Computing Oriented Spanners and Their Dilation

Kevin Buchin
TU Dortmund University, Germany

Antonia Kalb
TU Dortmund University, Germany

Anil Maheshwari
Carleton University, Ottawa, Canada

Saeed Odak
University of Ottawa, Canada

Carolin Rehs
TU Dortmund University, Germany

Michiel Smid
Carleton University, Ottawa, Canada

Sampson Wong
BARC, University of Copenhagen, Denmark

Abstract

Given a point set P in a metric space and a real number $t \geq 1$, an *oriented t-spanner* is an oriented graph $\overrightarrow{G} = (P, \overrightarrow{E})$, where for every pair of distinct points p and q in P, the shortest oriented closed walk in $\overrightarrow{G}$ that contains p and q is at most a factor t longer than the perimeter of the smallest triangle in P containing p and q. The *oriented dilation* of a graph $\overrightarrow{G}$ is the minimum t for which $\overrightarrow{G}$ is an oriented t-spanner.

For arbitrary point sets of size n in $\mathbb{R}^d$, where $d \geq 2$ is a constant, the only known oriented spanner construction is an oriented 2-spanner with $\binom{n}{2}$ edges. Moreover, there exists a set P of four points in the plane, for which the oriented dilation is larger than 1.46, for any oriented graph on P.

We present the first algorithm that computes, in Euclidean space, a sparse oriented spanner whose oriented dilation is bounded by a constant. More specifically, for any set of n points in $\mathbb{R}^d$, where d is a constant, we construct an oriented $(2+\varepsilon)$-spanner with $\mathcal{O}(n)$ edges in $\mathcal{O}(n \log n)$ time and $\mathcal{O}(n)$ space. Our construction uses the well-separated pair decomposition and an algorithm that computes a $(1+\varepsilon)$-approximation of the minimum-perimeter triangle in P containing two given query points in $\mathcal{O}(\log n)$ time.

While our algorithm is based on first computing a suitable undirected graph and then orienting it, we show that, in general, computing the orientation of an undirected graph that minimises its oriented dilation is NP-hard, even for point sets in the Euclidean plane.

We further prove that even if the oriented graph is already given, computing its oriented dilation is APSP-hard for points in a general metric space. We complement this result with an algorithm that approximates the oriented dilation of a given graph in subcubic time for point sets in $\mathbb{R}^d$, where d is a constant.

2012 ACM Subject Classification Theory of computation → Computational geometry

Keywords and phrases spanner, oriented graph, dilation, orientation, well-separated pair decomposition, minimum-perimeter triangle

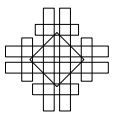
Digital Object Identifier 10.4230/LIPIcs.SoCG.2025.27

Related Version *Full Version*: https://arxiv.org/abs/2412.08165

Funding *Anil Maheshwari*, *Michiel Smid*: funded by the Natural Sciences and Engineering Research Council of Canada (NSERC).
Antonia Kalb: This work was supported by a fellowship of the German Academic Exchange Service (DAAD).

1 Introduction

While geometric spanners have been researched for decades (see [4, 19] for a survey), directed versions have only been considered more recently. This is surprising since, in many applications, edges may be directed or even oriented if two-way connections are not permitted. Oriented spanners were first proposed in ESA'23 [5] and have since been studied in [6] and [7].



Formally, let P be a point set in a metric space and let $\overrightarrow{G}$ be an oriented graph with vertex set P. For a real number $t \geq 1$, we say that $\overrightarrow{G}$ is an *oriented t-spanner*, if for every pair p, q of distinct points in P, the shortest oriented closed walk in $\overrightarrow{G}$ that contains p and q is at most a factor t longer than the shortest simple cycle that contains p and q in the complete undirected graph on P. Since we consider point sets in a metric space, we observe that the latter is the minimum-perimeter triangle in P containing p and q.

Recall that a *t-spanner* is an undirected graph G with vertex set P, in which for any two points p and q in P, there exists a path between p and q in G whose length is at most t times the distance $|pq|$. Several algorithms are known that compute $(1+\varepsilon)$-spanners with $\mathcal{O}(n)$ edges for any set of n points in $\mathbb{R}^d$. Examples are the greedy spanner [19], Θ-graphs [16] and spanners based on the well-separated pair decomposition (WSPD) [22]. Moreover, it is NP-hard to compute an undirected t-spanner with at most m edges [12].

In the oriented case, while it is NP-hard to compute an oriented t-spanner with at most m edges [5], the problem of constructing sparse oriented spanners has remained open until now. For arbitrary point sets of size n in $\mathbb{R}^d$, where $d \geq 2$ is a constant, the only known oriented spanner construction is a simple greedy algorithm that computes, in $\mathcal{O}(n^3)$ time, an oriented 2-spanner with $\binom{n}{2}$ edges [5]. If P consists of three vertices of an equilateral triangle in $\mathbb{R}^2$ and a fourth point in its centre, then the smallest oriented dilation for P is $2\sqrt{3} - 2 \approx 1.46$ [5]. Thus, prior to our work, no algorithms were known that compute an oriented t-spanner with a subquadratic number of edges for any constant t.

In this paper, we introduce the first algorithm to construct sparse oriented spanners for general point sets. For any set P of n points in $\mathbb{R}^d$, where d is a constant, the algorithm computes an oriented $(2+\varepsilon)$-spanner with $\mathcal{O}(n)$ edges in $\mathcal{O}(n \log n)$ time and $\mathcal{O}(n)$ space.

While our approach uses the WSPD [8, 22], this does not immediately yield an oriented spanner. An undirected spanner can be obtained from a WSPD by adding one edge per well-separated pair. This does not work for constructing an oriented spanner because a path in both directions needs to be considered for every well-separated pair. To construct such paths, we present a data structure that, given a pair of points, returns a point with an approximate minimum summed distance to the pair of the points, i.e. the smallest triangle in the complete undirected graph on P. Our algorithm first constructs an adequate undirected sparse graph and subsequently orients it using a greedy approach.

We also show in this paper that, in general, it is NP-hard to decide whether the edges of an arbitrary graph can be oriented in such a way that the oriented dilation of the resulting graph is below a given threshold. This holds even for Euclidean graphs.

For the problem of computing the oriented dilation of a given graph, there is a straightforward cubic time algorithm. We show a subcubic approximation algorithm.

The hardness of computing the dilation of a given graph, especially in the undirected case, is a long-standing open question, and only cubic time algorithms are known [15]. It is conjectured that computing the dilation is as hard as the all-pairs shortest paths problem (APSP). In this paper, we demonstrate that computing the oriented dilation is APSP-hard.

2 Preliminaries

Let $(P, |\cdot|)$ be a finite metric space, where the distance between any two points p and q in P is denoted by $|pq|$. When the choice between Euclidean and general metric distances is relevant for our results, we use the adjective *Euclidean* or respectively *metric graph*.

If p and q are distinct points in P, then $\Delta^*(p, q)$ denotes the shortest simple cycle containing p and q in the complete undirected graph on P; this is the triangle Δ_{pqx}, where

$$x = \arg\min\{(|px| + |qx|) \mid x \in P \setminus \{p, q\}\}.$$

We use $|\Delta^*(p, q)|$ to denote the perimeter of this triangle.

An *oriented graph* $\overrightarrow{G} = (P, \overrightarrow{E})$ is a directed graph such that for any two distinct points p and q in P, at most one of the two directed edges (p, q) and (q, p) is in $\overrightarrow{E}$. In other words, if (p, q) is in $\overrightarrow{E}$, then (q, p) is not in $\overrightarrow{E}$.

For two distinct points p and q in P, we denote by $C_{\overrightarrow{G}}(p, q)$ the *shortest closed walk* containing p and q in the oriented graph $\overrightarrow{G}$. The length of this walk is denoted by $|C_{\overrightarrow{G}}(p, q)|$; if such a walk does not exist, then $|C_{\overrightarrow{G}}(p, q)| = \infty$.

▶ **Definition 1** (oriented dilation). *Let $(P, |\cdot|)$ be a finite metric space and let $\overrightarrow{G}$ be an oriented graph with vertex set P. For two points $p, q \in P$, their* oriented dilation *is defined as*

$$\mathrm{odil}_{\overrightarrow{G}}(p, q) = \frac{|C_{\overrightarrow{G}}(p, q)|}{|\Delta^*(p, q)|}.$$

The oriented dilation of $\overrightarrow{G}$ is defined as $\mathrm{odil}(\overrightarrow{G}) = \max\{\mathrm{odil}_{\overrightarrow{G}}(p, q) \mid p, q \in P\}$.

An oriented graph with oriented dilation at most t is called an *oriented t-spanner*. Note thatthere is no oriented 1-spanner for sets of at least 4 points.

Let $G = (P, E)$ be an undirected graph. We call $\overrightarrow{G} = (P, \overrightarrow{E})$ an *orientation* of G, if for each undirected edge $\{p, q\} \in E$ it holds either $(p, q) \in \overrightarrow{E}$ or $(q, p) \in \overrightarrow{E}$.

2.1 Well-separated pair decomposition

In 1995, Callahan and Kosaraju [8] introduced the *well-separated pair decomposition* (WSPD) as a tool to solve various distance problems on multidimensional point sets.

For a real number $s > 0$, two point sets A and B form an *s-well-separated pair*, if A and B can each be covered by balls of the same radius, say ρ, such that the distance between the balls is at least $s \cdot \rho$. An *s-well-separated pair decomposition* for a set P of points is a sequence of s-well-separated pairs $\{A_i, B_i\}$, such that $A_i \cap B_i = \emptyset$ and for any two points $p, q \in P$, there is a unique index i such that $p \in A_i$ and $q \in B_i$, or $p \in B_i$ and $q \in A_i$.

The following properties are used repeatedly in this paper.

▶ **WSPD-Property 1** (Smid [22]). *Let $s > 0$ be a real number and let $\{A, B\}$ be an s-well-separated pair. For points $a, a' \in A$ and $b, b' \in B$, it holds that*

(i) $|aa'| \leq 2/s \cdot |ab|$, *and*

(ii) $|a'b'| \leq (1 + 4/s)|ab|$.

Callahan and Kosaraju [8] showed a fast WSPD-construction based on a split tree.

▶ **WSPD-Property 2** (Callahan and Kosaraju [8]). *Let P be a set of n points in $\mathbb{R}^d$, where d is a constant, and let $s > 0$ be a real number. An s-well-separated pair decomposition for P consisting of $\mathcal{O}(s^d n)$ pairs, can be computed in $\mathcal{O}(n \log n + s^d n)$ time.*

2.2 Approximate nearest neighbour queries

For approximate nearest neighbour queries, we want to preprocess a given point set P in a metric space, such that for any query point q in the metric space, we can report quickly its *approximate nearest neighbour* in P. That is, if q^* is the exact nearest neighbour of q in P, then the query algorithm returns a point q' in P such that $|qq'| \leq (1 + \varepsilon) \cdot |qq^*|$. Such a data structure is offered in [3] for the case when P is a point set in the Euclidean space $\mathbb{R}^d$:

▶ **Lemma 2** (Arya, Mount, Netanyahu, Silverman and Wu [3]). *Let P be a set of n points in $\mathbb{R}^d$, where d is a constant. In $\mathcal{O}(n \log n)$ time, a data structure of size $\mathcal{O}(n)$ can be constructed that supports the following operations:*

- *Given a real number $\varepsilon > 0$ and a query point q in $\mathbb{R}^d$, return a $(1+\varepsilon)$-approximate nearest neighbour of q in P in $\mathcal{O}(\log n)$ time.*
- *Inserting a point into P and deleting a point from P in $\mathcal{O}(\log n)$ time.*

3 Sparse oriented constant dilation spanner

This section presents the first construction for a sparse oriented $(2+\varepsilon)$-spanner for general point sets in $\mathbb{R}^d$. Our algorithm consists of the following four steps:

1. Compute the WSPD on the given point set.
2. For two points of each well-separated pair, approximate their minimum-perimeter triangle by our algorithm using approximate nearest neighbour queries presented in Section 3.1.
3. Construct an undirected graph whose edge set consists of edges of approximated triangles.
4. Orient the undirected graph using a greedy algorithm (see Lemma 3).

In Step 4, we modify a greedy algorithm from [5], which originally works as follows. Sort the $\binom{n}{3}$ triangles of the complete graph in ascending order by their perimeter. In this order, orient each triangle clockwise or anti-clockwise if possible; otherwise, we skip this triangle. At the end of the algorithm, all remaining edges are oriented arbitrarily. In [5], they show that this greedy algorithm yields an oriented 2-spanner. Note that this graph has $\binom{n}{2}$ edges.

Our modification of the greedy algorithm is to only consider the approximate minimum-perimeter triangles, rather than all $\binom{n}{3}$ triangles. Lemma 3 states that, for any two points whose triangles are $(1+\varepsilon)$-approximated in Step 2, the dilation between those points is at most $(2+2\varepsilon)$ in the resulting orientation.

▶ **Lemma 3.*** *Let P be a set of n points in $\mathbb{R}^d$, let $\varepsilon_1 > 0$ be a real number, and let L be a list of m point triples $(p,q;r)$, with p, q, and r being pairwise distinct points in P. Assume that for each $(p,q;r)$ in L,*

$$|\Delta_{pqr}| \leq (1+\varepsilon_1) \cdot |\Delta^*(p,q)|.$$

Let $G = (P,E)$ be the undirected graph whose edge set consists of all edges of the m triangles Δ_{pqr} defined by the triples $(p,q;r)$ in L. In $\mathcal{O}(m \log n)$ time, we can compute an orientation $\overrightarrow{G}$ of G, such that $\mathrm{odil}_{\overrightarrow{G}}(p,q) \leq 2+2\varepsilon_1$ for each $(p,q;r)$ in L.

▶ Remark. The above lemma only guarantees an upper bound on $\mathrm{odil}_{\overrightarrow{G}}(p,q)$ for $(p,q;r)$ in L. For such a triple, $\mathrm{odil}_{\overrightarrow{G}}(p,r)$ and $\mathrm{odil}_{\overrightarrow{G}}(q,r)$ can be arbitrarily large.

In Algorithm 1, we pick up to two points of A and up to two points of B for each well-separated pair $\{A,B\}$ and approximate the minimum-perimeter triangle for every combination of picked points, including pairs of points from the same set. The latter is necessary, to bound the size of the minimum triangle of two points in the same set (see Proof of Theorem 4). Considering all such combinations introduces a minor computational overhead, but improves the readability of the proof. Now, we show that Algorithm 1 constructs a sparse oriented $(2+\varepsilon)$-spanner for point sets in the Euclidean space $\mathbb{R}^d$:

▶ **Theorem 4.** *Given a set P of n points in $\mathbb{R}^d$ and a real number $0 < \varepsilon < 2$, an oriented $(2+\varepsilon)$-spanner for P with $\mathcal{O}(n)$ edges can be constructed in $\mathcal{O}(n \log n)$ time and $\mathcal{O}(n)$ space.*

* Due to space restrictions, proofs of results marked by $*$ are given in the full version of this paper.

Algorithm 1 Sparse $(2+\varepsilon)$-Spanner.

Require: Set P of n points in $\mathbb{R}^d$, positive reals $\varepsilon < 2$, ε_1, s
Ensure: $(2+\varepsilon)$-spanner for P with $\mathcal{O}(n)$ edges
  Compute an s-WSPD $\{A_i, B_i\}$ for $1 \le i \le m = \mathcal{O}(n)$
  Compute the data structure M on P to approximate nearest neighbours (see Lemma 2 [3])
  $L = \emptyset$
  for $i = 1$ **to** m **do**
    Pick $\min\{|A_i|, 2\}$ points of A_i
    Pick $\min\{|B_i|, 2\}$ points of B_i
    $K =$ set of all picked points
    for each $\{p, q\} \in K \times K$ with $p \neq q$ **do**
      $\Delta_{pqr} = (1+\varepsilon_1)$-approximation of $\Delta^*(p, q)$ computed by Algorithm 2 given M, p, q, ε_1
      Add the point triple $(p, q; r)$ to L
  return oriented graph generated by the algorithm in Lemma 3, given P, ε_1, L

Proof. Let the constants in Algorithm 1 be equal to $\varepsilon_1 = \varepsilon/4$ and $s = 96/\varepsilon$. Let $\overrightarrow{G}$ be the graph returned by Algorithm 1.

For any two points $p, q \in P$, we prove that $|C_{\overrightarrow{G}}(p, q)| \le (2+\varepsilon) \cdot |\Delta^*(p, q)|$. The proof is by induction on the rank of $|pq|$ in the sorted sequence of all $\binom{n}{2}$ pairwise distances.

If p, q is a closest pair (for the definition see [22]), then the pair $\{\{p\}, \{q\}\}$ is in the WSPD. Therefore, a $(1+\varepsilon_1)$-approximation of $\Delta^*(p, q)$ is added to the list L of triangles, that define the edge set of $\overrightarrow{G}$, and, due to Lemma 3 and $\varepsilon_1 = \varepsilon/4$, we have $\mathrm{odil}_{\overrightarrow{G}}(p, q) \le 2 + 2\varepsilon_1 \le 2 + \varepsilon$.

Assume that p, q is not a closest pair. Let $\{A, B\}$ be the well-separated pair with $p \in A$ and $q \in B$. We distinguish cases based on whether p and/or q are picked by Algorithm 1 while processing the pair $\{A, B\}$.

Case 1. Neither p nor q are picked by Algorithm 1.

This implies $|A| \ge 3$ and $|B| \ge 3$. Let $a \in A$ and $b \in B$ be picked points. We prove

$$|C_{\overrightarrow{G}}(p, q)| \le |C_{\overrightarrow{G}}(p, a)| + |C_{\overrightarrow{G}}(a, b)| + |C_{\overrightarrow{G}}(b, q)| \le (2+\varepsilon) \cdot |\Delta^*(p, q)|.$$

A sketch of this proof is visualized in Figure 1.

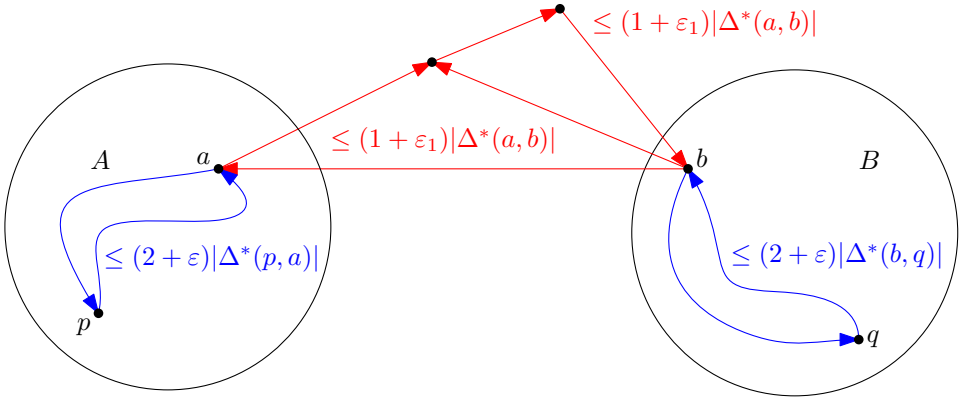


Figure 1 Visualization of Case 1: $|C_{\overrightarrow{G}}(p, q)| \le |C_{\overrightarrow{G}}(p, a)| + |C_{\overrightarrow{G}}(a, b)| + |C_{\overrightarrow{G}}(b, q)|$ in the proof of Theorem 4. $C_{\overrightarrow{G}}(a, b)$ is either a $(1+\varepsilon_1)$-approximation of $\Delta^*(a, b)$ or a union of two triangles (red), each of at most $(1+\varepsilon_1)|\Delta^*(a, b)|$. $|C_{\overrightarrow{G}}(p, a)|$ and $|C_{\overrightarrow{G}}(b, q)|$ (blue) are bounded inductively.

Since $|pa| < |pq|$ and $|bq| < |pq|$, we apply induction to bound $|C_{\overrightarrow{G}}(p,a)|$ and $|C_{\overrightarrow{G}}(b,q)|$. Since a and b are picked points, Lemma 3 applies, which gives us

$$|C_{\overrightarrow{G}}(p,q)| \leq (2+\varepsilon)|\Delta^*(p,a)| + (2+2\varepsilon_1)|\Delta^*(a,b)| + (2+\varepsilon)|\Delta^*(b,q)|.$$

Since $|A| \geq 3$, $|\Delta^*(p,a)|$ is at most the perimeter of any triangle with vertices p, a, and one other point in A. Each of the three edges of such a triangle can be bounded using WSPD-Property 1. It holds that

$$|\Delta^*(p,a)| \leq 3 \cdot 2/s \cdot |pq| \leq 3/s \cdot |\Delta^*(p,q)|, \tag{1}$$

where the last inequality $|\Delta^*(p,q)| \geq 2|pq|$ follows from the triangle inequality. Analogously, we bound $|\Delta^*(b,q)|$ by $3/s|\Delta^*(p,q)|$. By Lemma 5, we bound $|\Delta^*(a,b)|$ and achieve in total

$$|C_{\overrightarrow{G}}(p,q)| \leq 2(2+\varepsilon) \cdot 3/s|\Delta^*(p,q)| + (2+2\varepsilon_1)\,(1+8/s)\,|\Delta^*(p,q)|.$$

Since $\varepsilon_1 = \varepsilon/4$, $\varepsilon < 2$ and $s = 96/\varepsilon$, we conclude that

$$|C_{\overrightarrow{G}}(p,q)| \leq \left(2 + \frac{\varepsilon}{2} + \frac{28+10\varepsilon}{s}\right)|\Delta^*(p,q)| \leq \left(2 + \frac{\varepsilon}{2} + \frac{48}{s}\right)|\Delta^*(p,q)| = (2+\varepsilon)|\Delta^*(p,q)|.$$

Case 2. Either p or q is picked by Algorithm 1.

W.l.o.g., let p be picked for A and a distinct point b be picked for B.

Analogously to Case 1, we show that

$$|C_{\overrightarrow{G}}(p,q)| \leq |C_{\overrightarrow{G}}(p,b)| + |C_{\overrightarrow{G}}(b,q)| \leq (2+\varepsilon)|\Delta^*(p,q)|.$$

Case 3. Both p and q are picked by Algorithm 1.

An approximation of $\Delta^*(p,q)$ is added to L and then, due to Lemma 3 and $\varepsilon_1 = \frac{\varepsilon}{4}$, we have $\mathrm{odil}_{\overrightarrow{G}}(p,q) \leq 2 + 2\varepsilon_1 \leq 2 + \varepsilon$.

The initialisation is dominated by computing a WSPD in $\mathcal{O}(n \log n)$ time [8]. For each of the $\mathcal{O}(n)$ pairs, we pick at most four points and approximate the minimum triangle $\Delta^*(p,q)$ for at most six pairs $\{p,q\}$ of picked points. Therefore, computing the list L of $\mathcal{O}(n)$ triples costs $\mathcal{O}(n \log n)$ time (see Lemma 6) and orienting those costs $\mathcal{O}(n \log n)$ time (Lemma 3). Therefore, a $(2+\varepsilon)$-spanner can be computed in $\mathcal{O}(n \log n)$ time and $\mathcal{O}(n)$ space. ◀

3.1 The minimum-perimeter triangle

Let P be a set of n points in $\mathbb{R}^d$, and let p and q be two distinct points in P. The minimum triangle $\Delta^*(p,q)$ containing p and q can be computed naively in $O(n)$ time.

The following approach can be used to answer such a query in $O(n^{1-c})$ time, for some small constant $c > 0$ that depends on the dimension d. Note that computing $\Delta^*(p,q)$ is equivalent to computing the smallest real number $\rho > 0$, such that the ellipsoid

$$\sqrt{\sum_{i=1}^{d}(p_i - x_i)^2} + \sqrt{\sum_{i=1}^{d}(q_i - x_i)^2} \leq \rho$$

contains at least three points of P.

For a fixed real number ρ, we can count the number of points of P in the above ellipsoid: Using *linearization*, see Agarwal and Matousek [1], we convert P to a point set P' in a Euclidean space whose dimension depends on d. The problem then becomes that of counting

the number of points of P' in a query simplex. Chan [9] has shown that such queries can be answered in $O(n^{1-c})$ time, for some small constant $c > 0$. This result, combined with *parametric search*, see Megiddo [18], allows us to compute $\Delta^*(p, q)$ in $O(n^{1-c})$ time.

For many applications, including our $(2+\varepsilon)$-spanner construction, an approximation of the minimum triangle is sufficient. The following lemma shows that a WSPD with respect to $s = 2/\varepsilon$ provides a $(1+\varepsilon)$-approximation.

▶ **Lemma 5.*** *Let A and B be subsets of a point set P such that $\{A, B\}$ is an s-well-separated pair, and assume that $|B| \geq 2$.*

1. *For any point $a \in A$ and any two points $b, c \in B$, we have $|\Delta_{abc}| \leq (1 + 2/s)|\Delta^*(a, b)|$.*
2. *Assume that $|A| \geq 2$. For any two points $a, a' \in A$ and any two points $b, b' \in B$, it holds $|\Delta^*(a, b)| \leq (1 + 8/s)|\Delta^*(a', b')|$.*

Note that this works only if at least one subset of the well-separated pair contains at least two points. If both subsets have size one, we can use the following lemma to approximate the minimum triangle using approximate nearest neighbour queries.

▶ **Lemma 6.** *Let P be a set of n points in $\mathbb{R}^d$ and let $0 < \varepsilon_1 < 2$ be a real number. In $\mathcal{O}(n \log n)$ time, the set P can be preprocessed into a data structure of size $\mathcal{O}(n)$, such that, given any two distinct query points p and q in P, a $(1+\varepsilon_1)$-approximation of the minimum triangle $\Delta^*(p, q)$ can be computed in $\mathcal{O}(\log n)$ time.*

Proof. The data structure is the approximate nearest neighbour structure of Lemma 2 [3]. The query algorithm is presented in Algorithm 2. Let the constants in this algorithm be equal to $\varepsilon_2 = \varepsilon_1/2$, $\alpha = 4/\varepsilon_1$, and $\varepsilon_3 = \frac{2}{3\sqrt{d}} \cdot \varepsilon_1$.

Let p and q be two distinct points in P. We prove that Algorithm 2 returns a $(1+\varepsilon_1)$-approximation of $\Delta^*(p, q)$.

Algorithm 2 $(1+\varepsilon_1)$-Approximation of the Minimum-Perimeter Triangle.

Require: Query points $p, q \in P$, positive reals $\varepsilon_1 < 2$, ε_2, ε_3, $\alpha > 2$, data structure M to approximate nearest neighbours
Ensure: Triangle Δ such that $|\Delta| \leq (1+\varepsilon_1)|\Delta^*(p, q)|$
  $r = (1+\varepsilon_2)$-approximate nearest neighbour of p (which is neither p nor q)
  if $|pr| > \alpha|pq|$, **then return** Δ_{pqr}
  else
    Let B be the hypercube with sides of length $3\alpha|pq|$ that is centred at p and divided into cells with sides of length of $\frac{2}{3\sqrt{d}} \cdot \varepsilon_1|pq|$
    for each cell in B **do**
      $x_c = (1+\varepsilon_2)$-approximate nearest neighbour of the cell centre (with $x_c \neq p$, $x_c \neq q$)
    return the triangle Δ_{pqx_c} with the smallest perimeter across all cells

Throughout this proof, we denote the third point of $\Delta^*(p, q)$ by x. Let r be the $(1+\varepsilon_2)$-approximate nearest neighbour of p in $P \setminus \{p, q\}$ that is computed by the algorithm. By the triangle inequality, it holds

$$|\Delta_{pqr}| = |pq| + |qr| + |pr| \leq 2|pq| + 2|pr|. \tag{2}$$

Case 1. $|pr| > \alpha|pq|$.

Let p^* be the exact nearest neighbour of p in the set $P \setminus \{p, q\}$. Since r is a $(1+\varepsilon_2)$-approximate nearest neighbour of p in $P \setminus \{p, q\}$, and since $|\Delta^*(p, q)| \geq 2|px|$, we have

$$|pr| \leq (1+\varepsilon_2)|pp^*| \leq (1+\varepsilon_2)|px| \leq (1+\varepsilon_2)|\Delta^*(p, q)|/2.$$

Combining this inequality with Equation (2) and the assumption that $|pr| > \alpha|pq|$, we have

$$|\Delta_{pqr}| \leq (2+2/\alpha)|pr| \leq (1+1/\alpha)(1+\varepsilon_2)|\Delta^*(p, q)|.$$

Since $\varepsilon_2 = \varepsilon_1/2$, $\alpha = 4/\varepsilon_1$ and $\varepsilon_1 < 2$, we have

$$|\Delta_{pqr}| \leq (1+\varepsilon_1/4)(1+\varepsilon_1/2)|\Delta^*(p, q)| = (1+3\varepsilon_1/4+\varepsilon_1^2/8)|\Delta^*(p, q)| \leq (1+\varepsilon_1)|\Delta^*(p, q)|.$$

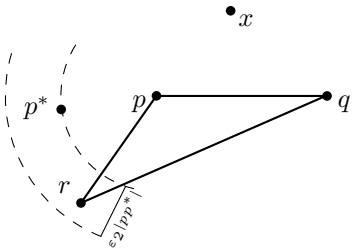


Figure 2 Visualization of Case 1 in the proof of Lemma 6. Since r is a $(1+\varepsilon_2)$-approximated nearest neighbour of p, the returned triangle Δ_{pqr} is a $(1+\varepsilon_1)$-approximation of $\Delta^*(p, q) = \Delta_{pqx}$.

Case 2. $|pr| \leq \alpha|pq|$.

Consider the hypercube B with sides of length $3\alpha|pq|$ that is centred at p.

We first prove that the third point x of $\Delta^*(p, q)$ is in B. Using Equation (2), the assumption that $|pr| \leq \alpha|pq|$, and the fact that $\alpha \geq 2$, we have

$$|\Delta^*(p, q)| \leq |\Delta_{pqr}| \leq (2+2\alpha)|pq| \leq 3\alpha|pq|.$$

Since $|\Delta^*(p, q)| \geq 2|px|$, it follows that $|px| \leq (3\alpha/2)|pq|$ and, therefore, $x \in B$.

Let C be the cell of B that contains x, let c be the centre of C, let c^* be the exact nearest neighbour of c in $P \setminus \{p, q\}$, and let x_c be the $(1+\varepsilon_2)$-approximate nearest neighbour of c that is computed by the algorithm. We first prove an upper bound on the distance $|xx_c|$:

$$|xx_c| \leq |xc| + |cx_c| \leq |xc| + (1+\varepsilon_2)|cc^*| \leq |xc| + (1+\varepsilon_2)|cx| = (2+\varepsilon_2)|cx| \leq 3|cx|.$$

Since x and c are in the same $\varepsilon_3|pq|$-sized cube and $\varepsilon_3 = \frac{2}{3\sqrt{d}} \cdot \varepsilon_1$, it follows that

$$|xx_c| \leq 3 \cdot \sqrt{d}/2\varepsilon_3|pq| = \varepsilon_1|pq| \leq (\varepsilon_1/2)|\Delta^*(p, q)|.$$

where the last inequality $|\Delta^*(p, q)| \geq 2|pq|$ follows from the triangle inequality.

Let $\Delta_{pqx_{c'}}$ be the triangle that is returned by the algorithm. Then

$$|\Delta_{pqx_{c'}}| \leq |\Delta_{pqx_c}| \leq |pq|+|qx|+|xx_c|+|x_cx|+|xp| = |\Delta^*(p, q)|+2|xx_c| \leq (1+\varepsilon_1)|\Delta^*(p, q)|.$$

The query algorithm performs two deletions, two insertions, and $1 + (3\alpha/\varepsilon_3)^d = \mathcal{O}(1)$ approximate nearest neighbour queries in the data structure of Lemma 2 [3]. Each such operation takes $\mathcal{O}(\log n)$ time. Therefore, the entire query algorithm takes $\mathcal{O}(\log n)$ time. ◀

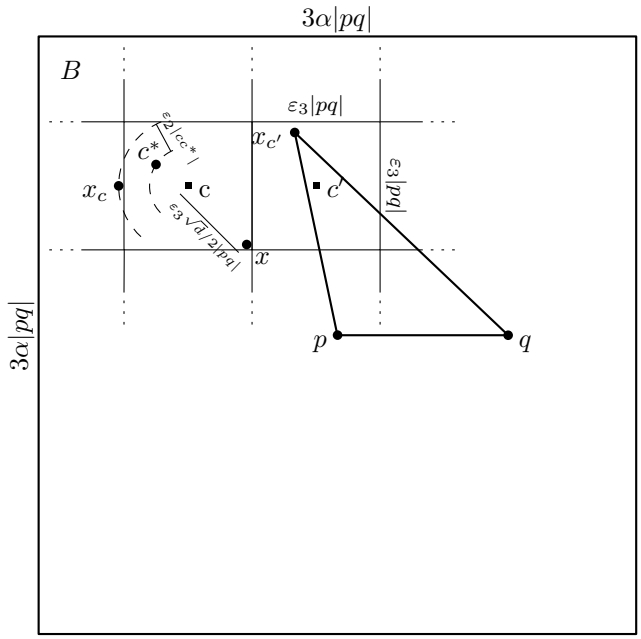


Figure 3 Visualization of Case 2 in the proof of Lemma 6. The third point x of $\Delta^*(p,q)$ lies in the cell centred at c. Since $\Delta_{pqx_{c'}}$ is returned, it holds $|\Delta_{pqx_{c'}}| \leq |\Delta_{pqx_c}|$, where $x_{c'}$ and x_c are $(1+\varepsilon_2)$-approximated nearest neighbours of c' and c.

3.2 NP-hardness

Our $(2+\varepsilon)$-spanner construction involves computing undirected $(1+\varepsilon)$-approximations of minimum-perimeter triangles, which are subsequently oriented. A similar approach is used in [5] where they prove, for convex point sets, orienting a greedy triangulation yields a plane oriented $O(1)$-spanner. Both results could be improved by optimally orienting the underlying undirected graph. The authors in [5] pose, as an open question, whether it is possible to compute, in polynomial time, an orientation of any given undirected geometric graph, that minimises the oriented dilation. We answer this question by showing NP-hardness.

▶ **Theorem 7.*** *Let P be a finite set of points in a metric space and let G be an undirected graph with vertex P. Given a real number t', it is NP-hard to decide if there exists an orientation $\overrightarrow{G}$ of G with oriented dilation* $\operatorname{odil}(\overrightarrow{G}) \leq t'$. *This is even true for point sets in the Euclidean plane.*

We give a proof idea, which is mainly described graphically, for details check the full version of this paper.

Proof sketch. We reduce from the NP-complete problem planar 3-SAT [17]. We start with a planar Boolean formula φ in conjunctive normal form with an incidence graph G_φ as illustrated in Figure 4. This graph can be embedded on a polynomial-size grid whose cells have side length one [13, 17]. Let $t' = 1.043$. We construct a graph G based on G_φ such that there is an orientation $\overrightarrow{G}$ of G with dilation $\operatorname{odil}(\overrightarrow{G}) \leq t'$ if and only if φ is satisfiable.

In the following, every point $p = (x, y)$ on the grid is replaced by a so-called *oriented point*, which is a pair of points $P = \{t(p), b(p)\}$ with *top* $t(p) = (x, y + \frac{\varepsilon}{2})$ and *bottom* $b(p) = (x, y - \frac{\varepsilon}{2})$, where $\varepsilon \geq 0$ is a small constant.

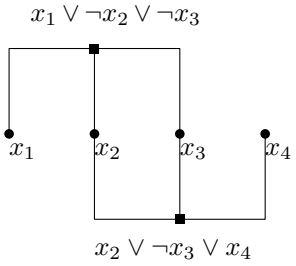


Figure 4 Example: Incidence graph of a planar 3-SAT formula embedded on a square grid.

We add an edge between $t(p)$ and $b(p)$, and its orientation encodes whether this points represent "true" or "false". W.l.o.g. we assume that an oriented edge from $b(p)$ to $t(p)$, in other words an *upwards* edge, represents "true", whereas an oriented edge from $t(p)$ to $b(p)$, in other words a *downwards* edge, represents "false".

First, we add (a polynomial number of) grid points on the edges such that all edges have length 1. Then, we replace all edges in the embedding of our formula graph G_φ by *wire* gadgets as shown in Figure 5. Note that wires propagate the orientation of oriented points – if two points next to each other on a wire have different orientations, their dilation is significantly larger than $t' = 1.043$, since the shortest oriented cycle needs to go through an additional oriented point. If they have the same orientation, their dilation is (almost) 1. To switch a signal (for a negated variable in a formula), we start the wire as in Figure 6.

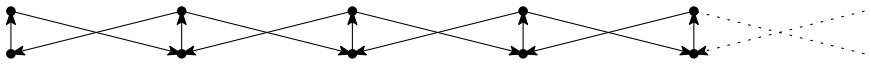


Figure 5 A wire where oriented points are oriented upwards.

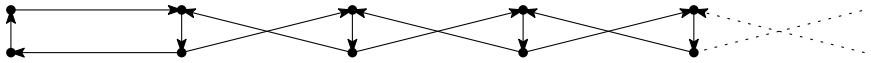


Figure 6 A wire where the orientation is switched to negate the signal.

To ensure that all clause gadgets encode the same orientation of oriented points as "true", we add a *tree of knowledge*. This is a tree with vertices on the grid shifted by $(0.5, 0.5)$ relative to the grid of G_φ and with wires as edges. The tree has two leaves per clause (see Figure 7). W.l.o.g., we assume that all oriented points of the tree of knowledge are oriented upwards (thus "true").

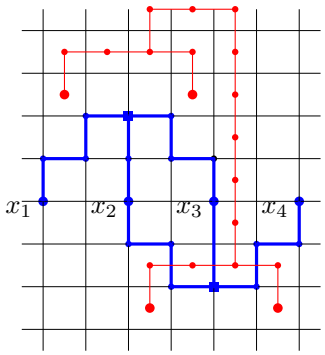


Figure 7 G_φ (blue) and its underlying grid together with a tree of knowledge (red).

All oriented points, which are not direct neighbours of a wire, are linked by a $K_{2,2}$ (compare to Figure 8). This ensures that the dilation of those points is (almost) 1. Note that the dilation of $t(p)$ and $b(p)$ is 1, regardless of whether p and p' are direct neighbours or not.

Figure 8 $K_{2,2}$ for oriented points with same orientation (left) and different orientation (right).

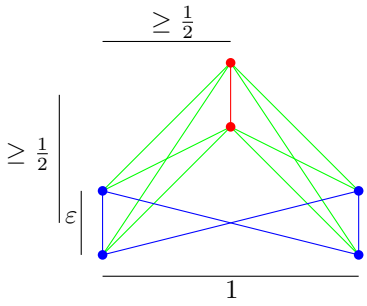


Figure 9 A wire (blue) can not be shortcut by $K_{2,2}$ gadgets (green).

Adding these $K_{2,2}$ gadgets does not affect the functionality of a wire gadget: Since any other point has at least distance $(0.5, 0.5)$ to two direct neighbours, a wire can not be shortcut by $K_{2,2}$ gadgets (compare to Figure 9).

For every clause, its two leaves in the tree of knowledge are not linked by a $K_{2,2,}$, but rather by a *clause gadget*. Figure 10 shows the two leaves and G_φ at the clause. We can assume that G_φ is embedded as shown, in particular, leaving the area directly above the clause empty. We illustrate a clause gadget in Figure 11, and in Figure 12 with more detail.

The oriented points L (left), R (right) and B (bottom) are the ends of the variable wires of a clause. They are placed such that they lie just inside an ellipse with the leaves l_1 and l_2 as foci, and without any other points in the ellipse (compare to Figures 10 and 11). As proven later, the gadget as shown in Figure 12 guarantees that, in order to obtain $\mathrm{odil}(L_1, L_2) \leq t'$, one of $\{L, R, B\}$ must lie on a shortest closed walk containing L_1, and L_2 and the orientation of that point must be the same as of L_1 and L_2 (thus, the literal is "true").

For each of $\{L, R, B\}$, we ensure there exists a *satellite* point, which is an oriented point on the variable wire, which is close but outside the ellipse. Its purpose is to ensure that the oriented dilation of L_1 (and likewise L_2) with L, B and R is below t', even if the orientation of that point is different as of L_1 and L_2, in other words, if the corresponding literal does not satisfy the clause.

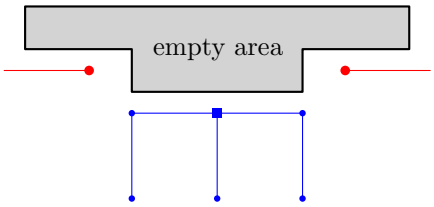


Figure 10 Embedded clause.

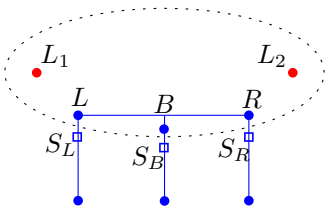


Figure 11 Clause gadget.

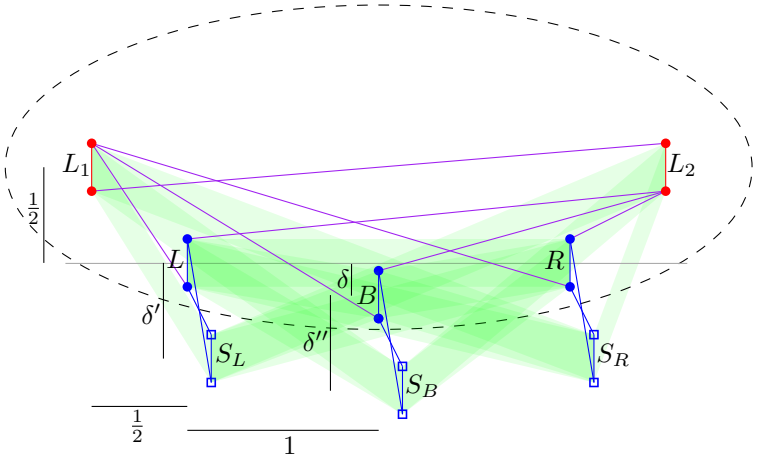


Figure 12 Detailed clause gadget. $K_{2,2}$ gadgets are indicated by green parallelograms.

By setting $\delta := 0.1335$, $\delta' := 0.0152$ and $\delta'' := 0.0304$, we obtain the following properties:

- The dilation of L_1 (and likewise L_2) with L, B and R is below t', even if the orientation of that point is different as of L_1 and L_2.
- If one of $\{L, B, R\}$ is oriented upwards, the dilation of L_1 and L_2 is smaller than $t' := 1.043$.
- If none of $\{L, B, R\}$ is oriented upwards, the dilation of L_1 and L_2 is greater than t'.

Thus, formula φ is satisfiable if and only if there exists an orientation of our constructed graph with dilation at most 1.043. ◀

Following the construction in the proof of Theorem 7, Figure 13 illustrates the graph for the formula $\varphi = (x_1 \vee \neg x_2 \vee \neq x_3) \wedge (x_2 \vee \neg x_3 \vee x_4)$ (see also Figures 4 and 7).

4 Oriented dilation

Computing the dilation of a given graph is related to the all-pairs shortest paths problem. There is a well-known cubic-time algorithm for APSP by Floyd and Warshall [11, 23]. By using their algorithm (and computing the minimum triangles naively), the oriented dilation of a given graph with n points can be computed in $\mathcal{O}(n^3)$ time.

We prove APSP-hardness for computing the oriented dilation. Since the APSP-problem is believed to need truly cubic time [20], it is unlikely that computing the oriented dilation for a given graph is possible in subcubic time.

Our hardness proof is a subcubic reduction to MinimumTriangle, which is the problem of finding a minimum weight cycle in an undirected graph with weights in $[7M, 9M]$, where the minimum weight cycle is a triangle. For two computational problems A and B, a *subcubic reduction* is a reduction from A to B where any subcubic time algorithm for B would imply a subcubic time algorithm for A (for a more thorough definition see [24]). We use the notation $A \leq_3 B$ to denote the existence of a subcubic reduction from A to B.

Vassilevska Williams and Williams [24] proved APSP-hardness for detecting if a weighted graph has a triangle of negative edge weight. Following thier proof, we prove APSP-hardness of the more specific problem MinimumTriangle:

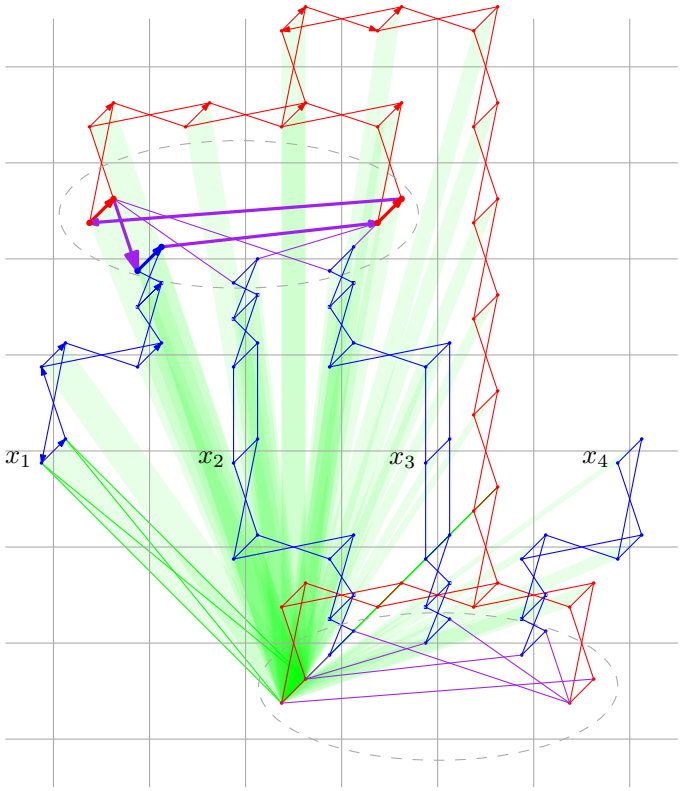


Figure 13 Graph constructed for $\varphi = (x_1 \vee \neg x_2 \vee \neg x_3) \wedge (x_2 \vee \neg x_3 \vee x_4)$. For visibility, oriented points are placed diagonally instead of vertically. Only for one point, all $K_{2,2}$-gadgets are indicated by green parallelograms. If the oriented point at the end of the wire from x_1 (blue) is – as indicated – oriented the same way as the tree of knowledge (red), this corresponds to setting it to true, resulting in an oriented closed walk in the clause gadget (purple) that gives a dilation smaller than 1.043.

▶ **Theorem 8.*** (APSP $\leq_3$ MinimumTriangle). *Let G be an undirected graph with weights in $[7M, 9M]$ whose minimum weight cycle is a triangle. Then, it is APSP-hard to find a minimum weight triangle in G.*

By reduction to MinimumTriangle, we prove APSP-hardness for ODIL, which is the problem of computing the oriented dilation of a given oriented graph in a metric space.

▶ **Theorem 9** (APSP $\leq_3$ Odil)**.** *Let $\overrightarrow{G}$ be an oriented metric graph. It is APSP-hard to compute the oriented dilation of $\overrightarrow{G}$.*

Proof. Due to Theorem 8, it is sufficient to prove MinimumTriangle $\leq_3$ Odil.

Let $G = (P, E)$ be an undirected graph with $|P| = n$ and weights $w : E \to [7M, 9M]$ in which the minimum weight cycle C of G is a triangle. Note that the weights of G fulfil the triangle inequality.

We define a metric space $(P', |\cdot|)$ with $P' = P \cup \{u, v\}$ by the following distances:

- $|pq| = w(p, q) + 15M$, for $\{p, q\} \in E$,
- $|pq| = 14M + 15M = 29M$, for $\{p, q\} \notin E$, and
- $|up| = |vp| = |vu| = 15M + 15M = 30M$ for $p \in P$.

So, for each missing edge of a complete graph on P, we assign its weight to be $14M$. Therefore, we obtain a metric with distances in $[7M, 14M]$. Then, we add the points u and v with distance $15M$ to each point in $p \in P$. To preserve the metric, we increase the distance of every tuple by $15M$. This construction is visualised in Figure 14.

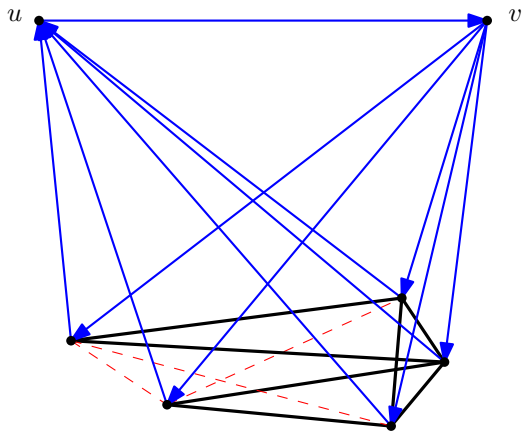


Figure 14 Graph construction in the proof of Theorem 8. Given an undirected graph on P (black), we add two points u, v with a large distance to all other points and augment it to a complete graph (red, dashed). The oriented graph $\overrightarrow{G}$ (blue) ensures that every shortest closed walk containing p and q in P visits u and v.

Let $\overrightarrow{G} = (P', \overrightarrow{E})$ be an oriented graph with $\overrightarrow{E} = \{(p,u),(v,p) \mid p \in P\} \cup \{(u,v)\}$.

We prove that the weight $w(C)$ of the minimum triangle can be computed by computing $\operatorname{odil}(\overrightarrow{G})$. In particular, we show that $w(C) = \frac{180M}{\operatorname{odil}(\overrightarrow{G})} - 45M$.

We proceed by computing the oriented dilation of $\overrightarrow{G}$. We first note that all edges of $\overrightarrow{G}$ have weight $30M$.

For every tuple of u or v and a point $p \in P$, the shortest closed walk is a triangle of length $|C_{\overrightarrow{G}}(u,p)| = |C_{\overrightarrow{G}}(v,p)| = |C_{\overrightarrow{G}}(u,v)| = 3 \cdot 30M = 90M$. Their minimum triangle contains two edges of length $30M$. Therefore, their dilation is

$$\operatorname{odil}(u,p) = \frac{|C_{\overrightarrow{G}}(u,p)|}{|\Delta^*(u,p)|} \leq \frac{90M}{2 \cdot 30M} = 1.5.$$

Analogously, it holds that $\operatorname{odil}(v,p) \leq 1.5$ for $p \in P$.

For u and v, it holds that $\operatorname{odil}(u,v) = 1$, because every triangle in P' containing u and v is an oriented closed walk in $\overrightarrow{G}$.

For two distinct points $p, q \in P$, the shortest closed walk containing p and q visits u and v and has length $|C_{\overrightarrow{G}}(p,q)| = 6 \cdot 30M = 180M$. For their minimum triangle holds $|\Delta^*(p,q)| \leq 3 \cdot 29M = 87M$. Since their dilation is larger than 2, a worst-case pair of $\overrightarrow{G}$ must be a pair in P. Therefore, the oriented dilation of the constructed graph $\overrightarrow{G}$ is

$$\operatorname{odil}(\overrightarrow{G}) = \max_{p,q \in P} \frac{|C_{\overrightarrow{G}}(p,q)|}{|\Delta^*(p,q)|} = \frac{180M}{\min\limits_{p,q,p^* \in P} w'(p,q) + w'(q,p^*) + w'(p,p^*)} > 2.$$

Since C is a triangle and $w(C) \leq 27M$, it holds that

$$\min_{p,q,p^* \in P} |pq| + |qp^*| + |p,p^*| = \min_{p,q,p^* \in P} w(p,q) + w(q,p^*) + w(p,p^*) + 3 \cdot 15M = w(C) + 45M.$$

Therefore, the oriented dilation of $\overrightarrow{G}$ is achieved by the minimum-length triangle C. Consequently, given the oriented dilation of $\overrightarrow{G}$, the length of C is $w(C) = \frac{180M}{\operatorname{odil}(\overrightarrow{G})} - 45M$.

The most expensive operation in our reduction is defining the metric on $\mathcal{O}(n^2)$ points with distances in $[22M, 30M]$. Following the definitions in [24], all instances have size $\mathcal{O}(n^2 \operatorname{polylog}(M))$ and all operations take subcubic time.

Thus, computing the oriented dilation of an oriented metric graph is at least as hard as finding a minimum weight triangle in an undirected graph. ◀

To compute the oriented dilation, testing all $\binom{n}{2}$ point tuples seems to be unavoidable. However, to approximate the oriented dilation, we show that, a linear set of tuples suffices.

▶ **Theorem 10.*** *Let P be a set of n points in $\mathbb{R}^d$, let $\overrightarrow{G}$ be an oriented graph on P, and let $\varepsilon > 0$ be a real number. Assume that, in $T(n)$ time, we can construct a data structure such that, for any two query points p and q in P, we can return, in $f(n)$ time, a k-approximation to the length of a shortest path in G between p and q. Then we can compute a value $\overline{\operatorname{odil}}$ in $\mathcal{O}(T(n) + n(\log n + f(n)))$ time, such that $(1-\varepsilon) \cdot \operatorname{odil}(\overrightarrow{G}) \leq \overline{\operatorname{odil}} \leq k \cdot \operatorname{odil}(\overrightarrow{G})$.*

Our approximation algorithm (Algorithm 3) uses the WSPD to achieve a suitable selection of $\mathcal{O}(n)$ point tuple that needs to be considered. The algorithm and its analysis work similar to our $(2+\varepsilon)$-spanner construction (compare to Algorithm 1). For each selected tuple, the algorithm approximates their oriented dilation by approximating their minimum triangle and shortest closed walk.

The precision and runtime of this approximation depend on approximating the length of a shortest path between two query points. The point-to-point shortest path problem in directed graphs is an extensively studied problem (see [14] for a survey). Alternatively, shortest path queries can be preprocessed via approximate APSP (see [10, 21, 25]). For planar directed graphs, the authors in [2] present a data structure build in $\mathcal{O}(n^2/r)$ time, such that the exact length of a shortest path between two query points can be returned in $\mathcal{O}(\sqrt{r})$ time. Setting $r = n^{2/3}$ minimizes the runtime of our Algorithm 3 to $\mathcal{O}(n^{4/3})$. It should be noted that even with an exact shortest path queries ($k = 1$) the oriented dilation remains an approximation due to the minimum-perimeter triangle approximation.

Algorithm 3 Approximate Oriented Dilation.

Require: Oriented graph $\overrightarrow{G} = (P, \overrightarrow{E})$, positive reals ε, ε_1, s, parameter k
Ensure: Value $\overline{\operatorname{odil}}$, such that $(1-\varepsilon)\operatorname{odil}(\overrightarrow{G}) \leq \overline{\operatorname{odil}} \leq k \operatorname{odil}(\overrightarrow{G})$
  Compute s-WSPD with the pairs $\{A_i, B_i\}$ for $1 \leq i \leq m$
  Compute the data structure M on P to approximate nearest neighbours (see Lemma 2 [3])
  $\overline{\operatorname{odil}} = 1$
  for $i = 1$ **to** m **do**
    $A =$ Pick $\min\{|A_i|, 2\}$ points of A_i
    $B =$ Pick $\min\{|B_i|, 2\}$ points of B_i
    for each $\{a, b\} \in A \times B$ **do**
      $\overline{d}(a,b) = k$-approximation of the length of a shortest path from a to b in $\overrightarrow{G}$
      $\overline{d}(b,a) = k$-approximation of the length of a shortest path from b to a in $\overrightarrow{G}$
      $\Delta_{abc} = (1+\varepsilon_1)$-approximation of $\Delta^*(a,b)$ by Algorithm 2 given M, a, b, ε_1
      $\overline{\operatorname{odil}} = \max\left\{\frac{\overline{d}(a,b)+\overline{d}(b,a)}{|\Delta_{abc}|}, \overline{\operatorname{odil}}\right\}$
  return $\overline{\operatorname{odil}}$

5 Conclusion and outlook

We present an algorithm for constructing a sparse oriented $(2+\varepsilon)$-spanners for multidimensional point sets. Our algorithm computes such a spanner efficiently in $O(n \log n)$ time for point sets of size n. In contrast, [5] presents a plane oriented $\mathcal{O}(1)$-spanner, but only for points in convex position. Developing algorithms for constructing plane oriented $\mathcal{O}(1)$-spanners for general two-dimensional point sets remains an open problem. Another natural open problem is to improve upon $t = 2 + \varepsilon$. For this, the question arises: Does every point set admit an oriented t-spanner with $t < 2$? Even for complete oriented graphs this is open.

Both algorithms could be improved by optimally orienting the underlying undirected graph, which is constructed first. We discard this idea by proving that, given an undirected graph, it is NP-hard to find its minimum dilation orientation. In particular, is the problem NP-hard for plane graphs or for complete graphs?

In the second part of this paper, we study the problem of computing the oriented dilation of a given graph. We prove APSP-hardness of this problem for metric graphs. We complement this by a subcubic approximation algorithm. This algorithm depends on approximating the length of a shortest path between two given points. Therefore, improving the approximation of a shortest path between two given points in directed/oriented graphs is an interesting problem. The APSP-hardness of computing the oriented dilation of metric graphs seems to be dominated by the computation of a minimum-perimeter triangle. However, for Euclidean graphs, the minimum-perimeter triangle containing two given points can be computed in $o(n)$ time. This raises the question: is computing the oriented dilation of a Euclidean graph easier than for general metric graphs?

Finally, as noted in [5], in many applications some bi-directed edges might be allowed. This opens up a whole new set of questions on the trade-off between dilation and the number of bi-directed edges. Since this is a generalisation of the oriented case, both of our hardness results also apply to such models.

References

1 Pankaj K. Agarwal and Jirí Matousek. On range searching with semialgebraic sets. *Discret. Comput. Geom.*, 11:393–418, 1994. doi:10.1007/BF02574015.

2 Srinivasa Rao Arikati, Danny Z. Chen, L. Paul Chew, Gautam Das, Michiel H. M. Smid, and Christos D. Zaroliagis. Planar spanners and approximate shortest path queries among obstacles in the plane. In Josep Díaz and Maria J. Serna, editors, *Proc. 4th Annu. European Sympos. Algorithms (ESA)*, volume 1136 of *Lecture Notes in Computer Science*, pages 514–528. Springer-Verlag, 1996. doi:10.1007/3-540-61680-2_79.

3 Sunil Arya, David M. Mount, Nathan S. Netanyahu, Ruth Silverman, and Angela Y. Wu. An optimal algorithm for approximate nearest neighbor searching fixed dimensions. *J. ACM*, 45(6):891–923, 1998. doi:10.1145/293347.293348.

4 Prosenjit Bose and Michiel H. M. Smid. On plane geometric spanners: A survey and open problems. *Comput. Geom. Theory Appl.*, 46(7):818–830, 2013. doi:10.1016/j.comgeo.2013.04.002.

5 Kevin Buchin, Joachim Gudmundsson, Antonia Kalb, Aleksandr Popov, Carolin Rehs, André van Renssen, and Sampson Wong. Oriented spanners. In Inge Li Gørtz, Martin Farach-Colton, Simon J. Puglisi, and Grzegorz Herman, editors, *Proc. 31st Annu. European Sympos. Algorithms (ESA)*, volume 274 of *LIPIcs*, pages 26:1–26:16. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023. doi:10.4230/LIPIcs.ESA.2023.26.

6 Kevin Buchin, Antonia Kalb, Guangping Li, and Carolin Rehs. Experimental analysis of oriented spanners on one-dimensional point sets. In *Proc. 36th Canad. Conf. Comput. Geom. (CCCG)*, pages 223–231, 2024.

7 Kevin Buchin, Antonia Kalb, Carolin Rehs, and André Schulz. Oriented dilation of undirected graphs. In *30th European Workshop on Computational Geometry (EuroCG)*, pages 65:1–65:8, 2024.

8 Paul B. Callahan and S. Rao Kosaraju. A decomposition of multidimensional point sets with applications to k-nearest-neighbors and n-body potential fields. *J. ACM*, 42(1):67–90, 1995. doi:10.1145/200836.200853.

9 Timothy M. Chan. Optimal partition trees. *Discret. Comput. Geom.*, 47(4):661–690, 2012. doi:10.1007/S00454-012-9410-Z.

10 Michal Dory, Sebastian Forster, Yael Kirkpatrick, Yasamin Nazari, Virginia Vassilevska Williams, and Tijn de Vos. Fast 2-approximate all-pairs shortest paths. In *Annu. ACM-SIAM Sympos. Discrete Algorithms (SODA)*, pages 4728–4757. SIAM, 2024. doi:10.1137/1.9781611977912.169.

11 Robert W. Floyd. Algorithm 97: Shortest path. *Communications of the ACM*, 5:345–345, 1962. doi:10.1145/367766.368168.

12 Panos Giannopoulos, Rolf Klein, Christian Knauer, Martin Kutz, and Dániel Marx. Computing geometric minimum-dilation graphs is NP-hard. *Internat. J. Comput. Geom. Appl.*, 20(2):147–173, 2010. doi:10.1142/S0218195910003244.

13 Michael Godau. *On the complexity of measuring the similarity between geometric objects in higher dimensions*. Dissertation, Free University Berlin, 1999. doi:10.17169/refubium-7780.

14 Andrew V. Goldberg and Chris Harrelson. Computing the shortest path: A search meets graph theory. In *Annu. ACM-SIAM Sympos. Discrete Algorithms (SODA)*, pages 156–165, 2005. doi:10.5555/1070432.1070455.

15 Joachim Gudmundsson and Christian Knauer. Dilation and detours in geometric networks. In *Handbook of Approximation Algorithms and Metaheuristics*, pages 53–69. Chapman and Hall/CRC, 2018. doi:10.1201/9781420010749-62.

16 J. Mark Keil and Carl A. Gutwin. Classes of graphs which approximate the complete euclidean graph. *Discret. Comput. Geom.*, 7:13–28, 1992. doi:10.1007/BF02187821.

17 David Lichtenstein. Planar formulae and their uses. *SIAM Journal on Computing*, 11(2):329–343, 1982. doi:10.1137/0211025.

18 Nimrod Megiddo. Applying parallel computation algorithms in the design of serial algorithms. *J. ACM*, 30(4):852–865, 1983. doi:10.1145/2157.322410.

19 Giri Narasimhan and Michiel H. M. Smid. *Geometric Spanner Networks*. Cambridge University Press, 2007. doi:10.1017/CBO9780511546884.

20 K. R. Udaya Kumar Reddy. A survey of the all-pairs shortest paths problem and its variants in graphs. *Acta Univ. Sapientiae Inform*, 8:16–40, 2016. doi:10.1515/ausi-2016-0002.

21 Barna Saha and Christopher Ye. Faster approximate all pairs shortest paths. In *Annu. ACM-SIAM Sympos. Discrete Algorithms (SODA)*, pages 4758–4827. SIAM, 2024. doi:10.1137/1.9781611977912.170.

22 Michiel H. M. Smid. The well-separated pair decomposition and its applications. In *Handbook of Approximation Algorithms and Metaheuristics (2)*. Chapman and Hall/CRC, 2018.

23 Stephen Warshall. A theorem on boolean matrices. *J. ACM*, 9(1):11–12, 1962. doi:10.1145/321105.321107.

24 Virginia Vassilevska Williams and R. Ryan Williams. Subcubic equivalences between path, matrix, and triangle problems. *J. ACM*, 65(5), 2018. doi:10.1145/3186893.

25 Uri Zwick. All pairs shortest paths using bridging sets and rectangular matrix multiplication. *J. ACM*, 49(3):289–317, 2002. doi:10.1145/567112.567114.