reduction of the remaining columns becomes considerably cheaper than in the naive version. This idea leads to the following result:

▶ **Theorem 1.** *Let $\epsilon > 0$ be any constant and m be the number of edges. The time complexity of the modified reduction algorithm for $\widetilde{\mathcal{M}}$ is $O(m^{4+\epsilon})$ in expectation.*

References

1 U. Bauer, M. Kerber, J. Reininghaus, and H. Wagner. Phat – Persistent Homology Algorithms Toolbox. *Journal of Symbolic Computation*, 78:76–90, 2017.

2 M. Kerber. Persistent Homology: State of the art and challenges. *Internationale Mathematische Nachrichten*, 231:15–33, 2016.

3 S. Oudot. *Persistence Theory: From Quiver Representations to Data Analysis*, volume 209 of *Mathematical Surveys and Monographs.* American Mathematical Society, 2015.

A Toroidal Maxwell-Cremona-Delaunay Correspondence

Jeff Erickson and Patrick Lin
University of Illinois, Urbana-Champaign
{jeffe,plin15}@illinois.edu

We consider three classes of geodesic embeddings of graphs on the Euclidean flat torus: graphs having a positive equilibrium stress, reciprocal graphs (for which there is an orthogonal embedding of the dual graph), and weighted Delaunay complexes. The classical Maxwell-Cremona correspondence and the well-known correspondence between convex hulls and weighted Delaunay triangulations imply that these three concepts are essentially equivalent for plane graphs; indeed, all three conditions are equivalent to G being the projection of the 1-skeleton of a convex polyhedron in $\mathbb{R}^3$. However, this three-way equivalence does not extend directly to geodesic graphs on the torus. Reciprocal and Delaunay graphs are equivalent, and every reciprocal graph is in positive equilibrium, but not every positive equilibrium graph is reciprocal. We establish a weaker correspondence: Every positive equilibrium graph on any flat torus is equivalent to a reciprocal/Delaunay graph on *some* flat torus.

Definitions

Fix a non-singular matrix $M = \left[\begin{smallmatrix} a & b \\ c & d \end{smallmatrix}\right]$. The *flat torus* $\mathbb{T}_M$ is the 2-manifold obtained by identifying opposite sides of the parallelogram with vertices $(0,0), (a,c), (b,d), (a+b, c+d)$. The *unit square flat torus* is $\mathbb{T}_\square = \mathbb{T}_I$, where I is identity matrix $\left[\begin{smallmatrix} 1 & 0 \\ 0 & 1 \end{smallmatrix}\right]$. We consider graphs embedded on flat tori in which edges are *geodesics*—projections of straight line segments in the universal cover $\mathbb{R}^2$.

As usual, we regard each edge of an embedded graph as a pair of oppositely directed *darts*, each directed from one endpoint, called its *tail*, to the other endpoint, called its *head*. We write $u{\to}v$ to denote a dart with tail u and head v. We can specify any geodesic graph G on the unit square flat torus $\mathbb{T}_\square$ by identifying each vertex with a coordinate vector in $[0,1]^2$ and assigning a homology signature $[u{\to}v] \in \mathbb{Z}^2$ to each dart $u{\to}v$. The homology signature $[u{\to}v]$ records the number of times $u{\to}v$ crosses the vertical boundary of the unit square from left to right, and the number of times $u{\to}v$ crosses the horizontal boundary upward. Reversing any dart negates its homology signature. Exactly the same representation gives us a geodesic embedding of G on any other flat torus $\mathbb{T}_M$, via the linear transformation represented by the matrix M; we consider these to be the same embedding on different tori.

Every dart $u{\to}v$ in G has an associated *displacement vector* $\Delta_{u\to v} = (\Delta x_{u\to v}, \Delta y_{u\to v}) = v - u + [u{\to}v]$. Equivalently, $\Delta_{u\to v} = \hat{v} - \hat{u}$, where $\hat{u}{\to}\hat{v}$ is an arbitrary lift of $u{\to}v$ to the universal cover $\mathbb{R}^2$. Reversing a dart negates its displacement vector. When the choice of dart doesn't matter, we also write $\Delta_e = (\Delta x_e, \Delta y_e)$ for the displacement vector of (an arbitrary fixed dart of) edge e.

Fix a geodesic graph G on some flat torus $\mathbb{T}_M$. We consider three types of torus graphs.

- We call G an *equilibrium* graph if we can assign a *positive* real value ω_e to each edge of G, so that $\sum_{uv} \omega_{uv} \Delta_{u\to v} = (0,0)$ for every vertex v of G. The vector ω is called an *equilibrium stress*.
- We call G *reciprocal* if there is a geodesic embedding of the dual graph G^* on the same flat torus $\mathbb{T}_M$, such that each edge of G is orthogonal to its dual.
- Finally, we consider intrinsic weighted Delaunay complexes (dual to power diagrams) [1,2]. We call G *Delaunay* if we can assign a weight r_v^2 to each vertex v, so that G becomes the weighted Delaunay complex of its vertices.

Our definition of equilibrium restricts to *positive* stresses, unlike Borcea and Streinu [3] who study equilibrium stresses of flat torus graphs G that define periodic liftings from the (infinite planar) universal cover of G into $\mathbb{R}^3$: their stresses are necessarily not all positive.

Results

For planar graphs (with a fixed convex outer face), the Maxwell-Cremona correspondence [8, 9, 11, 15, 16] and the well-known correspondence between convex liftings and weighted Delaunay triangulations [1, 4, 12] imply equivalences between positive (interior) equilibrium stresses, orthogonal embeddings of the dual graph, and Delaunay vertex weights. These equivalences partially generalize to geodesic torus graphs.

▶ **Lemma 1.** *Let G be a geodesic torus graph. If ω is an equilibrium stress for G on any flat torus, then ω is an equilibrium stress for G on every flat torus.*

▶ **Lemma 2.** *Let G be a weighted Delaunay complex on some flat torus $\mathbb{T}$, and let G^* be the corresponding weighted Voronoi diagram on $\mathbb{T}$. Every edge e of G is orthogonal to its dual e^*.*

▶ **Lemma 3.** *Let G and G^* be dual geodesic graphs on some flat torus $\mathbb{T}_M$, such that every edge e of G is orthogonal to its dual e^*.*
(a) *G is a weighted Delaunay complex, and some translation of G^* is the corresponding weighted Voronoi diagram.*
(b) *The vector ω defined by $\omega_e = |e^*|/|e|$ is a positive equilibrium stress for G.*

Let G be a geodesic graph on some flat torus $\mathbb{T}$. We call a positive equilibrium stress vector ω for G a *reciprocal stress* if there is a geodesic embedding of the dual graph G^* on $\mathbb{T}$ such that for every edge e of G, e is orthogonal to its dual e^*, and $|e^*| = \omega_e \cdot |e|$.

▶ **Lemma 4.** *Not every positive equilibrium stress for G is a reciprocal stress. More generally, not every equilibrium graph on $\mathbb{T}$ is reciprocal/Delaunay on $\mathbb{T}$.*

We characterize which equilibrium stresses are reciprocal as follows. Any equilibrium stress ω for G defines three *isotropy* parameters, in terms of the displacement vectors of the edges of G on $\mathbb{T}_\square$.

$$\alpha = \sum_e \omega_e \Delta x_e^2, \qquad \beta = \sum_e \omega_e \Delta y_e^2, \qquad \gamma = \sum_e \omega_e \Delta x_e \Delta y_e.$$

▶ **Lemma 5.** *ω is a reciprocal stress for G on $\mathbb{T}_\square$ if and only if $(\alpha, \beta, \gamma) = (1, 1, 0)$.*

▶ **Theorem 6.** *A positive equilibrium stress vector ω is reciprocal on some flat torus if and only if $\alpha\beta - \gamma^2 = 1$. In particular, if $\alpha\beta - \gamma^2 = 1$, then ω is a reciprocal stress for G on the flat torus $\mathbb{T}_M$ if and only if $M = R\left[\begin{smallmatrix}1 & -\gamma \\ 0 & \beta\end{smallmatrix}\right]$ for some orthogonal matrix R.*

Note that $\alpha\beta - \gamma^2 = 1$ is merely a scaling condition; for any equilibrium stress vector ω, the scaled stress vector $\omega/\sqrt{\alpha\beta - \gamma^2}$ meets the conditions of Theorem 6. We conclude that every equilibrium graph on any flat torus is a weighted Delaunay complex on *some* flat torus.

Generalizations of Tutte's spring-embedding theorem [18] imply that for every essentially 3-connected graph G on any flat torus $\mathbb{T}$, and every positive stress vector ω, there is an isotopic embedding of G on $\mathbb{T}$ for which ω is an equilibrium stress [6, 13, 14, 17]. It follows that every essentially 3-connected torus graph G is isotopic to a weighted Delaunay complex on some flat torus. The existence of a *single* Delaunay embedding isotopic to G already follows from results of Colin de Verdière on circle-packing representations of surface graphs [5, 7], but our derivation characterizes the space of *all* Delaunay embeddings isotopic to G.

References

1 Franz Aurenhammer. Power diagrams: properties, algorithms and applications. *SIAM J. Comput.* 16(1):78–96, 1987.

2 Alexander I. Bobenko and Boris A. Springborn. A discrete Laplace-Beltrami operator for simplicial surfaces. *Discrete Comput. Geom.* 38:740–756, 2007.

3 Ciprian Borcea and Ileana Streinu. Liftings and stresses for planar periodic frameworks. *Discrete Comput. Geom.* 53(4):747–782, 2015. arXiv:1501.03549.

4 Kevin Q. Brown. Voronoi diagrams from convex hulls. *Inform. Process. Lett.* 9(5):223–228, 1979.

5 Yves Colin de Verdière. Empilements de cercles: Convergence d'une méthode de point fixe. *Forum Math.* 1(1):395–402, 1989.

6 Yves Colin de Verdière. Comment rendre géodésique une triangulation d'une surface? *L'Enseignment Mathématique* 37:201–212, 1991.

7 Yves Colin de Verdière. Un principe variationnel pour les empilements de cercles. *Invent. Math.* 104(1):655–669, 1991.

8 Henry Crapo and Walter Whiteley. Plane self stresses and projected polyhedra I: The basic pattern. *Topologie structurale / Structural Topology* 20:55–77, 1993. ⟨http://www-iri.upc.es/people/ros/StructuralTopology/ST20/st20.html⟩.

9 Henry Crapo and Walter Whiteley. Spaces of stresses, projections and parallel drawings for spherical polyhedra. *Beitr. Algebra Geom.* 35(2):259–281, 1994. ⟨https://www.emis.de/journals/BAG/vol.35/no.2/⟩.

10 Luigi Cremona. *Le figure reciproche nella statica grafica.* Tipografia di Giuseppe Bernardoni, 1872. ⟨http://www.luigi-cremona.it/download/Scritti_matematici/1872_statica_grafica.pdf⟩. English translation in [11].

11 Luigi Cremona. *Graphical Statics.* Oxford Univ. Press, 1890. ⟨https://archive.org/details/graphicalstatic02cremgoog⟩. English translation of [10] by Thomas Hudson Beare.

12 Herbert Edelsbrunner and Raimund Seidel. Voronoi diagrams and arrangements. *Discrete Comput. Geom.* 1(1):25–44, 1986.

13 Steven J. Gortler, Craig Gotsman, and Dylan Thurston. Discrete one-forms on meshes and applications to 3D mesh parameterization. *Comput. Aided Geom. Design* 23(2):83–112, 2006.

14 László Lovász. Discrete analytic functions: An exposition. *Eigenvalues of Laplacians and other geometric operators*, 241–273, 2004. Surveys in Differential Geometry 9, Int. Press.

15 James Clerk Maxwell. On reciprocal figures and diagrams of forces. *Phil. Mag. (Ser. 4)* 27(182):250–261, 1864.

16 James Clerk Maxwell. On reciprocal figures, frames, and diagrams of forces. *Trans. Royal Soc. Edinburgh* 26(1):1–40, 1870.

17 Dvir Steiner and Anath Fischer. Planar parameterization for closed 2-manifold genus-1 meshes. *Proc. 9th ACM Symp. Solid Modeling Appl.*, 83–91, 2004.

18 William T. Tutte. How to draw a graph. *Proc. London Math. Soc.* 13(3):743–768, 1963.

On Minimal-Perimeter Polyforms

Gill Barequet · Gil Ben-Shachar
Dept. of Computer Science, Technion—Israel Inst. of Technology, Haifa, Israel
{barequet,gilbe}@cs.technion.ac.il

Abstract

We discuss polyforms (lattice animals) on different lattices. We show a set of conditions which is sufficient for a family of polyforms to have the following property: Inflating a set of minimal-perimeter polyforms of a certain size yields all minimal-perimeter polyforms of a new, larger size.

2012 ACM Subject Classification Mathematics of computing → Combinatoric problems

Keywords and phrases Polyforms, Polyominoes, Polyhexes, Polyiamonds, Minimal-Perimeter

Digital Object Identifier 10.4230/LIPIcs...

Funding Work on this paper by both authors has been supported in part by ISF Grant 575/15.

1 Introduction

A polyform is a shape composed of a finite number of copies of a tile, connected by their edges. The best known examples of polyforms are polyominoes, polyhexes, and polyiamonds, in which the tiles are squares, hexagons, and triangles, respectively. The study of polyforms began independently in the 1950-60s in statistical physics [4] and in mathematics [5].

The perimeter of a polyform is the set of empty cells which share an edge with the polyform. A minimal-perimeter polyform is a polyform with the smallest possible perimeter for its size. Minimal-perimeter polyominoes were studied by Sieben [6] and Altshular et al. [1], both providing a characterization of all minimal-perimeter polyominoes that have the maximum size for a given perimeter size. The latter work was later generalized to polyhexes and polyiamonds by Vainsencher and Bruckstien [7].

Recently, we provided some results on the number of minimal-perimeter polyominoes [2, 3]. In this paper, we generalize these results to any type of polyforms satisfying some properties, and show that polyhexes and polyiamonds satisfy these conditions.

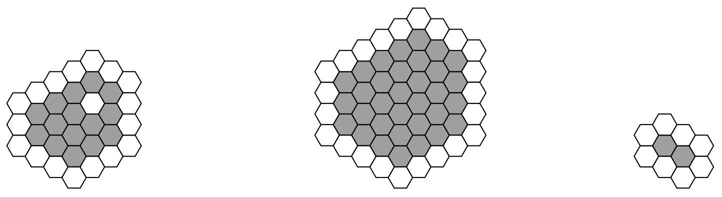
2 Preliminaries

Let Q be a polyform. The *perimeter* of Q, denoted as $\mathcal{P}(Q)$, is the set of all empty cells that are neighbors of at least one cell of Q. Similarly, the *border* of Q, denoted by $\mathcal{B}(Q)$, is the set of cells of Q that are neighbors of at least one empty cell.

The inflated polyform of Q is defined as $I(Q) := Q \cup \mathcal{P}(Q)$. Similarly, the deflated polyform of Q is defined as $D(Q) := Q \backslash \mathcal{B}(Q)$, these concepts are demonstrated in Figure 1. Let $\mathcal{F}$ be a family of polyforms on some lattice. Denote by $\epsilon^{\mathcal{F}}(n)$ the minimum perimeter of a polyform of type $\mathcal{F}$ and size n, and by $M_n^{\mathcal{F}}$ the set of all minimal-perimeter polyforms of type $\mathcal{F}$ and size n.

▶ **Theorem 1.** *[2, Thm. 4] Let $\mathcal{S}$ be the family of all polyominoes (polyforms on the square lattice). Then, for $n \geq 3$, we have that $\left|M_n^{\mathcal{S}}\right| = \left|M_{n+\epsilon^{\mathcal{S}}(n)}^{\mathcal{S}}\right|$.*

This theorem is a corollary of another theorem, stating that the inflation operation induces bijections between sets of minimal-perimeter polyominoes. The result is "chains" of sizes for which the number of minimal-perimeter polyominoes is identical. In this work,



(a) Q **(b)** $I(Q)$ **(c)** $D(Q)$

Figure 1 A polyhex Q, its inflated polyhex, and its deflated polyhex. The gray cells are the polyhex cells, while the white cells are the perimeter.

we generalize Theorem 1 from polyominoes to polyforms of any type, under a certain set of conditions.

3 Minimal-Perimeter Polyforms

Our main result is identifying a set of conditions, which is sufficient for a family of minimal-perimeter polyforms to satisfy a claim similar to that in Theorem 1.

▶ **Theorem 2.** ***Consider a polyform family $\mathcal{F}$, and the following set of conditions:***

- *The function $\epsilon^{\mathcal{F}}(n)$ is weakly monotone increasing.*
- *There exists some constant c, for which, for any minimal-perimeter polyform, we have that $\mathcal{P}(Q) = \mathcal{B}(Q) + c$.*
- *If Q is a minimal-perimeter polyform, then $I(Q)$ does not contain holes.*
- *Removing any single cell from a minimal-perimeter polyomino Q does not break the polyform into pieces, and adding any single cell to Q does not create a hole in it.*
- *If Q is a minimal-perimeter polyform, then $D(Q)$ is a valid (connected) polyform.*
- *If Q_1, Q_2 are two different minimal-perimeter polyforms, then $I(Q_1)$ and $I(Q_2)$ are different as well.*

If all the above conditions hold for $\mathcal{F}$, then we have that $\left|M_n^{\mathcal{F}}\right| = \left|M_{n+\epsilon^{\mathcal{F}}(n)}^{\mathcal{F}}\right|$. If these conditions are not satisfied for only some finite amount of sizes of polyforms, then the claim holds from some nominal size n_0.

As in the case of polyominoes, this theorem is the result of a bijection between minimal-perimeter polyforms, induced by the inflation operation. It can be shown that the above set of conditions is satisfied for both hexagonal and triangular lattices, hence, Theorem 2 holds in both cases. However, the second condition is not fulfilled in the cubical lattice in three and higher dimensions, and, indeed, it seems that the main property, subject of the theorem, does not hold in that case. Our next goal is to investigate the behavior of the inflation and deflation operations in higher dimensions.

References

1 Y. Altshuler, V. Yanovsky, D. Vainsencher, I.A. Wagner, and A.M. Bruckstein. On minimal perimeter polyminoes. In *Proc. Conf. Discrete Geometry for Computer Imagery*, pages 17–28, Szeged, Hungary, October 2006.

2 G. Barequet and G. Ben-Shachar. Properties of minimal-perimeter polyominoes. In *Proc. International Computing and Combinatorics Conference*, Qingdao, China, July 2018.

3 G. Barequet and G. Ben-Shachar. Minimal-perimeter polyominoes: Chains, roots, and algorithms. In *Proc. Conf. on Algorithms and Discrete Applied Mathematics*, pages 109–123, Kharagpur, India, February 2019.

4 S.R. Broadbent and J.M. Hammersley. Percolation processes: I. crystals and mazes. *Mathematical Proceedings of the Cambridge Philosophical Society*, 53(3):629–641, 1957.

5 M. Eden. A two-dimensional growth process. *Dynamics of fractal surfaces*, 4:223–239, 1961.

6 N. Sieben. Polyominoes with minimum site-perimeter and full set achievement games. *European J. of Combinatorics*, 29(1):108–117, 2008.

7 D. Vainsencher and A.M. Bruckstein. On isoperimetrically optimal polyforms. *Theoretical Computer Science*, 406(1-2):146–159, 2008.

A 1/4-Approximation Algorithm for the Maximum Hidden Vertex Set Problem in Simple Polygons *

Carlos Alegría[1]
Posgrado en Ciencia e Ingeniería de la Computación, UNAM, Mexico
calegria@uxmcc2.iimas.unam.mx

Pritam Bhattacharya[2]
Dept. of Computer Science & Engineering, Indian Institute of Technology Kharagpur, India
pritam.bhattacharya@cse.iitkgp.ernet.in

A well known class of visibility problems are those related to hiding. Given a simple polygon, we say that two points in its interior are *visible to each other* if the straight line segment connecting the points does not intersect the exterior of the polygon. Conversely, the points are said to be *hidden to each other* if they are not mutually visible. In this paper we study the Maximum Hidden Vertex Set (MHVS) problem, where given a simple polygon P, the objective is to find the largest possible subset of vertices of P such that every pair of vertices are hidden to each other. The MHVS problem is known to be NP-Hard [6]. In fact, it was shown to be APX-hard by Eidenbenz [3] even when the input polygon has no holes. Nevertheless, an exact solution can be computed in polynomial time for certain special classes of simple polygons. A simple polygon P is said to be *weakly visible* from an edge uv if every point $q \in P$ is visible from some point on uv. A maximum hidden vertex set can be computed in $O(n^2)$ time in a polygon weakly visible from a convex edge [4] (i.e. an edge lying between two convex vertices), and in $O(ne)$ time in so called *convex fans*, where e is the number of edges of its vertex visibility graph [5]. Metaheuristics have also been explored for obtaining approximate solutions for polygons without any holes [1]. In this paper, we present a $\frac{1}{4}$-approximation algorithm, which runs in $O(n^2)$ time, for finding the maximum hidden vertex set in an n-sided simple polygon containing no holes.

Let P be a simple polygon containing no holes. Our algorithm is based on a link distance based partitioning of P from Bhattacharya et al. [2] (which is itself adapted from the partitioning method used by Suri [7]) that partitions P into a collection of disjoint visibility windows. Given two points s and t inside P, the *link distance* between s and t is the minimum number of line segments required to connect them using a *link path*, which is basically a polyline in the interior of P. The visibility window decomposition given by Bhattacharya et al. [2] is essentially a hierarchical partitioning of P into weakly visible subpolygons, where any two subpolygons on the same level are at the same link distance from a chosen vertex p. In Figure 1, the link distances of the points x and y from p are 5 and 2 respectively.

As P is a simple polygon without any holes, the dual graph of this hierarchical partitioning is a tree. Each node of this tree represents a visibility polygon of the partition, and the children of a node are the regions inside the pockets formed by constructed edges belonging to their parent's visibility polygon. We classify these constructed edges as *left* or *right* constructed edges (indicated by the colours red and green respectively in Figure 1) based on whether a link path originating from p needs to take a left turn or a right turn to finally enter the weakly visible subpolygon created by it. Based on their level in the dual

* Preliminary results were obtained during the working and interacting sessions of the IRP-DCCG at UAB, Barcelona. We thank Subir Ghosh, Jayson Lynch, and Bodhayan Roy for useful discussions.
[1] Partially supported by the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No 734922.
[2] Supported by cycle 11 of the Tata Consultancy Services (TCS) Research Scholarship.

tree and on what type of constructed edge created them, our algorithm separates the weakly visible subpolygons into four disjoint subsets, which are as follows:

- R_1, containing subpolygons created by a left constructed edge at odd levels in the tree
- R_2, containing subpolygons created by a right constructed edge at odd levels in the tree
- R_3, containing subpolygons created by a left constructed edge at even levels in the tree
- R_4, containing subpolygons created by a right constructed edge at even levels in the tree

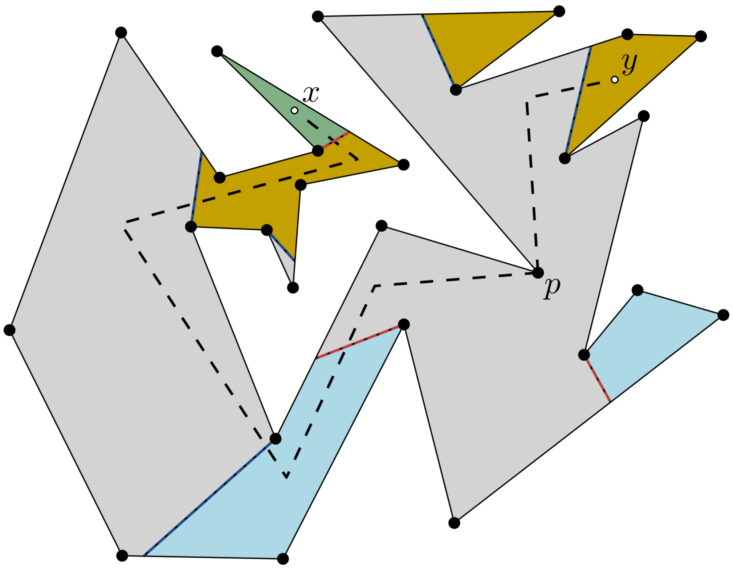


Figure 1 The partitioning of P into weakly visible subpolygons, where the colour of each subpolygon indicates whether it belongs to the set R_1, R_2, R_3 or R_4.

Observe that the vertices of P inside different regions of the same set are hidden from each other (see Figure 1), either because they belong to non-consecutive levels of the tree, or because both of them belong to the same level in the tree and are both created by a left (or right) constructed edge. Note that the separation process can be completed in $O(n)$ time. Observe that each subpolygon in the hierarchical partitioning of P is weakly visible from the constructed edge (of its parent's visibility polygon) that created it. So, within each subpolygon, we can compute the (exact) maximum hidden set of vertices using the algorithm by Ghosh et al. [4], which computes the maximum hidden set of a weak visibility polygon with n vertices in $O(n^2)$ time. Since the vertices of P inside two subpolygons belonging to the same set (from among R_1, R_2, R_3, R_4) are hidden from each other, the union of the maximum hidden sets of the subpolygons in each of these sets is a valid hidden set for P. Thus, we can clearly compute four valid hidden sets S_1, S_2, S_3, S_4, that correspond to the union of the maximum hidden sets computed for every subpolygon belonging to R_1, R_2, R_3, R_4 respectively, in $O(n^2)$ time. Out of these four valid hidden vertex sets of P, we choose the one containing the most number of vertices as our approximation of the actual maximum hidden set of P. If $\mathcal{S}_{\max}$ denotes the actual maximum hidden vertex set of P, then clearly: $\max(|S_1|, |S_2|, |S_3|, |S_4|) \geq (|S_1| + |S_2| + |S_3| + |S_4|)/4 \geq \frac{|\mathcal{S}_{\max}|}{4}$. Therefore, by choosing from among $S_1, \ldots, S_4$ the set containing the maximum number of vertices, we obtain a $\frac{1}{4}$-approximation of $\mathcal{S}_{\max}$. Note that the overall algorithm runs in $O(n^2)$ time. This leads us to our main result, which we summarize below.

▶ **Theorem 1.** *Given a simple polygon P with n vertices, there exists a $\frac{1}{4}$-approximation algorithm for computing the maximum hidden vertex set in P, which runs in $O(n^2)$ time.*

References

1 Antonio L. Bajuelos, Samtiago Canales, Gregorio Hernández, and A. Mafalda Martins. Estimating the maximum hidden vertex set in polygons. In *International Conference on Computational Sciences and Its Applications, ICCSA 2018*, pages 421–432, 2008. doi:10.1109/ICCSA.2008.19.

2 Pritam Bhattacharya, Subir Kumar Ghosh, and Sudebkumar Pal. Constant approximation algorithms for guarding simple polygons using vertex guards. *ArXiv e-prints*, 2017. arXiv:1712.05492v2.

3 Stephan Eidenbenz. Inapproximability of finding maximum hidden sets on polygons and terrains. *Computational Geometry*, 21(3):139–153, 2002. doi:10.1016/S0925-7721(01)00029-3.

4 Subir Kumar Ghosh, Anil Maheshwari, Sudebkumar Prasant Pal, Sanjeev Saluja, and C.E. Veni Madhavan. Characterizing and recognizing weak visibility polygons. *Computational Geometry*, 3(4):213–233, 1993. doi:10.1016/0925-7721(93)90010-4.

5 Subir Kumar Ghosh, Thomas Caton Shermer, Binay Kumar Bhattacharya, and Partha Pratim Goswami. Computing the maximum clique in the visibility graph of a simple polygon. *Journal of Discrete Algorithms*, 5(3):524–532, 2007. doi:10.1016/j.jda.2006.09.004.

6 T. Shermer. Hiding people in polygons. *Computing*, 42(2):109–131, 1989. doi:10.1007/BF02239742.

7 Subhash Suri. *Minimum link paths in polygons and related problems.* PhD thesis, The Johns Hopkins University, Baltimore, Maryland, 1987.

Hardness of Approximation for Geometric Set Cover and Related Problems

Sima Hajiaghaei Shanjani[1]
Department of Computer Science, University of Victoria, Canada
Sima@uvic.ca

1 Introduction

We study the problem of Geometric Set Cover, where the goal is to find a minimum sized cover for a given set X of n points in the plane by a family T of given objects. This is a fundamental problem which has been studied for over 30 years. It has long been known to be NP-hard and was shown to be APX-hard for a large class of geometric objects including axis-aligned rectangles and triangles assuming $P \neq NP$ by Chan and Grant in 2014 [2]. First, we study a version of Geometric Set Cover, where the given objects are axis-aligned rectangles. This version of the problem was shown to be APX-hard, but no explicit lower bound was known for it prior to this work. We present a specific constant $c = (1537/1536)$ and show that it is NP-hard to approximate within c factor of the optimum. This implies also the first specific constant c factor of approximation for the general Geometric Set Cover problem.

We also study Geometric Red-Blue Set Cover, where the points are given in two sets R and B, red elements and blue elements respectively, and the goal is to select a subfamily T' of given family T of geometric objects such that T' covers all the blue points while minimizing the number of covered reds. Chan and Hu in [3] show that the problem is NP-hard even when the objects are unit-squares. Finally, we consider the problem of Boxes Class Cover (BCC): points are given in two sets R and B, red points and blue points respectively, and the goal is to find the minimum number of axis-aligned rectangles T that cover all the blue points but no reds. This problem is introduced in 2012 by Bereg et al., who showed the problem is NP-hard [1]. Prior to this work, to the best of our knowledge, no hardness of approximation result has been shown for Geometric Red-Blue Set Cover and BCC.

2 Results and the Sketch of the Techniques

We present hardness of approximation proofs for some geometric problems addressed in Theorem 1. In the process of the proof, we also define a new version of MAX 3SAT problem in Definition 2, and then prove a hardness of approximation for it.

▶ **Theorem 1.** *The following problems are NP-hard to approximate within c factor of the optimum, where* $c = (1537/1536)$*: (i) Geometric Set Cover, even in the restricted case where all the objects are axis-aligned rectangles and each rectangle contains at most 5 points. (ii) Geometric Red-Blue Set Cover, even in the restricted case where all the objects are axis-aligned rectangles and each rectangle contains only one red point and at most 5 blue points. (iii) Boxes Class Cover*

▶ **Definition 2.** *Max Restricted Mixed 3SAT (MAX RM-3SAT).* This problem is a variant of MAX 3SAT where all the clauses are of size 2 and 3 and have the following properties:

1. All the clauses of size 3 have a literal in negated form and a literal in non-negated form.

[1] [Research funded by NSERC Discovery Grant RGPIN 2016-04234]

2. Any variable appears in exactly one clause of size 3, i.e., if v_i is a variable in this formula, only one of v_i or $\bar{v_i}$ can appear in any clause of size 3.
3. Any variable appears in exactly one of the clause of size 2 in negated form, and in exactly one of the clauses of size 2 in non-negated form.

Sketch of the techniques: We show the hardness results in Theorem 1 by series of reductions. In the first reduction, we transfer any instance of MAX E3SAT, the version of the MAX 3SAT problem in which each clause is of length exactly three, to an instance of Max RM-3SAT by renaming the variables and adding new clauses. Håstad [4] shows that it is NP-hard to approximate MAX E3SAT within a factor greater than 7/8 even when the problem is restricted to just satisfiable instances of the problem. We use this fact to and the reduction to show that it is NP-hard to approximate MAX RM-3SAT within 255/256 factor of the optimum.

In the second reduction for BCC, for any instance of MAX RM-3SAT we construct a point structure with red and blue points in polynomial time. Then we show a relation between the number of satisfied clauses in an optimal solution of MAX RM-3SAT and the size of the optimum solution in the corresponding instance of BBC. By considering the hardness result for MAX RM-3SAT, this relation implies the c-hardness result for BCC.

The third and forth reduction are modified version of the second one. For the reduction from MAX RM-3SAT to Geometric Set Cover, we modify the second reduction in a way that specific type of rectangles in BCC (called canonical), are considered as the family T of candidate rectangles and all the blue points as members of X. Then, for the reduction from MAX RM-3SAT to Geometric Red-Blue Set Cover, we will consider the same blue points in BCC as blue points and the same family T of candidate rectangles T used in Geometric Set Cover. For red points, we observe that there is a space in each of the candidate rectangles to which we can add exactly one distinct red point. These modified reductions lead to c-hardness result for both Geometric Set Cover and Geometric Red-Blue Set Cover.

Intuition of the structure used in MAX RM-3SAT $\rightarrow$ BCC: For Φ, an instance of MAX RM-3SAT, we change the order of the clauses to have all the clauses of size 3 first and then clauses of size 2. We rename the jth variable of the kth clause of this order to $X_{3(k-1)+j}$. Then, for each variable X_i, $1 \leq i \leq 3m$, we add 4 blue points on $(\pm 7i, \pm 7i)$ coordinates and 16 red points on $(\pm 7i \pm 1, \pm 7i \pm 1)$ coordinates.

The intuition of this structure is to locate the points in a way that an axis-aligned rectangle cannot cover points corresponding to two different variables together without covering a red point. We call an axis-aligned rectangle that covers two points with the same x (resp. y)-coordinate a *vertical* (resp. *horizontal rectangle*). We show that, the BCC on this arrangement of points has to have an optimal solution that covers blue points corresponding to each variable by exactly two rectangles, either both *vertical* or both *horizontal.* The idea of our reduction from MAX RM-3SAT to BCC is that the choice of vertical vs horizontal corresponds to true vs false assignment. For each clause, we add some red and blue points to force the choice of the covering blue-rectangles to be *horizontal* or *vertical* in the optimal solution of BCC based on the structure of the clauses of Φ. The location of these points are different in each type of clauses depending on the size of the clause and the number of negated literals in the clause. Here we only provide the coordinates of the added points for one type of the clauses to explain the idea more clearly. For each clause $c = (X_j \vee \bar{X}_{j+1} \vee \bar{X}_{j+2})$, we add three blue points on coordinates $(-7j, 7(j+1))$, $(7j, 7(j+2))$, and $(-7j+1, 7(j+2)-1)$. Moreover, we add nine red point on coordinates $(-7j-1, 7(j+1) \pm 1)$, $(-7j-1, 7(j+2)-1)$, $(-7j+1, 7(j+1)-1)$, $(-7j+1, 7(j+2)+1)$, $(7j+1, 7(j+2) \pm 1)$, $(7j-1, 7(j+2)+1)$, and $(-7j+2, 7(j+2)-2)$.

References

1 Sergey Bereg, Sergio Cabello, José Miguel Díaz-Báñez, Pablo Pérez-Lantero, Carlos Seara, and Inmaculada Ventura. The class cover problem with boxes. *Comput. Geom.*, 45(7):294–304, 2012. URL: https://doi.org/10.1016/j.comgeo.2012.01.014, doi:10.1016/j.comgeo.2012.01.014.

2 Timothy M. Chan and Elyot Grant. Exact algorithms and apx-hardness results for geometric packing and covering problems. *Comput. Geom.*, 47(2):112–124, 2014. URL: https://doi.org/10.1016/j.comgeo.2012.04.001, doi:10.1016/j.comgeo.2012.04.001.

3 Timothy M. Chan and Nan Hu. Geometric red-blue set cover for unit squares and related problems. *Comput. Geom.*, 48(5):380–385, 2015. URL: https://doi.org/10.1016/j.comgeo.2014.12.005, doi:10.1016/j.comgeo.2014.12.005.

4 Johan Håstad. Some optimal inapproximability results. In *Proceedings of the Twenty-Ninth Annual ACM Symposium on the Theory of Computing, El Paso, Texas, USA, May 4-6, 1997*, pages 1–10, 1997. URL: https://doi.org/10.1145/258533.258536, doi:10.1145/258533.258536.