Title
Online Spanners in Euclidean and General Metrics

Author
Khodabandeh, Hadi

Publication Date
2022

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA,
IRVINE

Online Spanners in Euclidean and General Metrics

THESIS

submitted in partial satisfaction of the requirements
for the degree of

MASTER OF SCIENCE

in Computer Science

by

Hadi Khodabandeh

Thesis Committee:
Distinguished Professor David Eppstein, Chair
Distinguished Professor Michael T. Goodrich
Associate Professor Milena Mihail

2022

DEDICATION

To Arash and Pouneh,
two innocent students who lost their lives in the shot down of flight PS752.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ALGORITHMS

ACKNOWLEDGMENTS

I would like to first thank my advisor and my thesis committee chair, professor David Eppstein, for introducing me to this topic and guiding me through my adventure in graph theory and computational geometry, and helping me with the difficulties along the way. It has been an honor to be his student.

I learned a ton from my co-authors of this work: Sujoy Bhore, Arnold Filtser, and Csaba D. Tóth, who are prominent figures in this area, and I am honored for having the chance to work with them. I also thank my other thesis committee members: professors Michael Goodrich and Milena Mihail for reviewing this thesis and reflecting their valuable comments.

I would like to thank my family for supporting me from the other side of the world, and my friends and colleagues for being a valuable part of my life and my career. The theory lab in UCI gave me the chance to know all these beautiful people.

VITA

Hadi Khodabandeh

EDUCATION

Master of Science in Computer Science **2022**
University of California, Irvine *Irvine, California*

Bachelor of Science in Computer Engineering **2018**
Sharif University of Technology *Tehran, Iran*

RESEARCH EXPERIENCE

Graduate Research Assistant **2018–2022**
University of California, Irvine *Irvine, California*

Undergraduate Research Intern **2017**
Max Planck Institute for Informatics *Saarbrücken, Germany*

Undergraduate Research Intern **2016**
Hong Kong University of Science and Technology *Clear Water Bay, Hong Kong*

TEACHING EXPERIENCE

Teaching Assistant **2019–2022**
University of California, Irvine *Irvine, California*

REFEREED CONFERENCE PUBLICATIONS

Non-approximability and Polylogarithmic Approximations of the Single-Sink Unsplittable and Confluent Dynamic Flow Problems [53] — **Dec 2017**
ISAAC 2017

On the Edge Crossings of the Greedy Spanner [40] — **Jun 2021**
SoCG 2021

How to catch marathon cheaters: New approximation algorithms for tracking paths [54] — **Aug 2021**
WADS 2021

PREPRINTS

Optimal Spanners for Unit Ball Graphs in Doubling Metrics [41] — **Jun 2021**

An accurate non-accelerometer-based ppg motion artifact removal technique using cyclegan [91] — **Jun 2021**

Online Spanners in Metric Spaces [14] — **Oct 2021**

ABSTRACT OF THE THESIS

Online Spanners in Euclidean and General Metrics

By

Hadi Khodabandeh

Master of Science in Computer Science

University of California, Irvine, 2022

Distinguished Professor David Eppstein, Chair

Spanners are fundamental graph structures that preserve lengths of shortest paths in an input graph G, up to some multiplicative distortion. Given an edge-weighted graph $G = (V, E)$, a subgraph $H = (V, E_H)$ is a t-spanner of G, for $t \geq 1$, if for every $u, v \in V$, the distance between u and v in H is at most t times their distance than in G.

In this thesis, we study the existing literature on offline and online spanners, and we introduce some new results on online spanners in metric spaces. Suppose that we are given a sequence of points $(s_1, \ldots, s_n)$, where the points are presented one-by-one, i.e., point s_i is presented at the step i, and $S_i = \{s_1, \ldots, s_i\}$ for $i = 1, \ldots, n$. The objective of an online algorithm is to maintain a geometric t-spanner G_i for S_i for all i. The algorithm is allowed to add edges to the spanner when a new point arrives, however, it is not allowed to remove any edge from the spanner. The performance of an online algorithm is measured by its competitive ratio, which is the supremum, over all sequences of points, of the ratio between the weight of the spanner constructed by the algorithm and the minimum weight of a t-spanner on S_n. Here the weight of a spanner is the sum of all its edge weights.

Under the L_2-norm in $\mathbb{R}^d$ for arbitrary constant $d \in \mathbb{N}$, we present an online algorithm for $(1+\varepsilon)$-spanner with competitive ratio $O_d(\varepsilon^{-d} \log n)$, improving the previous bound of $O_d(\varepsilon^{-(d+1)} \log n)$. Moreover, the spanner maintained by the algorithm has $O_d(\varepsilon^{1-d} \log \varepsilon^{-1}) \cdot n$

edges, almost matching the (offline) optimal bound of $O_d(\varepsilon^{1-d}) \cdot n$. In the plane, a tighter analysis of the same algorithm provides an almost quadratic improvement of the competitive ratio to $O(\varepsilon^{-3/2} \log \varepsilon^{-1} \log n)$, by comparing the online spanner with an instance-optimal spanner directly, bypassing the comparison to an MST (i.e., lightness). As a counterpart, we design a sequence of points that yields a $\Omega_d(\varepsilon^{-d})$ lower bound for the competitive ratio for online $(1+\varepsilon)$-spanner algorithms in $\mathbb{R}^d$ under the L_1-norm.

Then we turn our attention to online spanners in general metrics. Note that, it is not possible to obtain a spanner with stretch less than 3 with a subquadratic number of edges, even in the offline settings, for general metrics. We analyze an online version of the celebrated greedy spanner algorithm, dubbed *ordered greedy*. With stretch factor $t = (2k-1)(1+\varepsilon)$ for $k \geq 2$ and $\varepsilon \in (0,1)$, we show that it maintains a spanner with $O(\varepsilon^{-1} \log \frac{k}{\varepsilon}) \cdot n^{1+\frac{1}{k}}$ edges and $O(\varepsilon^{-1} n^{\frac{1}{k}} \log^2 n)$ lightness for a sequence of n points in a metric space. We show that these bounds cannot be significantly improved, by introducing an instance that achieves an $\Omega(\frac{1}{k} \cdot n^{1/k})$ competitive ratio on both sparsity and lightness. Furthermore, we establish the trade-off among stretch, number of edges and lightness for points in ultrametrics, showing that one can maintain a $(2+\varepsilon)$-spanner for ultrametrics with $O(n \cdot \varepsilon^{-1} \log \varepsilon^{-1})$ edges and $O(\varepsilon^{-2})$ lightness.

Chapter 1

Introduction

1.1 Problem Definition

Let $\mathcal{M} = (P, \delta)$ be a finite metric space. Let $G = (P, E)$ be the graph on the points of P in $\mathcal{M}$ whose edges are weighted with the distances between their endpoints. The graph G is a *t-spanner*, for $t \geq 1$, if $\delta_G(u, v) \leq t \cdot \delta(u, v)$, where $\delta_G(u, v)$ is the length of the shortest path between u and v in G, and $\delta(u, v)$ is the distance between u and v in $\mathcal{M}$. The *stretch factor* t of G is the maximum distortion between the metrics δ and δ_G. Spanners were first introduced by Peleg and Schäffer [81], and since then they have turned out to be one of the fundamental graph structures with numerous applications in the area of distributed systems and communication, distributed queuing protocol, compact routing schemes, etc. [33, 68, 82, 83].

The study of Euclidean spanners, where $P \subset \mathbb{R}^d$ with L_2-norm, was initiated by Chew [30]. Since then a large body of research has been devoted to Euclidean spanners due to its vast range of applications across domains, such as topology control in wireless networks, efficient regression in metric spaces, approximate distance oracles, data structures, and many more [57, 62, 86, 90]. Some of the results generalize to metric spaces with constant doubling dimensions [23] (the doubling dimension of $\mathbb{R}^d$ is d).

1.1.1 Lightness and Sparsity

Lightness and sparsity are two fundamental parameters for spanners. The lightness of a spanner $G = (P, E)$ is the ratio $w(G)/w(MST)$ between the total weight of G and the weight of a minimum spanning tree (MST) on P. The sparsity of G is the ratio $|E(G)|/|E(MST)| \approx |E(G)|/|P|$ between the number of edges of H and an MST. Since every spanner is connected and thus contain a spanning tree, the lightness and sparsity of a spanner G, resp., are trivial lower bounds for the ratio of $w(G)$ and $|E(G)|$ to the optimum weight and the number of edges.

1.1.2 Online Spanners and Competitive Ratio

We are given a sequence of points $(s_1, \ldots, s_n)$ in a metric space, where the points are presented one-by-one, i.e., point s_i is revealed at the step i, and $S_i = \{s_1, \ldots, s_i\}$ for $i \in \{1, \ldots, n\}$. The objective of an online algorithm is to maintain a t-spanner G_i for S_i for all i. The algorithm is allowed to *add* edges to the spanner when a new point arrives, however it is not allowed to *remove* any edge from the spanner. Moreover, the algorithm does not know the value of the total number points in advance.

The performance of an online algorithm ALG is measured by comparing it to the offline optimum OPT using the standard notion of competitive ratio [21, Ch. 1]. The *competitive ratio* of an online t-spanner algorithm ALG is defined as $\sup_\sigma \frac{\mathrm{ALG}(\sigma)}{\mathrm{OPT}(\sigma)}$, where the supremum is taken over all input sequences σ, $\mathrm{OPT}(\sigma)$ is the minimum weight of a t-spanner for the (unordered) set of points in σ, and $\mathrm{ALG}(\sigma)$ denotes the weight of the t-spanner produced by ALG for this input sequence. Note that, in order to measure the competitive ratio it is important that σ is a finite sequence of points.

1.2 History

In the online minimum spanning tree problem, points of a finite metric space arrive one-by-one, and we need to connect each new point to a previous point to maintain a spanning tree. Imase and Waxman [69] proved $\Theta(\log n)$-competitiveness, which is the best possible bound. Later, Alon and Azar [2] studied this problem for points in Euclidean plane, and proved a lower bound $\Omega(\log n / \log\log n)$ for the competitive ratio. Their result was the first to analyze the impact of auxiliary points (Steiner points) on a geometric network problem in the online setting. Several algorithms were proposed over the years for the online minimum Steiner tree and Steiner forest problems, on graphs in both weighted and unweighted settings; see [1, 5, 12, 65, 79]. However, these algorithms do not provide any guarantee on the stretch factor. This leads to the following open problem.

Question 1.1. Determine bounds on the competitive ratios for the weight and the number of edges of online t-spanners, for $t \geq 1$.

Previously, Gupta et al. [64, Theorem 1.5] constructed online spanners for terminal pairs in the same model we consider here. The analysis of [64] implicitly implies that, given a sequence of n points in an online fashion in a general metric space, one can maintain a $O(\log n)$-spanner with $O(n)$ edges and $O(\log n)$ lightness, as pointed out by one of the authors [89]. Recent work on online *directed* spanners [59] is not comparable to our results.

In the geometric setting, $(1+\varepsilon)$-spanners are possible in any constant dimension $d \in \mathbb{N}$. Tight worst-case bounds $\Theta_d(\varepsilon^{-d})$ and $\Theta_d(\varepsilon^{1-d})$ on the lightness and sparsity of offline $(1+\varepsilon)$-spanners have recently been established by Le and Solomon [73]. Online Euclidean spanners in $\mathbb{R}^d$ have been introduced by Bhore and Tóth [17]. In the real line (1D), they have established a tight bound of $O((\varepsilon^{-1}/\log\varepsilon^{-1})\log n)$ for the competitive ratio of any online $(1+\varepsilon)$-spanner algorithm for n points. In dimensions $d \geq 2$, the dynamic algorithm DefSpanner of Gao et al. [52] maintains a $(1+\varepsilon)$-spanner with $O_d(\varepsilon^{-(d+1)}n)$ edges and $O_d(\varepsilon^{-(d+1)}\log n)$ lightness, and works under the online model (as it never deletes edges when new points arrive). However, no lower bound better than the 1-dimensional $\Omega((\varepsilon^{-1}/\log\varepsilon^{-1})\log n)$ is currently known in higher dimensions.

1.3 New Results

See Table 1.1 for an overview of our results.

Family	**Stretch**	**Size**	**Lightness**	**Ref/comments**
General metrics	$(2k-1)(1+\varepsilon)$	$O(\varepsilon^{-1}\log(\frac{1}{\varepsilon}))n^{1+\frac{1}{k}}$	$O(n^{\frac{1}{k}}\varepsilon^{-1}\log^2 n)$	Theorem 4.1
	$O(\log n)$	$O(n)$	$O(\log n)$	[64, 89]
α-HST	$2\frac{\alpha}{\alpha-1}$	$n-1$	1	Lemmas 4.8 and 4.9
Ultrametric	$O(\varepsilon^{-1})$	$n-1$	$1+\varepsilon$	Theorem 4.10
	$2+\varepsilon$	$O(n\varepsilon^{-1}\log\varepsilon^{-1})$	$O(\varepsilon^{-2})$	Theorem 4.12
Doubling d-space	$1+\varepsilon$	$\varepsilon^{-O(d)}\,n$	$\varepsilon^{-O(d)}\,\log n$	DefSpanner [52]
Euclidean d-space	$1+\varepsilon$	$O_d(\varepsilon^{-d})\,n$	$O_d(\varepsilon^{-(d+1)}\log n)$	DefSpanner [52]
	$1+\varepsilon$	$O_d(\varepsilon^{1-d})\,n$	$\Omega(\varepsilon^{-1}n)$	ordered Θ-graph [85]
	$1+\varepsilon$	$\tilde{O}_d(\varepsilon^{1-d})\,n$	$O(\varepsilon^{-d}\log n)$	Theorem 3.2
Real line	$1+\varepsilon$	$O(n)$	$\tilde{\Theta}(\varepsilon^{-1}\log n)$	ordered greedy [17]
Family	**Stretch**	**Size**	**Competitive Ratio**	**Ref/comments**
General metrics	$2k-1$	-	$\Omega(\frac{1}{k}\cdot n^{\frac{1}{k}})$	Theorem 4.6
Euclidean plane	$1+\varepsilon$	$\tilde{O}(\varepsilon^{-1})\,n$	$\tilde{O}(\varepsilon^{-3/2}\log n)$	Theorem 3.6
$\mathbb{R}^d$ with L_1-norm	$1+\varepsilon$	-	$\Omega(\varepsilon^{-d})$	Theorem 3.13

Table 1.1: Overview of online spanners algorithms. In the last three rows, we compare the spanner weight directly with the optimum weight (rather than the MST) to bound the competitive ratio.

1.3.1 Upper Bounds for Points in $\mathbb{R}^d$

Under the L_2-norm in $\mathbb{R}^d$, for arbitrary constant $d \in \mathbb{N}$, we present an online algorithm for $(1+\varepsilon)$-spanner with lightness $O_d(\varepsilon^{-d}\log n)$ and sparsity $O(\varepsilon^{1-d}\log\varepsilon^{-1})$ (Theorem 3.2 in Section 3.1). This improves upon the previous lightness bound of $O_d(\varepsilon^{-(d+1)}\log n)$ by Gao et al. [52, Lemma 3.8]. In the plane, we give a tighter analysis of the same algorithm and achieve an almost quadratic improvement of the *competitive ratio* to $O(\varepsilon^{-3/2}\log\varepsilon^{-1}\log n)$ (Theorem 3.6 in Section 3.2). Recall that in the offline setting, $\Theta(\varepsilon^{-2})$ is a tight worst-case bound for the lightness of a $(1+\varepsilon)$-spanner in the plane [73]. We obtain a better dependence on ε by comparing the online spanner with an instance-optimal spanner directly, bypassing the comparison to an MST (i.e., lightness). The logarithmic dependence on n cannot be eliminated in the online setting, based on the lower bound in $\mathbb{R}^1$ [17].

1.3.2 Lower Bounds for Points in $\mathbb{R}^d$

As a counterpart, we design a sequence of points that yields a $\Omega_d(\varepsilon^{-d})$ lower bound for the competitive ratio for online $(1+\varepsilon)$-spanner algorithms in $\mathbb{R}^d$ under the L_1-norm (Theorem 3.13 in Section 3.3). This improves the previous bound of $\Omega(\varepsilon^{-2}/\log\varepsilon^{-1})$ in $\mathbb{R}^2$ under the L_1-norm. It remains open whether a similar lower bound holds in $\mathbb{R}^d$ under the L_2-norm; the current best lower bound is $\Omega((\varepsilon^{-1}/\log\varepsilon^{-1})\log n)$, established in [17], holds already for the real line ($d=1$).

1.3.3 Points in General Metrics

In Section 4.1, we study online spanners in general metrics. Note that it is not possible to obtain a spanner with stretch less than 3 with a subquadratic number of edges, even in the offline settings, for general metrics. We analyze an online version of the celebrated greedy spanner algorithm, dubbed *ordered greedy*. With stretch factor $t = (2k-1)(1+\varepsilon)$ for $k \geq 2$ and $\varepsilon \in (0,1)$, we show that it maintains a spanner with $O(\varepsilon^{-1}\log\frac{1}{\varepsilon})\cdot n^{1+\frac{1}{k}}$ edges and $O(\varepsilon^{-1}n^{\frac{1}{k}}\log^2 n)$ lightness for a sequence of n points in a metric space (Theorem 4.1). We show (in Theorem 4.6) that these bounds cannot be significantly improved, by introducing an instance where every online algorithm will have $\Omega(\frac{1}{k}\cdot n^{1/k})$ competitive ratio on both sparsity and lightness. Next, we establish the trade-off among stretch, number of edges and lightness for points in ultrametrics. Specifically, we show that it is possible to maintain a $(2+\varepsilon)$-spanner with $O(\varepsilon^{-1}\log\varepsilon^{-1})\cdot n$ edges and $O(\varepsilon^{-2})$ lightness in ultrametrics (Theorem 4.12). Note that as the uniform metric (shortest path on a clique) is an ultrametric, any subquadratic spanner must have stretch at least 2.

Chapter 2

Related Work

There has been an extensive effort in the literature for finding bounded-degree and lightweight spanners for different graph classes or proving the efficiency of the existing constructions. From the most general case of weighted graphs [75] to graphs of bounded width [22, 39] and graphs whose weights are coming from a more restricted Euclidean or doubling metric space [49, 23]. Different models have also been considered for this problem. In the offline model the points are given at the start of the algorithm, and the algorithm decides which edges to include in the spanner. In the online model, the points are given one by one to the algorithm, and in each step, the algorithm has to provide a spanner to the current point set. The algorithm cannot remove a previously added edge in this model. In the dynamic setting, the algorithm can remove previous edges. The objective in this case can be maintaining a constant lightness via minimal changes or a via a fast update time. It is known that the update time for vertex addition in the online and dynamic models is lower bounded by $\Omega(\log n)$ [58] which comes from the close relationship of the spanning property and finding approximate nearest neighbors [37, 77] of the new point. In the rest of this section, we briefly cover some of the existing algorithms and results in each of the aforementioned models.

2.1 Geometric Spanners in the offline setting

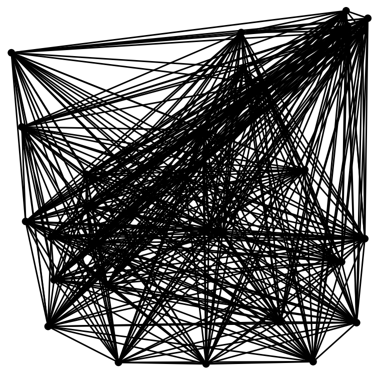
In the offline settings, where the points are given to the algorithm at the beginning of the execution, there are various ways to construct a $(1+\varepsilon)$-spanner for a point set that is located in a Euclidean space of dimension d. The Yao-graph [90] is one of these constructions, the greedy spanner is another construction, while WSPDs [87] can also yield a spanner construction if the pairs in a WSPD have small diameter with respect to their distance. Among the spanner constructions for Euclidean spaces, however, the greedy spanner is known to have the highest quality.

A greedy spanner can be constructed by running the greedy spanner algorithm (Algorithm 1) on a set of points on the Euclidean plane. This short procedure adds edges one at a time to the spanner it constructs, in ascending order by length. For each pair of vertices, in this order, it checks whether that pair already satisfies the bounded stretch inequality using the edges already added. If not, it adds a new edge connecting the pair. Therefore, by construction, each pair of vertices satisfies the inequality, either through previous edges or (if not) through the newly added edge. The resulting graph is therefore a t-spanner. Examples of the results of this algorithm, for three different stretch factors, are shown in Figure 2.1.

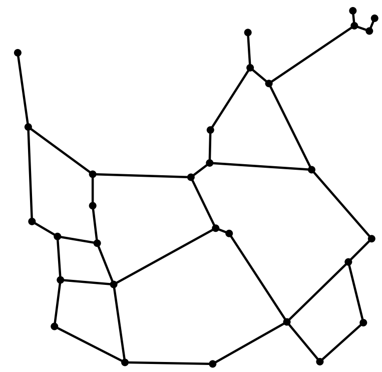
Algorithm 1 The naive greedy spanner algorithm.

```
procedure NAIVE-GREEDY(V)
    Let S be a graph with vertices V and edges E = {}
    for each pair (P, Q) ∈ V^2 in increasing order of d(P, Q) do
        if d_S(P, Q) > t · d(P, Q) then
            Add edge PQ to E
    return S
```

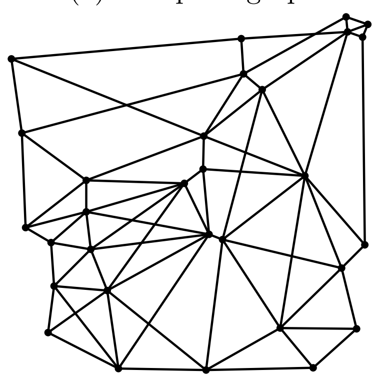
A naïve implementation of the greedy spanner algorithm runs in time $\mathcal{O}(n^3 \log n)$, where n is the number of given points [24]. Bose et al. [24] improved the running time of Algorithm 1 to near-quadratic time using a bounded version of Dijkstra's algorithm. Narasimhan et al. proposed an approximate version of the greedy spanner algorithm that reached a running time of $\mathcal{O}(n \log n)$, based on the use of approximate shortest path queries [32, 61, 80].



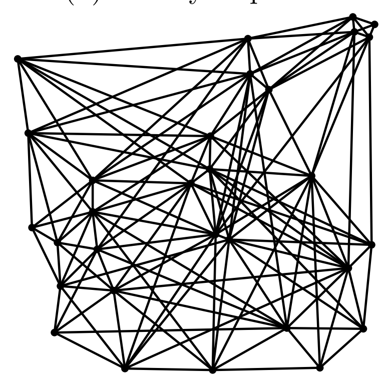
(a) Complete graph



(b) Greedy 2-spanner



(c) Greedy 1.2-spanner



(d) Greedy 1.05-spanner

Figure 2.1: A comparison of the complete graph on 30 random points on the plane with greedy spanners of parameters 2, 1.2, and 1.05 on the same point set.

Despite the simplicity of Algorithm 1, Farshi and Gudmundsson [43] observed that in practice, greedy spanners are surprisingly good in terms of the number of edges, weight, maximum vertex degree, and also the number of edge crossings. Many of these properties have been proven rigorously. Filster and Solomon [49] proved that greedy spanners have size and lightness that is optimal to within a constant factor for worst-case instances. They also achieved a near-optimality result for greedy spanners in spaces of bounded doubling dimension. Borradaile, Le, and Wulff-Nilsen [23] recently proved optimality for doubling metrics, generalizing a result of Narasimhan and Smid [80], and resolving an open question posed by Gottlieb [56], and Le and Solomon showed that no geometric t-spanner can do asymptotically better than the greedy spanner in terms of number of edges and lightness [73]. Finally, Eppstein and Khodabandeh [40, 41] proved a linear bound on the number of edge crossings of the greedy spanner in the two dimensional Euclidean plane. This in turn implies the existence of sublinear separators and separator hierarchies for greedy spanners of point sets in the two dimensional Euclidean plane. Therefore, sublinear separators and separator hierarchies can be used to implement efficient recursive algorithms on these subgraphs [38, 54, 34]. The existence of sublinear separators were later generalized to higher dimensions by Le and Cuong [76].

2.2 Dynamic & Streaming Algorithms for Graph Spanners

A t-spanner in a graph $G = (V, E)$ is subgraph $H = (V, E')$ such that $\delta_H(u, v) \leq t \cdot \delta_G(u, v)$ for all pairs of vertices $u, v \in V$. That is, the stretch t is the maximum distortion between the graph distances δ_G and δ_H. Importantly, when G changes (under edge/vertex insertions or deletions), the underlying metric δ_G changes, as well. The distance $\delta_G(u, v)$ may dramatically decrease upon the insertion of the edge uv. In contrast, our model assumes that the distances in the underlying metric space $\mathcal{M} = (P, \delta)$ remain fixed, but the algorithm can only see the distances between the points that have been presented. For this reason, our results are not directly comparable to models where the underling graph changes dynamically.

For *unweighted* graphs with n vertices, the current best fully dynamic and single-pass streaming algorithms can maintain spanners that achieve almost the same stretch-sparsity trade-off available for the static case: $2k - 1$ stretch and $O(n^{1+\frac{1}{k}})$ edges, for $k \geq 1$, which is attained by the greedy algorithm [4], and conjectured to be optimal due to the Erdős girth conjecture [42]. In the dynamic model, the objective is design algorithms and data structures that minimize the worst-case update time needed to maintain a t-spanner for S over all steps, regardless of its weight, sparsity, or lightness. See [8, 11, 13, 19] for some excellent work on dynamic spanners. In the streaming model the input is a sequence (or stream) of edges representing the edge set E of the graph G. A (single-pass) streaming algorithm decides, for each newly arriving edge, whether to include it in the spanner. The graph G is too large to fit in memory, and the objective is to optimize work space and update time [7, 9, 35, 44, 47, 78].

2.3 Incremental Algorithms for Geometric Spanners

We briefly review three previously known incremental $(1+\varepsilon)$-spanner algorithms in Euclidean d-space from the perspective of competitive analysis.

2.3.1 Deformable Spanners

Gao et al. [52] designed a dynamic DefSpanner algorithm that maintains a $(1+\varepsilon)$-spanner for a dynamic set S in Euclidean d-space. For point insertions, it only adds new edges, so it is an online algorithm, as well. It maintains a $(1+\varepsilon)$-spanner with $O_d(\varepsilon^{-d}) \cdot n$ edges and $O_d(\varepsilon^{-(d+1)} \log n)$ lightness. Since the $\|\mathrm{MST}(S)|$ is a lower bound for the optimal spanner weight, its competitive ratio is also $O_d(\varepsilon^{-(d+1)} \log n)$. The key ingredient of DefSpanner is *hierarchical nets* [67, 72, 84], a form of hierarchical clustering, which can be maintained dynamically. Hierarchical nets naturally generalize to doubling spaces, and so DefSpanner also maintains a $(1+\varepsilon)$-spanner with $\varepsilon^{-O(d)} \cdot n$ edges and lightness $\varepsilon^{-O(d)}$ in for doubling dimension d [55, 84].

2.3.2 Well-Separated Pair Decomposition (WSPD)

Well-separated pair decomposition was introduced by Callahan and Kosaraju [27] (see also [60, 66, 80, 88]). For a set S in a metric space, a WSPD is a collection of unordered pairs

$W = \{\{A_i, B_i\} : i \in I\}$ such that (1) $A_i, B_i \subset S$ for all $i \in I$; (2) $\min\{\|ab\| : a \in A_i, b \in B_i\} \le \varrho \cdot \max\{\mathrm{diam}(A_i), \mathrm{diam}(B_i)\}$ for all $i \in I$, where ϱ is the *separation ratio*; (3) for each point pair $\{a, b\} \subset S$ there exists a pair $\{A_i, B_i\}$ such that A_i and B_i each contain one of a and b. Given a WSPD with separation ratio $\varrho > 4$, any graph that contains at least one edge between A_i and B_i, for all $i \in I$, is a spanner with stretch $t = 1 + 8/(\varrho - 4)$. Setting $\varrho \ge 12\varepsilon^{-1}$ for $0 < \varepsilon < 1$, we obtain $t \le 1 + \varepsilon$.

Hierarchical clustering provides a WSPD [66, Ch. 3]. Perhaps the simplest hierarchical subdivisions in $\mathbb{R}^d$ are quadtrees. Let $\mathcal{T}$ be a quadtree for a finite set $S \subset \mathbb{R}^d$. The root of $\mathcal{T}$ is an axis-aligned cube of side length a_0, which contains S; it is recursively subdivided into 2^d congruent cubes until each leaf cube contains at most one point in S. For all pairs of cubes $\{Q_1, Q_2\}$ at level ℓ of $\mathcal{T}$, create a pair $\{A_i, B_i\}$ with $A_i = Q_1 \cap S$ and $B_i = Q_2 \cap S$ whenever $D_\ell \le \mathrm{dist}(Q_1, Q_b) < 2D_\ell$ for $D_\ell = \varrho \cdot \mathrm{diam}(Q_1) = 12\varepsilon^{-1} \cdot \sqrt{d} \cdot a_0/2^\ell$; and repeat for all levels $\ell \ge 0$. Properties (1)–(3) of a WSPD are easily verified [66, Ch. 3]. The resulting $(1 + \varepsilon)$-spanner has $O_d(\varepsilon^{-d}) \cdot n$ edges [66, 67] and lightness $O_d(\varepsilon^{-(d+1)} \log n)$ [17].

For point insertions in $\mathbb{R}^d$, a dynamic quadtree only adds nodes, which in turn creates new pairs in the WSPD, and new edges in the spanner. This is an online algorithm with the same guarantees as DefSpanner [17, 67] (see also [51] for an efficient implementation).

2.3.3 Ordered Yao-Graphs and Θ-Graphs

One of the first constructions for (offline) sparse $(1 + \varepsilon)$-spanner in Euclidean d-space were the Yao- and Θ-graphs [31, 71, 85]. Incremental versions of Yao-graphs and Θ-graphs were introduced by Bose et al. [26]. Let $S = \{s_1, \ldots, s_n\}$ be an ordered set of points in $\mathbb{R}^2$. For each $s_i \in S$, partition the plane into k cones with apex s and aperture $2\pi/k$. The *ordered Yao-graph* $Y_k(S)$ contains an edge between s_i and a closest *previous* point in $\{s_j : j < i\}$ in each cone. The graph $\Theta_k(S)$ is defined similarly, but in each cone the distance to the apex is measured by the orthogonal projection to a ray within the cone. Bose et al. [26] showed that the ordered Yao- and Θ-graphs have spanning ratio at most $1/(1 - 2\sin(\pi/k))$ for $k > 8$; tighter bounds were later obtained in [25]. In particular, the ordered Yao- and Θ-graphs are $(1 + \varepsilon)$-spanners for $k \geq \Omega(\varepsilon^{-1})$.

The construction generalizes to $\mathbb{R}^d$ for all $d \in \mathbb{N}$ [85]. For an angle $\alpha \in (0, \pi)$, let $A \subset \mathbb{S}^{d-1}$ be a maximal set of points in the $(d-1)$-sphere such that $\min_{a,b \in A} \mathrm{dist}(a, b) \leq \alpha$ (in radians). A standard volume argument shows that $|A| \leq O_d(\alpha^{1-d})$. For each $a_i \in A$, create a cone C_i with apex at the origin o, aperture α, and symmetry axis oa_i. Note that $\mathbb{R}^d \subseteq \bigcup_i C_i$. Given a finite set $P \subset \mathbb{R}^d$, we translate each cone C_i to a cone $C_i(p)$ with apex $p \in P$. For every cone $C_i(p)$, the Yao-graph contains an edge between p and a closest point in $P \cap C_i(p)$. For every $\varepsilon > 0$ and $d \in \mathbb{N}$, there exists an angle $\alpha = \alpha(d, \varepsilon) = \Theta_d(\varepsilon)$ for which the Yao-graph is a $(1 + \varepsilon)$-spanner for every finite set $P \subset \mathbb{R}^d$.

Ordered Yao- and Θ-graphs give online algorithms for maintaining a $(1 + \varepsilon)$-spanner for a sequence of points in $\mathbb{R}^d$. The sparsity of these spanners is bounded by the number of cones per vertex, $O_d(\varepsilon^{1-d})$, which matches the (offline) lower bound of $\Omega_d(\varepsilon^{1-d})$ [73]. However, their weight may be significantly higher than optimal: For n equally spaced points in a unit circle, in any order, Yao- and Θ-graphs yield $(1 + \varepsilon)$-spanners of weight $\Omega(\varepsilon^{-1}n)$, hence lightness $\Omega(\varepsilon^{-1}n)$, while the optimum weight is $O(\varepsilon^{-2})$ [73].

2.3.4 Online Steiner Spanners

An important variant of online spanners is when it is allowed to use auxiliary points (Steiner points) which are not part of the input sequence of points, but are present in the metric space. An online algorithm is allowed *add* Steiner points, however, the spanner must achieve the given stretch factor only for the input point pairs. It has been observed through a series of work in recent years, that Steiner points allow for substantial improvements over the bounds on the sparsity and lightness of Euclidean spanners in the offline settings and highly nontrivial insights are required to argue the bounds for Steiner spanners, and often they tend to be even more intricate than their non-Steiner counterpart; see [15, 16, 73, 74]. Bhore and Tóth [17] showed that if an algorithm can use Steiner points, then the competitive ratio for weight improves to $O(\varepsilon^{(1-d)/2}\log n)$ in the Euclidean d-space.

Chapter 3

Euclidean metrics

We present an online algorithm for a sequence of points in Euclidean d-space (Section 3.1). It combines features from several previous approaches, and maintains a $(1+\varepsilon)$-spanner of lightness $O_d(\varepsilon^{-d}\log n)$ and sparsity $O_d(\varepsilon^{1-d}\log\varepsilon^{-1})$ for $d\geq 1$. Lightness is an upper bound for the competitive ratio for weight; the sparsity almost matching the optimal bound $O_d(\varepsilon^{1-d})$ attained by ordered Yao-graphs. In the plane ($d=2$), we show that the same algorithm achieves competitive ratio $O(\varepsilon^{-3/2}\log\varepsilon^{-1}\log n)$ using a tighter analysis: A charging scheme that charges the weight of the online spanner to a minimum weight spanner (Section 3.2).

3.1 An Improvement in All Dimensions

We combine features from two incremental algorithms for geometric spanners, and obtain an online $(1+\varepsilon)$-spanner algorithm for a sequence of n points in $\mathbb{R}^d$. We maintain a dynamic quadtree for hierarchical clustering, and use a modified ordered Yao-graph in each level of the hierarchy. In particular, we limit the weight of the edges in the Yao-graph in each level of the hierarchy (thereby avoiding heavy edges). We start with an easy observation.

Lemma 3.1. *Let $G = (S, E)$ be a t-spanner and let $w > 0$. Let $G' = (S, E')$, where $E' = \{e \in E : \|e\| \leq w\}$ is the set of edges of weight at most w. Then for every $a, b \in S$ with $\|ab\| < w/t$, graph G' contains an ab-path of weights at most $t\,\|ab\|$.*

Proof. Since G is a t-spanner, it contains an ab-path P_{ab} of weight at most $t\,\|ab\| \leq w$. By the triangle inequality, every edge in this path has weight at most w, hence present in G'. Consequently G' contains P_{ab}. □

3.1.1 Online Algorithm ALG_1

The input is a sequence of points $(s_1, s_2, \ldots)$ in $\mathbb{R}^d$, $d \geq 1$. The set of the first n points is denoted by $S_n = \{s_i : 1 \leq i \leq n\}$. For every n, we dynamically maintain a quadtree $\mathcal{T}_n$ for S_n. Every node of $\mathcal{T}_n$ corresponds to a cube. The root of $\mathcal{T}_n$, at level 0, corresponds to a cube Q_0 of side length $a_0 = \Theta(\mathrm{diam}(S_n))$. At every level $\ell \geq 0$, there are at most $2^{d\ell}$ interior-disjoint cubes, each of side length $a_\ell = a_0\,2^{-\ell}$. A cube $Q \in \mathcal{T}_n$ is *nonempty* if $Q \cap S_n \neq \emptyset$. For every nonempty cube Q, we maintain a representative $s(Q) \in Q \cap S_n$, selected at the time when Q becomes nonempty. At each level ℓ, let P_ℓ be the sequence of representatives, in the order in which they are created.

For each level ℓ, we maintain a modified ordered Yao-graph $G_\ell = (P_\ell, E_\ell)$ as follows. When a new point p is inserted into P_ℓ, cover $\mathbb{R}^d$ with $\Theta_d(\varepsilon^{1-d})$ cones of aperture $\alpha(d, \varepsilon)$ as in the construction of Yao-graphs. In each cone C_i, find a point $q_i \in C_i \cap P_\ell$ closest to p; and add pq_i to E_ℓ if $\|pq_i\| < 24a_\ell\sqrt{d} \cdot \varepsilon^{-1}$. The algorithm maintains the spanner $G = \bigcup_{\ell \geq 0} G_\ell$.

3.1.2 Analysis

Theorem 3.2. *Let $d \geq 1$ and $\varepsilon \in (0,1)$. The online algorithm* ALG$_1$ *maintains, for a sequence of n points in Euclidean d-space, an $(1+O(\varepsilon))$-spanner with weight $O_d(\varepsilon^{-d}\log n)\cdot \|MST\|$ and $O_d(\varepsilon^{1-d}\log\varepsilon^{-1})\cdot n$ edges.*

Note that Theorem 3.2 implies that the competitive ratio of this algorithm is also $O_d(\varepsilon^{-d}\log n)$.

Proof. **Stretch Analysis.** We give a bound on the stretch factor in two steps: First, we define an auxiliary graph $H = (S, E')$ which is a $(1+\varepsilon)$-spanner for S by the analysis of WSPDs. Then we show that G contains an ab-path of weight at most $(1+\varepsilon)\|ab\|$ for each edge of H. Overall, the stretch of G is at most $(1+\varepsilon)^2 = (1+O(\varepsilon))$ for all $a, b \in S$.

First Layer: WSPD. For each level $\ell \geq 0$, let $H_\ell = (P_\ell, E'_\ell)$ be the graph that contains an edge between two representatives $a, b \in P_\ell$ whenever $\|ab\| \leq 12a_\ell\sqrt{d}\cdot\varepsilon^{-1}$. Let $H = \bigcup_{\ell\geq 0} H_\ell$. The auxiliary graph H_ℓ contains an edge between the representatives of any such pair of cubes at level ℓ. As noted, $H = \bigcup_{\ell\geq 0} H_\ell$ is a $(1+\varepsilon)$-spanner (cf. [66, 67]).

Second Layer: Near-Sighted Yao-graphs. As H is a $(1+\varepsilon)$-spanner, for every $a, b \in S_n$, it contains an ab-path of weight at most $(1+\varepsilon)\|ab\|$. Consider such a path $P_{ab} = (a = p_0, \ldots, p_m = b)$. Each edge $p_{i-1}p_i$ is in H_ℓ for some $\ell \geq 0$. By construction, every edge in H_ℓ has weight at most $12a_\ell\sqrt{d}\cdot\varepsilon^{-1}$. For every level ℓ, the ordered Yao-graph $Y(P_\ell)$ with angle $\alpha(d,\varepsilon)$ is a $(1+\varepsilon)$-spanner. The graph $G_\ell = (P_\ell, E_\ell)$ constructed by ALG$_1$ at level ℓ is a subgraph of $Y(P_\ell)$. By Lemma 3.1, for every $p, q \in P_\ell$ with $\|pq\| \leq 12a_\ell\sqrt{d}\cdot\varepsilon^{-1}$, graph G_ℓ contains a pq-path of weight at most $(1+\varepsilon)\|pq\|$.

Overall, H contains an ab-path $P_{ab} = (p_0, \ldots, p_m)$ of weight at most $(1+\varepsilon)\|ab\|$. For each edge $p_{i-1}p_i$ of P_{ab}, graph G contains a $p_{i-1}p_i$-path of weight $(1+\varepsilon)\|p_{i-1}p_i\|$. The concatenation of these paths is an ab-path of weight $(1+\varepsilon)^2\|ab\| \leq (1+O(\varepsilon))\|ab\|$.

Weight Analysis. We may assume, without loss of generality, that the root of the quadtree $\mathcal{T}_n$ is the unit cube $[0,1]^d \subset \mathbb{R}^d$, which has diameter $\sqrt{d}$. This implies $\text{diam}(S_n) \leq \sqrt{d} = O_d(1)$. Assume further that $n > 1$, and $\frac{1}{4} \leq \text{diam}(S_n) \leq \|MST(S_n)\|$.

Every edge in E_ℓ at level ℓ has weight $O_d(\varepsilon^{-1}\, 2^{-\ell})$. In particular, every edge at level $\ell \geq 2\log n$ has weight $O_d(\varepsilon^{-1}/n^2)$; and the total weight of these edges is $O_d(\varepsilon^{-1}) \leq O_d(\varepsilon^{-1}\|MST(S_n)\|)$.

It remains to bound the weight of the edges on levels $\ell = 1, \ldots, \lfloor 2\log n \rfloor$. At level ℓ of the quadtree $\mathcal{T}_n$, there are at most $2^{d\ell}$ nodes, hence $|P_\ell| \leq 2^{d\ell}$. If $|P_\ell| < 3^d$, then G_ℓ has at most $O(3^{2d}) = O_d(1)$ edges, each of weight at most $\text{diam}(P_\ell) \leq \text{diam}(S_n) \leq \|\text{MST}(S_n)\|$, and so $\|E_\ell\| \leq O_d(\|\text{MST}(S_n)\|)$. Assume now that $|G_\ell| \geq 3^d$. By the definition of ordered Yao-graphs, each vertex inserted into P_ℓ adds $\Theta(\varepsilon^{1-d})$ new edges, each of weight $O(\varepsilon^{-1}\, 2^{-\ell})$. The total weight of the edges in G_ℓ is at most

$$\|E_\ell\| \leq |P_\ell| \cdot \varepsilon^{1-d} \cdot \max_{e \in E_\ell} \|e\| \leq O_d(|P_\ell|\, \varepsilon^{-d}\, 2^{-\ell}). \tag{3.1}$$

We next derive a lower bound for $\|\text{MST}(S_n)\|$ in terms of $|P_\ell|$, when $|P_\ell| > 1$ and $\ell > 2$, using a standard volume argument. Define a graph on the vertex set P_ℓ such that two nodes $p, q \in P_\ell$ are adjacent iff p and q lie in neighboring quadtree cells of level ℓ. Since every quadtree cell has $3^d - 1$ neighbors, this graph is $(3^d - 1)$-degenerate, and contains an independent set I_ℓ of size at least $(3^d - 1)^{-1}|P_\ell| = \Omega_d(|P_\ell|)$. The distance between any two disjoint quadtreee cells at level ℓ is at least $2^{-\ell}$. Consequently, the open balls of radius $2^{-(\ell+1)}$ centered at the points in I_ℓ are pairwise disjoint. None of the balls contains S_n for $\ell > 2$, as the diameter of each of ball is $2^{-\ell}$ while $\text{diam}(S_n) \geq \frac{1}{4}$. For all $\ell > 2$, $\text{MST}(S_n)$ contains the center of each ball and a point in its exterior; hence the intersection of $\text{MST}(S_n)$ and each ball contains a path from the center to a boundary point, which has weight at least $2^{-(\ell+1)}$.

Summation over $|I_\ell|$ disjoint balls yields

$$\|MST(S_n)\| \geq |I_\ell| \cdot 2^{-(\ell+1)} \geq \Omega_d(|P_\ell|\, 2^{-\ell}). \tag{3.2}$$

Comparing inequalities (3.1) and (3.2), we obtain $\|E_\ell\| \leq O_d(\varepsilon^{-d}) \cdot \|MST(S_n)\|$. Summation over all levels $\ell \in \mathbb{N}$ yields $\|E\| \leq O_d(\varepsilon^{-d} \log n) \cdot \|\mathrm{MST}(S_n)\|$, as claimed.

Sparsity Analysis. We show that G has $O_d(\varepsilon^{1-d} \log \varepsilon^{-1}) \cdot n$ edges. Har-Peled proved that the auxiliary graph H is $O(\varepsilon^{-d})$-degenerate, and so it has $O_d(\varepsilon^{-d}) \cdot n$ edges [66, 67, Lemma 3.9]. As G is a subgraph of H, hence has $O_d(\varepsilon^{-d}) \cdot n$ edges as well. We improve this bound using a charging scheme.

For the quadtree $\mathcal{T}_n$ maintained by algorithm ALG$_1$, let $\mathcal{T}_n'$ denote the *compressed quadtree*, which is obtained from $\mathcal{T}_n$ by removing all leaves that correspond to empty cubes, and supressing nodes with a single child [10, 66]. For n points in $\mathbb{R}^d$, the compressed quadtree has $O_d(n)$ nodes (which are nodes of the original quadtree, as well). For each node Q of $\mathcal{T}_n'$, algorithm ALG$_1$ adds $O_d(\varepsilon^{1-d})$ edges between the representative $s(Q)$ and the closest points in each cone C_i (in P_ℓ, where $\ell \geq 0$ is the level of Q in $\mathcal{T}_n$). The total number of these edges for all nodes of $\mathcal{T}_n'$ is $O(\varepsilon^{1-d}) \cdot n$.

It remains to consider the nodes of the quadtree $\mathcal{T}_n$ that are compressed in $\mathcal{T}_n'$. Every compressed node is part of a descending chain of single-child nodes in $|mathcal T_n$. The number of such chains is $O_d(n)$, as each chain has a unique direct descendant in $\mathcal{T}_n'$. Let $Q_k, \ldots, Q_\ell$ be a maximal chain of single-child nodes in $\mathcal{T}_n$, where Q_j is on level j of $\mathcal{T}_n$ for $j = k, \ldots, \ell$. These are nested cubes $Q_k \subset Q_{k-1} \subset \ldots \subset Q_1$ with a common representative, $s = q(Q_k) = \ldots = s(Q_\ell)$; see Figure 3.1. Let C_i be a cones with apex s and aperture $\alpha(d, \varepsilon)$ in algorithm ALG$_1$; and let $q_{i,j}$ denote the closest point to s in $C_i \cap P_j$ for $j = k, \ldots, \ell$. If a point $q_{i,j} \in P_j$ represents some compressed cube Q' (in another compressed chain), then $q_{i,j}$ represents the parent of Q', as well. In this case, $q_{i,j} \in P_{\ell-1}$, which implies $q_{i,j} = q_{i,j-1}$.

Consequently, $q_{i,j} = q_{i,j-1} = \ldots = q_{i,k}$. We may assume that only $q_{i,k}$ represents a compressed node.

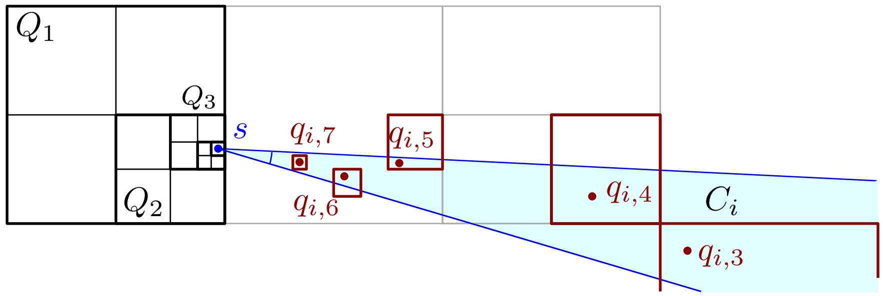


Figure 3.1: A point s is the representative of five nested squares in the quadtree. The closest point to s is $q_{i,\ell} \in C_i \cap P_\ell$ in the cone C_i at level $\ell = 3, \ldots, 7$.

The first $\leq \lceil \log \varepsilon^{-1} \rceil$ nodes (i.e., Q_j for $k \leq +\lceil \log \varepsilon^{-1} \rceil$) jointly contribute $O(\varepsilon^{1-d} \log \varepsilon^{-1})$ edges to G. Summation over all compressed chains yields $O(\varepsilon^{-1} \log \varepsilon^{-1}) \cdot n$ edges. For the remaining nodes in the chain (that is, nodes Q_j for $k < j \leq \ell - \log \varepsilon^{-1}$), we use thefollowing charging scheme: Charge the edge $sq_{i,j}$ to $q_{i,j}$. Since $j \neq k$, then $q_{i,j}$ represents a noncompressed node at level j of $\mathcal{T}_n$. Next we bound the charges received by $q_{i,j}$.

We claim that for every noncompressed node Q, the representative $q = s(Q)$ receives at most $O_d(1)$ units of charges. Indeed, suppose that $q \in P_\ell$ and an edge sq has been charged to q. Then $\|sq\| \leq 24 a_\ell \sqrt{d} \cdot \varepsilon^{-1}$. However, s is the only point in the cube $Q'_s := Q_{j-\lceil \log \varepsilon^{-1} \rceil}$ of side length $a_\ell \cdot 2^{\lceil \log \varepsilon^{-1} \rceil} \geq a_\ell \cdot \varepsilon^{-1}$ and $\mathrm{diam}(Q'_s) \geq a_\ell \sqrt{d} \cdot \varepsilon^{-1}$. Consequently, Q'_s lies in the ball B_q of radius $25 a_\ell \sqrt{d} \cdot \varepsilon^{-1}$ centered at q. However, comparing the volumes of B_q and Q'_s shows that B_q contains $O_d(1)$ interior-disjoint cubes Q'_s, and so q is charged at most $O_d(1)$ times. Summation over all all $O_d(n)$ noncompressed nodes over all levels $\ell > 0$ shows that the total number of edges that participate in the charging scheme is $O_d(n)$.

Overall, we have shown that G has at most $O(\varepsilon^{1-d} \log \varepsilon^{-1}) \cdot n$ edges. $\square$

3.2 Further Improvements in the Plane

We presents a tighter analysis of algorithm ALG_1 for $d = 2$ that compares the spanner weight to the offline optimum weight, and bypasses the comparison with the MST (i.e., lightness).

3.2.1 Minimum-Weight Euclidean $(1+\varepsilon)$-Spanner

For any $a, b \in \mathbb{R}^d$, an ab-path P_{ab} of Euclidean weight at most $(1+\varepsilon)\|ab\|$ lies in the ellipsoid $\mathcal{E}_{ab}$ with foci a and b and great axis of weight $(1+\varepsilon)\|ab\|$; see Figure 3.2. A key observation is that the minor axis of $\mathcal{E}_{ab}$ is $((1+\varepsilon)^2 - 1^2)^{1/2}\,\|ab\| \approx \sqrt{2\varepsilon}\,\|ab\|$. Furthermore, Bhore and Tóth [16] recently observed that the directions of "most" edges of the path P_{ab} are "close" to the direction of ab. Specifically, if we denote by $E(\alpha)$ the set of edges e in P_{ab} with $\angle(ab, e) \le \alpha$, then the following holds.

Lemma 3.3 (Bhore and Tóth [16])**.** *Let $a, b \in \mathbb{R}^d$ and let P_{ab} be an ab-path of weight $\|P_{ab}\| \le (1+\varepsilon)\|ab\|$. Then for every $i \in \{1, \ldots, \lfloor 1/\sqrt{\varepsilon} \rfloor\}$, we have $\|E(i \cdot \sqrt{\varepsilon})\| \ge (1 - 2/i^2)\,\|ab\|$.*

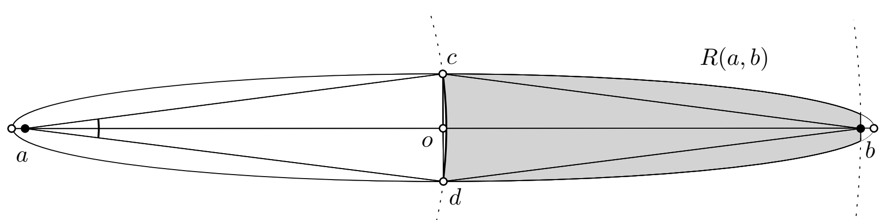


Figure 3.2: Any ab-path of weight at most $(1+\varepsilon)\|ab\|$ lies in the ellipse $\mathcal{E}_{ab}$ with foci a and b. The shaded region $R(a, b)$ is the part of the ellipse $\mathcal{E}_{ab}$ between two concentric circles centered at a.

Let $R(a, b) = \mathcal{E}_{ab} \cap \mathcal{N}(a, b)$, where $\mathcal{N}(a, b)$ is the annulus bounded by two concentric spheres centered at a, of radii $\frac{1+\varepsilon}{2}\,\|ab\|$ and $\|ab\|$; see Figure 3.2 for an example.

Lemma 3.4. *If $0 < \varepsilon < \frac{1}{9}$, then every ab-path P_{ab} of weight at most $\|P_{ab}\| \le (1+\varepsilon)\|ab\|$ contains interior-disjoint line segments $s \subset R(a, b)$ of total weight at least $\frac{1}{3}\,\|ab\|$ such that $\angle(\overrightarrow{ab}, s) \le 3 \cdot \sqrt{\varepsilon}$.*

Proof. Since the distance between the two concentric circles is $\frac{1-\varepsilon}{2}\|ab\|$, every ab-path contains a subpath of weight at least $\frac{1-\varepsilon}{2}\|ab\|$ in the ans $\mathcal{N}(a,b)$.

Let P_{ab} be an ab-path of weight at most $(1+\varepsilon)\|ab\|$. As noted above $P_{ab} \subset \mathcal{E}_{ab}$. Hence, $\|P_{ab} \cap \mathcal{N}(a,b)\| = \|P_{ab} \cap R(a,b)\| \geq \frac{1-\varepsilon}{2}\|ab\|$ in $R(ab)$; and so $\|P_{ab} \setminus R(a,b)\| = \|P_{ab}\| - \|P_{ab} \cap R(a,b)\| \leq \frac{1+3\varepsilon}{2}\|ab\|$.

Applying Lemma 3.3 with $i=3$, the total weight of the edges e of P_{ab} with $\mathrm{dir}(ab,e) \leq 3\cdot\sqrt{\varepsilon}$ is at least $\frac{7}{9}\|ab\|$. The parts of these edges lying outside of $R(a,b)$ have weight at most $\|P_{ab} \setminus R(a,b)\| \leq \frac{1+3\varepsilon}{2}\|ab\|$. Consequently, the remaining part of these edges are in $R(a,b)$, and their weight is at least $\left(\frac{7}{9} - \frac{1+3\varepsilon}{2}\right)\|ab\| \leq \frac{7-27\varepsilon}{18}\|ab\| \leq \frac{2}{9}\|ab\|$ if $\varepsilon < \frac{1}{9}$, as claimed □

We also need an observation from elementary geometry; see Figure 3.2.

Lemma 3.5. *For $a,b \in \mathbb{R}^d$, let cd be the minor axis of the ellipsoid $\mathcal{E}_{ab}$. Then $\angle cad \leq 2\,\varepsilon^{1/2}$.*

Proof. We may assume, without loss of generality, that $\|ab\| = 1$. Let o be the center of the ellipsoid $\mathcal{E}_{ab}$. Then $\sec\angle cao = (\cos\angle cao)^{-1} = \frac{\|ac\|}{\|ao\|} = 1+\varepsilon$. From the Taylor estimate $\sec(x) = 1 + \frac{1}{2}x^2 + \frac{5}{24}x^4 + \ldots \leq 1 + x^2$ for $0 < x < 1$, we have $\angle cao \geq \varepsilon^{1/2}$. Consequently, $\angle cad = 2\,\angle cao \geq 2\varepsilon^{1/2}$. □

Theorem 3.6. *Let $d=2$ and $\varepsilon \in (0,1)$. The online algorithm* ALG_1 *maintains, for a sequence of n points in Euclidean plane, an $(1+\varepsilon)$-spanner of weight $O(\varepsilon^{-3/2}\log\varepsilon^{-1}\log n)\cdot$* OPT, *where* OPT *denotes the minimum weight of an $(1+\varepsilon)$-spanner for the same point set.*

Proof. Theorem 3.2 has established that algorithm ALG_1 maintains a $(1+\varepsilon)$-spanner. The tighter competitive analysis uses Lemmas 3.4 and 3.5.

3.2.2 Competitive Analysis

Assume, without loss of generality, that $\mathrm{diam}(S_n) = \Theta(1)$, hence the side length of every quadtree square at level ℓ is $\Theta(2^{-\ell})$. For a set $S_n = \{s_1, \ldots, s_n\} \subset \mathbb{R}^2$, let $G^* = (S_n, E^*)$ be a $(1+\varepsilon)$-spanner of minimum weight, and let $\mathrm{OPT} = \|G^*\|$. Let $G = (S_n, E)$ be the spanner returned by the online algorithm ALG_1. Recall that $G = \bigcup_{\ell \geq 0} G_\ell$, where the total weight of all edges at levels $\ell > 2 \log n$ is less than $\mathrm{diam}(S_n)$, so it is enough to consider $\ell = 0, \ldots, \lceil 2 \log n \rceil$.

Claim 3.7. $\|G_\ell\| \leq O(\varepsilon^{-3/2} \log \varepsilon^{-1}) \cdot \mathrm{OPT}$ *for all* $\ell \geq 0$.

Claim 3.7 immediately implies $\|G\| \leq O(\varepsilon^{-3/2} \log \varepsilon^{-1} \log n) \cdot \mathrm{OPT}$. For every level $\ell \geq 0$, $G_\ell = (P_\ell, E_\ell)$ is a graph on the representatives P_ℓ. Note that G^* is a Steiner spanner with respect to the point set P_ℓ, as G^* is a spanner on all n points of the input.

We prove Claim 3.7 using a charging scheme: We charge the weight of every edge in G_ℓ to G^* (more precisely, to line segments along the edges of G^*), and then show that each line segment of weight w in G^* receives $O(\varepsilon^{-3/2} \log \varepsilon^{-1}) \cdot w$ charge.

For every point $p \in P_\ell$, algorithm ALG_1 greedily covers $\mathbb{R}^2$ by $\Theta(\varepsilon^{-1})$ cones of aperture $\pi/k = \Theta(\varepsilon^{-1})$ and apex p, and adds an edge pq_i in each nonempty cone C_i. For the competitive analysis, we greedily cover $\mathbb{R}^2$ by $\Theta(\varepsilon^{-1/2})$ cones of aperture $\sqrt{\varepsilon}$ and apex p. We use translates of the same cone cover for all $p \in P_\ell$. Standard volume argument implies that a cone of aperture $\sqrt{\varepsilon}$ intersects $O(\varepsilon^{-1/2})$ cones of aperture $\Theta(\varepsilon^{-1})$. We describe the charging scheme for each such cone $\widehat{C}$.

Charging Scheme

Consider a cone $\widehat{C}$ with apex p and aperture $\sqrt{\varepsilon}$. Let $E(\widehat{C})$ be the set of edges pq, $q \in \widehat{C}$ that algorithm ALG_1 adds to G_ℓ when p is inserted into P_ℓ. Since $\widehat{C}$ intersects $O(\varepsilon^{-1/2})$ cones of the ordered Yao-graph, then $|E(\widehat{C})| \leq O(\varepsilon^{-1/2})$. By construction, every edge in G_ℓ has weight at most $O(\varepsilon^{-1}2^{-\ell})$.

$$\|E(\widehat{C})\| = \sum_{pq \in E(\widehat{C})} \|pq\| \leq |E(\widehat{C})| \cdot O(\varepsilon^{-1}2^{-\ell}) \leq O(\varepsilon^{-3/2}2^{-\ell}). \tag{3.3}$$

Let $q_0 = q_0(\widehat{C})$ be a closest point in $P_\ell \cap \widehat{C}$ to p. (Possibly, q_0 arrived after p.) We distinguish between two cases:

Case 1: $\|pq_0\| < 2 \cdot 2^{-\ell}$**.** Since $q_0 \in P_\ell$, /nd P_ℓ contains at most one point in each quadtree cell of side length $\Theta(2^{-\ell})$, this case occurs for at most $O(1)$ times per apex p. On the one hand, the sum of weights over all $p \in P_\ell$ and all cones $\widehat{C}$ with $\|pq_0\| < 2 \cdot 2^{-\ell}$ is bounded by $O(|P_\ell| \cdot \varepsilon^{-3/2}2^{-\ell})$. On the other hand, $\mathrm{OPT} \geq \Omega(\|\mathrm{MST}(P_\ell)\|) \geq \Omega(|P_\ell| \cdot 2^{-\ell})$. Consequently, the total weight of all edges handled in Case 1 is $O(\varepsilon^{-3/2})\,\mathrm{OPT}$.

Case 2: $\|pq_0\| \geq 2 \cdot 2^{-\ell}$**.** The optimal spanner G^* contains a pq_0-path P_0 of weight at most $(1+\varepsilon)\|pq_0\|$. Recall P_0 lies in the ellipse $\mathcal{E}_0$ with foci p and q_0, and $R(p, q_0)$ is the half of $\mathcal{E}_0$ that contains q_0 (cf. Figure 3.2). Let $E^*(\widehat{C})$ be the set of maximal line segments e along edges in E^* such that $e \subset P_0 \cap R(p, q_0)$ and $\angle(e, pq_0) \leq 3 \cdot \sqrt{\varepsilon}$. By Lemma 3.4, we have $\|E^*(\widehat{C})\| \geq \frac{1}{3}\|pq_0\|$. We distribute the weight of all edges in $E(\widehat{C})$ uniformly among the line segments in $E^*(\widehat{C})$. That is, each segment of weight w in $E^*(\widehat{C})$ receives a charge of

$$\frac{\|E(\widehat{C})\|}{\|E^*(\widehat{C})\|} \cdot w \leq \frac{O(\varepsilon^{-3/2}2^{-\ell})}{\Omega(2^{-\ell})} \cdot w \leq O(\varepsilon^{-3/2}) \cdot w. \tag{3.4}$$

This completes the description of the charging scheme in Case 2.

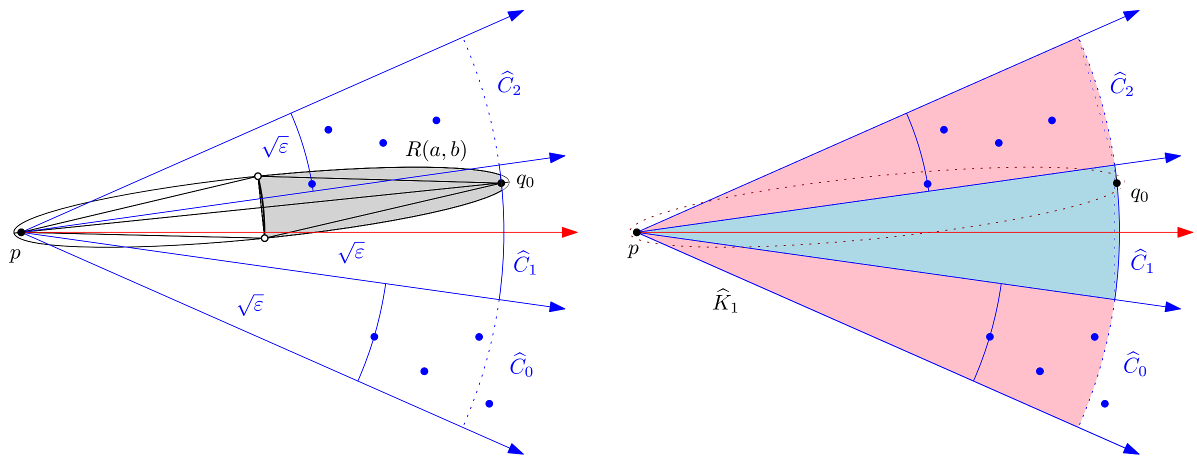


Figure 3.3: Left: There consecutive cones, $\widehat{C}_0$, $\widehat{C}_1$, and $\widehat{C}_1$, with apex p and aperture $\sqrt{\varepsilon}$. Point q_0 is the closest to p in $P_\ell \cap \widehat{C}_1$; and $R(p, q_0) \subset \widehat{K}_1 = \widehat{C}_0 \cup \widehat{C}_1 \cup \widehat{C}_2$. Right: No point in P_ℓ is in the blue sector $\widehat{K}$, but there may be points in the pink sectors.

Charges Received

A point along an edge of the optimal spanner G^* may receive charges from several cones $\widehat{C}$, possibly with different apices $p \in P_\ell$. Let L be a maximal line segment along an edge of G^* such that every point in L receives the same charges.

For a cone $\widehat{C}$ of aperture $\sqrt{\varepsilon}$, let $\widehat{K}$ denote a cone with the same apex and axis as $\widehat{C}$, but aperture $3\sqrt{\varepsilon}$; refer to Figure 3.3.

Claim 3.8. *If L receives charges from $\widehat{C}$, then $L \subset \widehat{K}$.*

Indeed, if L receive charges from $\widehat{C}$, then $L \subset R(p, q_0) \subset E_0$, where $\mathcal{E}_0$ is the ellipse with foci p and the closest point $q_0 \in \widehat{C} \cap P_\ell$. By Lemma 3.5, $R(p, q_0)$ lies in a cone with apex p, aperture $2\sqrt{\varepsilon}$, and axis pq_0. Consequently $L \subset R(p, q_0) \subset \widehat{K}$, which proves Claim 3.8.

Note that if L receives positive charge from a cone $\widehat{C}$ with apex p and closest point q_0, then $\angle(L, pq_0) \leq 3 \cdot \sqrt{\varepsilon}$. Since the aperture of the cones $\widehat{C}$ is $\sqrt{\varepsilon}$, then L receives charges from cones $\widehat{C}$ with at most $O(1)$ different orientations. We may restrict ourselves to cones $\widehat{C}$ that are translates of each other (but have different apices in P_ℓ).

Let $\mathcal{A}$ be the set of all translates of a cone $\widehat{C}$ with aperture $\sqrt{\varepsilon}$ and apices in P_ℓ, and L receives positive charge from $\widehat{C}$. We partition $\mathcal{A}$ into $O(\log \varepsilon^{-1})$ classes as follows. For $j = 1, \ldots, \lceil \log(2\varepsilon^{-1}) \rceil$, let $\mathcal{A}_j$ be the set of cones $\widehat{C} \in \mathcal{A}$ such that $2^{j-\ell} \leq \|pq_0\| < 2^{j+1-\ell}$, where $p \in P_\ell$ is the apex of $\widehat{C}$ and q_0 is the closest point in $P_\ell \cap \widehat{C}$ to p.

Claim 3.9. *For each j, segment L receives $O(\varepsilon^{-3/2})\,\|L\|$ total charges from all cones in $\mathcal{A}_j$.*

By refining (3.4) for a cone in $\widehat{C} \in \mathcal{A}_j$, we see that L receives a charge

$$\frac{\|E(\widehat{C})\|}{\|E^*(\widehat{C})\|} \cdot \|L\| \leq \frac{O(\varepsilon^{-3/2}2^{j-\ell})}{\Omega(2^{-\ell})} \cdot \|L\| \leq O(\varepsilon^{-3/2}2^{-j}) \cdot \|L\| \tag{3.5}$$

from each cone in $\mathcal{A}_j$. To prove Claim 3.9, it is enough to show that $|\mathcal{A}_j| \leq O(2^j)$.

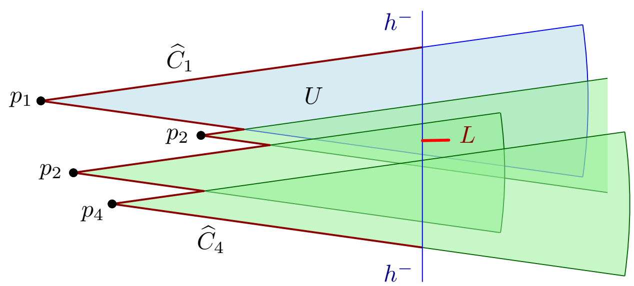


Figure 3.4: The union U of triangles $\widehat{C} \cap h^-$, where L receives charges from the cones $\widehat{C}$.

We may assume, without loss of generality, that the symmetry axis of every cone in $\mathcal{A}_j$ is parallel to the x-axis, and their apex is their leftmost point. Let h be a vertical line that contains the left endpoint of L, and let h^- be the left halfplane bounded by h; see Figure 3.4. The intersections $\widehat{C} \cap h$ and $\widehat{K} \cap h$ are vertical line segment of length $O(2^{j-\ell} \tan \sqrt{\varepsilon})$. We have $L \cap h \subset \widehat{K} \cap h$ by Claim 3.8; and obviously $\widehat{C} \cap h \subset \widehat{K} \cap h$. Consequently, a vertical line segment of length $O(2^{j-\ell} \tan \sqrt{\varepsilon})$ contains $h \cap \widehat{C}$ for all $\widehat{C} \in \mathcal{A}_j$.

Let U be the union of the triangles $\widehat{C} \cap h^-$ for all $\widehat{C} \in \mathcal{A}_j$. The interior of the $\widehat{C} \cap h^-$ does not contain any point in P_ℓ. Consequently, the apices of all cones lie on the boundary ∂U of U. The part of ∂U in h^- is a y-monotone curve with slopes $\pm\sqrt{\varepsilon}$. It follows that the length of ∂U is $O(2^{j-\ell} \tan \sqrt{\varepsilon} / \sin \sqrt{\varepsilon}) = O(2^{j-\ell} \csc \sqrt{\varepsilon}) = O(2^{j-\ell})$. This, in turn, implies that ∂U intersects $O(2^j)$ cubes of side length $a_0 2^{-\ell}$ at level ℓ of the quadtree, and so $|\mathcal{A}_j| \leq O(2^j)$, as required. This completes the proof Claim 3.9, and hence the proof of Theorem 3.6. $\square$

3.3 Lower Bounds in $\mathbb{R}^d$ Under the L_1 Norm

In this section we introduce a strategy based on the points on the integer lattice $\mathbb{Z}^d$, that achieves a new lower bound for the competitive ratio of an online $(1+\varepsilon)$-spanner algorithm in $\mathbb{R}^d$ under the L_1 norm.

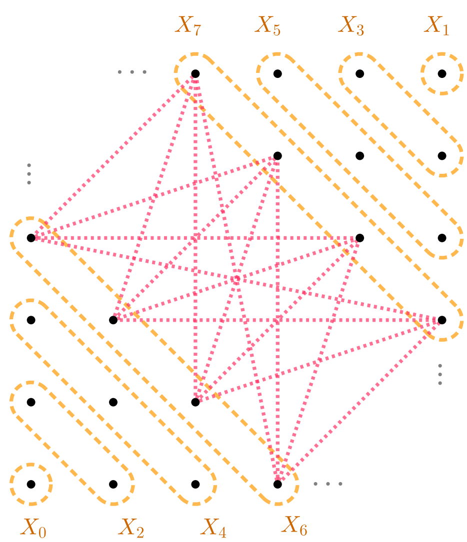


Figure 3.5: A sketch of the construction for the lower bound in two dimensions. Any online algorithm is required to add the red pairs.

3.3.1 Construction

We describe an adversary strategy with $\Omega_d(\varepsilon^{-d})$ points and show that any online algorithm returns a $(1+\varepsilon)$-spanner whose weight is $\Omega_d(\varepsilon^{-d})$ times the optimum weight. One can extend this result for arbitrary number of points, but that does not necessarily improve the lower bound. The final point set X consists of the points of the integer lattice $\mathbb{Z}^d$ in the hypercube $[0, \frac{1}{\varepsilon d})^d$, where $\varepsilon < \frac{1}{d}$. The points are presented in stages in order to deceive the online algorithm to add more edges than needed. In step $2i$, where $0 \leq i < \frac{1}{2\varepsilon}$, points $x \in X$ such that $\|x\|_1 = i$ will be given to the algorithm. In step $2i+1$, where $0 \leq i < \frac{1}{2\varepsilon}$, the adversary presents points $x \in X$ such that $\|x\|_1 = \lceil 1/\varepsilon \rceil - i$ (Figure 3.5). In other words, points are presented in batches according to their L_1 norms.

3.3.2 Competitive Ratio

Denote by X_i the set of points presented in step i. The idea is to show that there has to exist many edges between X_i and X_{i+1} in order to guarantee the $1+\varepsilon$ stretch-factor. Specifically, we define an *ordered-pair* as follows.

Definition 3.10 (ordered-pair)**.** A pair of points (x, y) in $\mathbb{R}^d$ is an *ordered-pair* if $x \in X_{2i}$ and $y \in X_{2i+1}$ for some i, and $x_k \leq y_k$ for all k, where x_k and y_k are the k-th coordinates of x and y respectively.

Now we show that any ordered-pair $(x, y) \in X_{2i} \times X_{2i+1}$ requires an edge in the spanner immediately after x and y are presented. To prove this, we show that previously presented points cannot serve as via points in a $(1+\varepsilon)$-path between x and y.

Lemma 3.11. *Let (x, y) be an ordered-pair. Then there is no $(1+\varepsilon)$-path between x and y that goes through any other point $z \in X_j$ with $j \leq i+1$.*

Proof. Let x_k, y_k, and z_k be the k-th coordinate of x, y, and z, respectively. Then the equality $\|x - z\|_1 + \|y - z\|_1 = \|x - y\|_1$ holds if and only if $x_k \leq z_k \leq y_k$, for all k. Since $z \neq x$ and $z \neq y$, we can conclude that $\|x\|_1 < \|z\|_1 < \|y\|_1$, which means that z is not added in the previous steps, which is a contradiction. So the equality does not hold and

$\|x-z\|_1 + \|y-z\|_1$ is strictly larger than $\|x-y\|_1$. As both expressions are integers, we have

$$\begin{aligned}\|x-z\|_1 + \|y-z\|_1 &\geq 1 + \|x-y\|_1 \\ &> \varepsilon\|x-y\|_1 + \|x-y\|_1 \\ &= (1+\varepsilon)\|x-y\|_1.\end{aligned}$$

The second inequality follows from the fact that $\|x-y\|_1 < \varepsilon^{-1}$ which holds for any two points in X. The above inequality shows that a $(1+\varepsilon)$-path between x and y cannot go through z and completes the proof of the lemma. $\square$

We next show that the total weight of the edges between ordered pairs is $\Omega_d(\varepsilon^{-2d})$.

Lemma 3.12. *The total weight of the edges between the ordered-pairs is $\Omega_d(\varepsilon^{-2d})$.*

Proof. Let $x = (x_1, \ldots, x_d)$ and $y = (y_1, \ldots, y_d)$ be two points in X. We show that if $x_k \in [\frac{1}{4\varepsilon(d+0.25)}, \frac{1}{4\varepsilon d}]$ for all $1 \leq k \leq d$, and $y_k \in [\frac{3}{4\varepsilon(d+0.25)}, \frac{3}{4\varepsilon d}]$ for all $1 \leq k \leq d-1$, then there is choice of y_d that makes (x, y) an ordered-pair. This would imply that there are $\Omega_d(\varepsilon^{-2d+1})$ ordered-pairs and by Lemma 3.11, each pair requires an edge of weight $\Omega_d(\varepsilon^{-1})$, thus the total weight of required edges would be $\Omega_d(\varepsilon^{-2d})$.

In order to find such a y_d, recall that $\|x\|_1 + \|y\|_1 = \lceil \varepsilon^{-1} \rceil$ holds because (x, y) is an ordered-pair. This equality uniquely determines the value of y_d,

$$y_d = \lceil \varepsilon^{-1} \rceil - \sum_{k=1}^{d} x_k - \sum_{k=1}^{d-1} y_k.$$

We just need to prove the inequalities $y_k \geq x_k$ and $y_k \leq 1/(\varepsilon d)$ for this unique y_k. This can simply be done by plugging the maximum (and minimum) values of x_ks and other y_ks and calculating the result,

$$y_d \geq \frac{1}{\varepsilon} - \frac{d}{4\varepsilon d} - \frac{3(d-1)}{4\varepsilon d} = \frac{3}{4\varepsilon d} > x_d.$$

Also,

$$y_d \leq \frac{1}{\varepsilon} + 1 - \frac{d}{4\varepsilon(d+0.25)} - \frac{3(d-1)}{4\varepsilon(d+0.25)} = 1 + \frac{1}{\varepsilon(d+0.25)} < \frac{1}{\varepsilon d}.$$

□

Now we can prove the main theorem of this section.

Theorem 3.13. *The competitive ratio of any online $(1+\varepsilon)$-spanner algorithm in $\mathbb{R}^d$ under the L_1-norm is $\Omega_d(\varepsilon^{-d})$.*

Proof. For the point set $X \subset \mathbb{R}^d$, the unit-distance graph is a Manhattan network: It contains a path of weight $\|xy\|_1$ for all $x, y \in X$. Its weight is $\Theta_d(\varepsilon^{-d})$ which is an upper bound for the weight of a $(1+\varepsilon)$-spanner for any $\varepsilon \geq 1$. By Lemma 3.12, any online algorithm returns a spanner of weight $\Omega_d(\varepsilon^{-2d})$. Thus its competitive ratio is $\Omega_d(\varepsilon^{-d})$. □

3.4 High Dimensional Euclidean Lower Bound

In this section, we show that for $t \in [(1+\varepsilon)\sqrt{2}, (1-\varepsilon)2]$, every online t-spanner algorithm in $\mathbb{R}^d$ must have competitive ratio $2^{\Omega(\varepsilon^2 d)}$. This would be a complementary lower bound for high dimensional Euclidean spaces, in contrast with the lower bound we proved in Section 3.3, where we assumed that d is a constant.

Theorem 3.14. *For $t \in [(1+\varepsilon)\sqrt{2}, (1-\varepsilon)2]$, the competitive ratio of any online t-spanner algorithm in $\mathbb{R}^d$ under the Euclidean norm is $2^{\Omega(\varepsilon^2 d)}$.*

Proof sketch. Let $A \subseteq \{\pm 1\}^d$ be a set of $2^{\Omega(\varepsilon^2 d)}$ points such that every $u, v \in A$ differ in $(1 \pm \varepsilon)\frac{d}{2}$ coordinates (such a set can be constructed randomly using Chernoff). In particular, $\|u - v\|_2$ is in $\sqrt{(1 \pm \varepsilon)2d}$. Every t-spanner for A must contain all $\binom{|A|}{2}$ edges, this is as the weight of any two edges is at least $2\sqrt{(1-\varepsilon)2d} > t \cdot \sqrt{(1+\varepsilon)2d}$.

Next the adversary introduces the point $\vec{0}$ with all zeros, which is at distance $\sqrt{d}$ from all other points. Let H be the star with $\vec{0}$ as a center. Then for every $u, v \in A$, there is a path in H of weight $2\sqrt{d} \le t \cdot \sqrt{(1-\varepsilon)2d} \le t \cdot \|v - u\|_2$. The competitive ratio is $\Omega\left(\binom{A}{2}/|A|\right) = 2^{\Omega(\varepsilon^2 d)}$. $\square$

Chapter 4

General metrics

4.1 The Ordered Greedy Spanner

In this section we study the online spanners problem on general metric spaces. The points arrive one by one, where for each new point we also receive its distances to all previously introduced points.

4.1.1 The Algorithm

In the offline setting, the celebrated greedy spanner algorithm [4] sorts the edges by increasing weight, and then processes them one by one, adding each edge if by the time of examination, the distance between its endpoints is too large. This algorithm achieves the existentially optimal[1] sparsity and lightness as a function of the stretch factor [50]. However, in the online model, we do not receive the edges in a sorted order, and therefore cannot execute the greedy algorithm. As an alternative, we propose here the *ordered greedy* algorithm. This is

[1]Specifically, if a t-spanner construction achieves an upper bound $m(n,t)$ and $l(n,t)$, resp., on the size and lightness of an n-vertex graph then this bound also holds for the greedy t-spanner [50].

a deterministic algorithm working against an adaptive adversary. The algorithm receives a stretch factor t, and works naturally as follows: We maintain a spanner H. When a point v_i arrives, we order its edges[2] in the original metric by weight. Each edge $\{v_{i'}, v_i\}$ is added to the spanner H if currently $d_H(v_{i'}, v_i) > t \cdot d_X(v_{i'}, v_i)$. Note that this algorithm can be easily executed in an online fashion.

4.1.2 The Analysis

Theorem 4.1. *Given an n-point metric space (X, d_X) in an (adaptive) adversarial order, with stretch factor $t = (2k-1)(1+\varepsilon)$ for $k \geq 2$ and $\varepsilon \in (0,1)$, the ordered greedy algorithm returns a spanner with $O(\varepsilon^{-1} \log \frac{1}{\varepsilon}) \cdot n^{1+\frac{1}{k}}$ edges and weight $O(\varepsilon^{-1} n^{\frac{1}{k}} \log^2 n) \cdot w(\mathrm{MST})$.*

Proof. The bounded stretch of our spanner is straightforward by construction, as every pair was examined at some point, and taken care of. Next we analyze the lightness.

In the online spanning tree problem, points of a finite metric space arrive one-by-one, and we need to connect each new point to a previous point to maintain a spanning tree. The ordered greedy algorithm connects each vertex v_i, to the closest vertex in $\{v_1, \ldots, v_{i-1}\}$. As was shown by Imase and Waxman [70], the tree created by the ordered greedy algorithm has lightness $O(\log n)$, which is the best possible [70]. Denote the online spanning tree by T_G. Note that the ordered greedy spanner H will contain T_G, as a shortest edge between a new vertex to a previously introduced vertex is always added to the spanner H. The following clustering lemma is frequently used for spanner constructions (see e.g. [3, 29, 36]). We provide a proof for the sake of completeness.

Claim 4.2. *For every $i \in \mathbb{N}$, the point set X can be partitioned into clusters $\mathcal{C}_i$ of diameter at most $D_i = \varepsilon \cdot (1+\varepsilon)^i$ w.r.t. the metric d_{T_G} such that $|\mathcal{C}_i| = O(\frac{w(T_G)}{\varepsilon \cdot (1+\varepsilon)^i})$.*

[2] By edges we mean point pairs in the metric space, we will often use notation from graph theory.

Proof. Let N_i be a maximal set of vertices such that for every $x, y \in N_i$, $d_{T_G}(x, y) > \frac{1}{2} \cdot D_i$. For every vertex $x \in N_i$ let $C_x = \left\{z : x = \mathrm{argmin}_{y \in N_i} d_X(z, y)\right\}$ be the Voronoi cell of x. Clearly, $\mathrm{diam}(C_x) \leq D_i$ for all x. Further, consider a continuous version of T_G (where each edge is an interval). Then as the graph T_G is connected, each cluster C_x contains at least $\frac{1}{4} D_i$ length of edges (as the balls $\left\{B_{T_G}(x, \frac{1}{4} D_i)\right\}_{x \in N_i}$ are pairwise disjoint). It follows that

$$|\mathcal{C}_i| = |N_i| \leq \frac{w(T_G)}{\frac{1}{4} D_i} = O\left(\frac{w(T_G)}{\varepsilon \cdot (1+\varepsilon)^i}\right),$$

as claimed. $\square$

For every i, consider the *scale* $E_i = \{e = \{u, v\} \in H : (1+\varepsilon)^{i-1} \leq d_X(u, v) < (1+\varepsilon)^i\}$. We now ready to bound the lightness.

Claim 4.3. *The weight of the ordered greedy spanner is* $O(n^{\frac{1}{k}} \cdot \varepsilon^{-2} \log^2 n) \cdot w(\mathrm{MST})$.

Proof. For scale i, consider the clusters $\mathcal{C}_i$ from Claim 4.2. We create an (unweighted) *cluster graph* $\mathcal{G}_i$ by contacting all the edges in each cluster and adding the edges E_i (i.e., for every $\{u, v\} \in E_i$ such that $u \in C_u$ and $v \in C_v$, we add the edge $\{c_u, c_v\}$ to $\mathcal{G}_i$. Consider a cluster $C \in \mathcal{C}_i$ where $C = (u_1, u_2, \ldots, u_{|C|})$ are the vertices ordered w.r.t. arrival times. We argue that for every $j = 1, \ldots, |C|$, the induced subgraph $T_G[\{u_1, \ldots, u_j\}]$ is connected. Assume for contradiction otherwise, and let j be the first index violating this rule. Let T_G^j be the tree T_G right after the arrival of u_j. On the one hand, T_G^j is connected, and so it contains a path P from u_j to $\{u_1, \ldots, u_{j-1}\}$. By the assumption that $T_G[\{u_1, \ldots, u_j\}]$ is disconnected, the a path P has interior vertices that are not $\{u_1, \ldots, u_j\}$. On the other hand, there is a path P' from u_j to $\{u_1, \ldots, u_{j-1}\}$ in $T_G[C]$. We conclude that T_G contains two different paths from u_1 to u_j, a contradiction to the fact that T_G is a tree. Furthermore, note that as T_G is a tree, the diameter of $T_G[\{u_1, \ldots, u_j\}]$ is bounded as well by D_i.

We next argue that $\mathcal{G}_i$ is a simple graph. Suppose for contradiction that there is a cluster

$C \in \mathcal{C}_i$ with a self loop. This implies that there are $v_a, v_b \in C$ such that $\{v_a, v_b\} \in E_i$. But this is impossible as $d_X(v_a, v_b) \le d_{T_G}(v_a, v_b) < D_i = \varepsilon \cdot (1+\varepsilon)^i$. Next, suppose for contradiction that there is an edge $\{C, C'\}$ in $\mathcal{G}_i$ of multiplicity two or higher. Then there are vertices $x_1, x_2 \in C$ and $y_1, y_2 \in C'$ such that $\{x_1, y_1\}, \{x_2, y_2\} \in E_i$. Assume, without loss of generality, that y_2 is the last arriving vertex among $\{x_1, x_2, y_1, y_2\}$. At the time $\{x_2, y_2\}$ is examined by the ordered greedy algorithm, there are paths from x_1 to x_2 and from y_1 to y_2 of weight at most D_i. As $\{x_1, y_1\}$ were already added to H, the spanner contains a x_2y_2-path of weight at most $2D_i + d_X(x_1, y_1) \le 2 \cdot \varepsilon \cdot (1+\varepsilon)^i + (1+\varepsilon)^i < t \cdot (1+\varepsilon)^i \le t \cdot d_X(x_2, y_2)$, which contradicts to the fact that the algorithm chose to add $\{x_2, y_2\}$. We conclude that $\mathcal{G}_i$ is indeed a simple graph.

Next, we argue that $\mathcal{G}_i$ has girth at least $2k+1$. Suppose for contradiction that there is a cycle $C_0C_1C_2\dots C_\beta C_0$ in $\mathcal{G}_i$ with $\beta \le 2k-1$, where the edge C_jC_{j+1} corresponds to the edge $\{x_j, y_{j+1}\} \in E_i$, modulo β. Assume, without loss of generality, that the edge $\{x_\beta, y_0\}$ was added last. Note that at the time the algorithm examines $\{x_\beta, y_0\}$, for every j, there is a path in H from y_j to x_j of weight at most D_i. Denote by $\widehat{H}$ the spanner H at this time. We conclude that

$$\begin{aligned} d_{\widehat{H}}(y_0, x_\beta) &\le \sum_{j=0}^{\beta} d_{\widehat{H}}(y_j, x_j) + \sum_{j=0}^{\beta-1} d_{\widehat{H}}(x_j, y_j) \\ &\le (\beta+1) \cdot D_i + \beta \cdot (1+\varepsilon)^i \\ &\le (2k-1)(1+3\varepsilon) \cdot (1+\varepsilon)^{i-1} \le (2k-1)(1+3\varepsilon) \cdot d_X(y_0, x_{2k-1}), \end{aligned}$$

which contradicts the fact that the edge $\{x_\beta, y_0\}$ was added to the algorithm.

A graph with girth $2k+1$ contains at most $O(n^{1+\frac{1}{k}})$ edges (see e.g. [20]). Hence the total weight of all the edges in E_i is bounded by

$$(1+\varepsilon)^i \cdot |E_i| = O(|\mathcal{C}_i|^{1+\frac{1}{k}}) \cdot (1+\varepsilon)^i = O(n^{\frac{1}{k}}) \cdot \frac{w(T_G)}{\varepsilon \cdot (1+\varepsilon)^i} \cdot (1+\varepsilon)^i = O(\varepsilon^{-1}\, n^{\frac{1}{k}}) \cdot w(T_G).$$

Let $e_{\max}$ be the heaviest edge in H, and let $i_{\max}$ be the index such that $\{x, y\} \in E_{i_{\max}}$. Note that for every scale $i \leq i_{\max} - \alpha$ have weight at most

$$w(E_i) \leq \binom{n}{2} \cdot (1+\varepsilon)^i \leq n^2 \cdot w(e_{\max}) \cdot (1+\varepsilon)^{-\alpha} \leq n^2 \cdot w(T_G) \cdot (1+\varepsilon)^{-\alpha} .$$

We conclude that the weight of the spanner is bounded by

$$\begin{aligned}
w(H) = \sum_{i \leq i_{\max}} w(E_i) &= \sum_{i=i_{\max}-\log_{1+\varepsilon} n^2}^{i_{\max}} w(E_i) + \sum_{i<i_{\max}-\log_{1+\varepsilon} n^2} w(E_i) \\
&\leq \log_{1+\varepsilon} n^2 \cdot O\left(\varepsilon^{-1} n^{\frac{1}{k}}\right) \cdot w(T_G) + \sum_{j \geq 1} w(T_G) \cdot (1+\varepsilon)^{-j} \\
&\leq O\left(\frac{\log n}{\log(1+\varepsilon)} \cdot \frac{n^{\frac{1}{k}}}{\varepsilon} + \frac{1}{\varepsilon}\right) \cdot w(T_G) \\
&= O\left(n^{\frac{1}{k}} \cdot \frac{\log n}{\varepsilon^2}\right) \cdot w(T_G) = O\left(n^{\frac{1}{k}} \cdot \frac{\log^2 n}{\varepsilon^2}\right) \cdot w(\mathrm{MST}) .
\end{aligned}$$

□

We next bound the sparsity of the ordered greedy spanner.

Claim 4.4. *The ordered greedy spanner has $O(\varepsilon^{-1} \log \frac{1}{\varepsilon}) \cdot n^{1+\frac{1}{k}}$ edges.*

Proof. We will assume for simplicity that the algorithm was executed with parameter $t = (2k-1)(1+2\varepsilon)$, later one can scale the results accordingly. Let $\{v_1, \ldots, v_n\}$ be the order in which the vertices arrived. Let H_i be the state of the spanner just after the arrival of v_i. We will greedily construct a laminar set system $N_0 \subseteq N_1 \subseteq \ldots$, where every pair of point in N_i will be at distance at least $(1+\varepsilon)^i$ w.r.t. the spanner H. Specifically, given a newly arrived vertex v_j which already joined N_i, v_j will join N_{i+1} if there is no vertex $v_{j'}$ (where $j' < j$) at distance $d_{H_j}(v_j, v_{j'}) \leq (1+\varepsilon)^{i+1}$ in the current spanner. Let $\Delta_i = \frac{(1+\varepsilon)^{i+1}-1}{\varepsilon}$. We will call each set N_i a *net*, and every point $v_j \in N_i$ a net point. We argue that the set N_i is Δ_i dominating, that is every vertex v_j has a net point $v_{j'} \in N_i$, such that at the time v_j

arrived, $d_{H_j}(v_j, v_{j'}) \le \Delta_i$.

Indeed, by induction there is a net point $v_q \in N_{i-1}$ such that $d_{H_j}(v_j, v_q) \le \Delta_{i-1}$, and $q < j$. If $v_q \in N_i$ then we are done. Otherwise, there is a point $v_p \in N_i$ such that $d_{H_q}(v_q, v_p) \le (1+\varepsilon)^i$ and $s < q$. Implying $d_{H_j}(v_j, v_p) \le d_{H_j}(v_j, v_q) + d_{H_q}(v_q, v_p) \le \frac{(1+\varepsilon)^i - 1}{\varepsilon} + (1+\varepsilon)^i = \frac{(1+\varepsilon)^{i+1}-1}{\varepsilon} = \Delta_i$. For i too small, let $N_i = X$, and $\Delta_i = 0$.

For every i, consider the *scale* $E_i = \{e = \{u, v\} \in H : (1+\varepsilon)^{i-1} \le d_X(u, v) < (1+\varepsilon)^i\}$. Set $s = \lceil \log_{1+\varepsilon}(\frac{4k}{2k-1} \cdot \frac{1+\varepsilon}{\varepsilon^{2}.}) \rceil$.

We argue that for every i, $|E_i| \le O\left(|N_{i-s} \setminus N_{i+s}|\right)^{1+\frac{1}{k}}$. For this goal, we construct an auxiliary graph $\mathcal{G}_i$ with N_{i-s} as vertices and E_i as edges. Specifically, for every $\{x, y\} \in E_i$, let $v_x v_y \in N_i$ be the closest vertices to x, y in N_i at the time they were added. Then we will add the edge $\{v_x, v_y\}$ to $\mathcal{G}_i$.

Clearly $\mathcal{G}_i$ does not contain self loops, as the distance between two vertices x, y who has the same closest vertex in N_i is bounded by $2\Delta_{i-s} < (1+\varepsilon)^{i-1}$. Suppose for contradiction that there is an edge $\{v, u\}$ in $\mathcal{G}_i$ of multiplicity two or higher. Then there are vertices x_1, x_2, y_1, y_2 such that v was the closest vertex to x_1, x_2, u was the closest vertex to y_1, y_2, and $\{x_1, y_1\}, \{x_2, y_2\} \in E_i$. Assume, without loss of generality, that y_2 is the last arriving vertex among $\{x_1, x_2, y_1, y_2\}$. At the time $\{x_2, y_2\}$ is examined by the ordered greedy algorithm, the pairs (x_1, x_2) and (y_1, y_2) already were examined, and hence H contain path from x_1 to x_2 and from y_1 to y_2 of weight at most $2 \cdot \Delta_{i-s}$. By our assumption, $\{x_1, y_1\}$ was already added to H. Hence the spanner contains a x_2y_2-path of weight at most

$$\begin{aligned} d_H(x_2, y_2) &\le d_H(x_2, y_1) + d_X(x_1, y_1) + d_H(y_1, y_2) \\ &\le 4\Delta_{i-s} + (1+\varepsilon)^i \\ &\le \frac{4(1+\varepsilon)^{i-s}}{\varepsilon} + (1+\varepsilon)^i \\ &\le \left(1 + \varepsilon + \frac{4}{\varepsilon(1+\varepsilon)^{s-1}}\right)(1+\varepsilon)^{i-1} \le t \cdot d_X(x_2, y_2) \ , \end{aligned}$$

a contradiction to the fact that the algorithm choose to add $\{x_2, y_2\}$.

Next, we argue that $\mathcal{G}_i$ has girth at least $2k+1$. Suppose for contradiction that there is a cycle $u_0u_1u_2 \ldots u_\beta u_0$ in $\mathcal{G}_i$ with $\beta \le 2k-1$, where the edge u_ju_{j+1} corresponds to the edge $\{x_j, y_{j+1}\} \in E_i$, modulo β. Assume, without loss of generality, that the edge $\{x_\beta, y_0\}$ was added last. Note that at the time the algorithm examines $\{x_\beta, y_0\}$, for every j, there is a path in H from y_j to x_j of weight at most $2 \cdot \Delta_{i-s}$. Denote by $\widehat{H}$ the spanner H at this time. We conclude that

$$\begin{aligned} d_{\widehat{H}}(y_0, x_\beta) &\le \sum_{j=0}^{\beta} d_{\widehat{H}}(y_j, x_j) + \sum_{j=0}^{\beta-1} d_{\widehat{H}}(x_j, y_j) \\ &\le (\beta+1) \cdot 2\Delta_{i-s} + \beta \cdot (1+\varepsilon)^i \\ &\le 2k \cdot \frac{2(1+\varepsilon)^{i-s}}{\varepsilon} + (2k-1) \cdot (1+\varepsilon)^i \\ &= (2k-1)(1 + \varepsilon + \frac{4k}{2k-1} \cdot \frac{1}{\varepsilon \cdot (1+\varepsilon)^{s-1}}) \cdot (1+\varepsilon)^{i-1} \\ &\le (2k-1)(1+2\varepsilon) \cdot d_X(y_0, x_{2k-1}) \ , \end{aligned}$$

which contradicts the fact that the edge $\{x_\beta, y_0\}$ was added to the algorithm.

Consider a pair of net points $u, v \in N_{i+s}$. Then the distance between u, v in $\mathcal{G}_i$ has to be at least 3. Otherwise, if $d_{\mathcal{G}_i}(u, v) \le 2$, there is a net point $z \in N_{i-s}$ and two edges

$\{x_0, y_1\}, \{x_1, y_2\} \in E_i$ corresponding to $\{u, z\}, \{z, v\}$ in $\mathcal{G}_i$. Then following the logic above,

$$\begin{aligned} d_{\widehat{H}}(u,v) &\le d_{\widehat{H}}(u,x_0) + d_{\widehat{H}}(x_0,y_1) + d_{\widehat{H}}(y_1,x_1) + d_{\widehat{H}}(x_1,y_2) + d_{\widehat{H}}(y_2,v) \\ &\le 4\Delta_{i-s} + 2 \cdot (1+\varepsilon)^i \\ &\le \left(\frac{8}{\varepsilon(1+\varepsilon)^s} + 2 \right) \cdot (1+\varepsilon)^i < (1+\varepsilon)^{i+s} , \end{aligned}$$

a contradiction to the fact that both u, v joined N_i. It follows that there are no edges between vertices in N_{i+s}, and furthermore, each vertex in $N_{i-s} \setminus N_{i+s}$ is connected to at most a single vertex in N_{i+s}. We conclude that the number of edges incident on N_{i+s} vertices is bounded by $|N_{i-s} \setminus N_{i+s}|$. As the induced graph $\mathcal{G}_i[N_{i-s} \setminus N_{i+s}]$ has girth $2k+1$, it contains at most $O\left(|N_{i-s} \setminus N_{i+s}|^{1+\frac{1}{k}} \right)$ edges (see e.g. [20]). We conclude

$$|E_i| = E\left(\mathcal{G}_i\right) = E\left(G\left[N_{i-s} \setminus N_{i+s}\right]\right) + |N_{i-s} \setminus N_{i+s}| = O\left(|N_{i-s} \setminus N_{i+s}|^{1+\frac{1}{k}} \right) .$$

We conclude a bound on the number of edges:

$$\begin{aligned} |E(H)| &= \sum_{i \ge 0} |E_i| \le \sum_{i \ge 0} O\left(|N_{i-s} \setminus N_{i+s}|^{1+\frac{1}{k}} \right) \\ &\le O(n^{\frac{1}{k}}) \cdot \sum_{i \ge 0} |N_{i-s} \setminus N_{i+s}| = O(s \cdot n^{1+\frac{1}{k}}) = O(\frac{\log \frac{1}{\varepsilon}}{\varepsilon} \cdot n^{1+\frac{1}{k}}) , \end{aligned}$$

where the second to last equality follows as each vertex can participate in at least $2s$ different addends in the sum. $\square$

The theorem now follows. $\square$

4.2 Lower Bound for General metrics

In this section we prove an $\Omega(\frac{1}{k} \cdot n^{\frac{1}{k}})$ lower bound on the competitive ratio of an online $(2k-1)$-spanner of n-vertex graphs. Our lower bound holds in both cases where the quality is measured by number of edges or the weight. It follows that our upper bound in Theorem 4.1 cannot be substantially improved, even if we consider competitive ratio instead of lightness/sparsity.

4.2.1 Erdős Girth Conjecture

Recall that the Erdős Girth Conjecture [42] states that for every $n, k \geq 1$, there exists an n-vertex graph with $\Omega(n^{1+\frac{1}{k}})$ edges and girth $2k+2$. The proof of the following lemma is based on a counting argument form the recent lower bound proof for (static) vertex fault tolerant emulators by Bodwin, Dinitz, and Nazari [18].

Lemma 4.5. *Assuming the Erdős girth conjecture, for every $n, k \geq 1$, there exists an n-point metric space (X, d_X) with diameter $2k-1$, such that every $(2k-1)$-spanner has $\Omega(\frac{1}{k} \cdot n^{1+\frac{1}{k}})$ edges and weight $\Omega(n^{1+\frac{1}{k}})$.*

Proof. Let $G = (V, E_G)$ be the graph fulfilling the Erdős girth conjecture. That is, G is an unweighted n-vertex graph with girth $2k+2$ and $|E_G| = \Omega(n^{1+\frac{1}{k}})$ edges. Set a metric d_X over V as follows, [3]

$$\forall u, v \in V \quad d_X(u, v) = \min\{d_G(u, v), 2k-1\}.$$

Suppose that $H = (V, E_H)$ is a $(2k-1)$-spanner for (V, d_X) with weight function w_H, where the weight of an edge $e' \in \{u, v\} \in E_H$ is $w_H(e') = d_X(u, v)$. Let $E' = E_H \setminus E_G$ be the edges

[3] Note that $\forall x, y, z \in V$, $d_X(x, z) = \min\{d_G(x, z), 2k-1\} \leq \min\{d_G(x, y) + d_G(y, z), 2k-1\} \leq \min\{d_G(x, y), 2k-1\} + \min\{d_G(y, z), 2k-1\} = d_X(x, y) + d_X(y, z)$. Thus d_X is a metric space.

of H which are not in G. We say that an edge $e' \in E'$ *covers* an edge $e \in E_G$, if there is a shortest path in G between the endpoints of e' going through e of weight at most k. Note that as e' has weight at most k, there is a unique shortest path in G between its endpoints. In particular, each edge $e \in E'$ can cover at most k edges in E_G.

Consider an edge $e = \{v_0, v_s\} \in E_G \setminus E_H$. We argue that some edge $e' \in E'$ must cover e. Suppose for contradiction otherwise, and let $P = (v_0, v_1, \ldots, v_s)$ be the shortest path in H between the endpoints v_0, v_s of e. Suppose first that P contains an edge v_i, v_{i+1} of weight at least $w_H(\{v_i, v_{i+1}\}) \geq k+1$. In particular, $d_G(\{v_i, v_{i+1}\}) \geq k+1$. Then by the triangle inequality, $d_G(v_0, v_i) + d_G(v_{i+1}, v_s) \geq d_G(v_i, v_{i+1}) - d_G(v_0, v_s) \geq k$. It follows that P has weight at least $2k+1$, a contradiction to the fact that H is a $2k-1$ spanner. We conclude that for every $i \in \{0, \ldots, s-1\}$, $d_X(v_i, v_{i+1}) = d_G(v_i, v_{i+1}) \leq k$. In particular, in G there is a unique path $P_i = (u_0^i, \ldots, u_{s_i}^i)$ between v_i to v_{i+1} of weight $d_G(v_i, v_{i+1}) \leq k$. As no edge covers e, e does not belong to any of these paths. The concatenation of this paths $P_0 \circ P_1 \circ \cdots \circ P_{s-1}$ is a path in G of at most $2k-1$ edges between the endpoints of e. It follows that G contains a $2k$-cycle, a contradiction.

For conclusion, as every edge in $E_G \setminus E_H$ is covered, and every edge in $E' = E_H \setminus E_G$ can cover at most k edges, it follows that $|E_H \setminus E_G| \geq \frac{1}{k} \cdot |E_G \setminus E_H|$. In particular,

$$|E_H| = |E_H \cap E_G| + |E_H \setminus E_G| \geq |E_H \cap E_G| + \frac{1}{k} \cdot |E_G \setminus E_H| \geq \frac{1}{k} \cdot |E_G| \,.$$

To bound the weight, for each edge $e' = \{s, t\} \in E'$, let $A_{e'}$ be the set of edges in E_G covered by e'. Note that $w_H(e') = d_G(s,t) = |A_{e'}|$. As all the edges in $E_G \setminus E_H$ are covered, we conclude

$$\begin{aligned}
w_H(E_H) &= w_H(E_H \cap E_G) + w_H(E_H \setminus E_G) \\
&= |E_H \cap E_G| + \sum_{e' \in E'} |A_{e'}| \\
&\geq |E_H \cap E_G| + |E_G \setminus E_H| = |E_G| = \Omega(n^{1+\frac{1}{k}}),
\end{aligned}$$

the lemma now follows. $\square$

4.2.2 Competitive Ratio Lower Bound

Theorem 4.6. *Assuming Erdős girth conjecture, the competitive ratio of any online* $(2k-1)$*-spanner algorithm for n-point metrics is* $\Omega(\frac{1}{k} \cdot n^{\frac{1}{k}})$*, for both weight and edges.*
In more details, there is an n-point metric space (X, d_X) *with a* $(2k-1)$*-spanner* $H_{\mathrm{OPT}} = (X, E_{\mathrm{OPT}})$*, and order over* X *for which every* $(2k-1)$*-spanner produced by an online algorithm will have* $\Omega(\frac{1}{k} \cdot n^{\frac{1}{k}}) \cdot |E_{\mathrm{OPT}}|$ *edges, and* $\Omega(\frac{1}{k} \cdot n^{\frac{1}{k}}) \cdot w(H_{\mathrm{OPT}})$ *weight.*

Proof. Consider the metric space (X, d_X) from Lemma 4.5 with parameters $n-1$ and k. Let X' be the metric space X with an additional point r at distance $\frac{k-1}{2}$ from all the points in X. Note that no pairwise distance is changed due to the introduction of r. The adversary provides the online algorithm the points in X first (in some arbitrary order), and the point r last. After the algorithm received all the points in X', it has a $2k-1$-spanner H_{n-1}. According to Lemma 4.5, H_{n-1} has $\Omega(\frac{1}{k} \cdot (n-1)^{1+\frac{1}{k}}) = \Omega(\frac{1}{k} \cdot n^{1+\frac{1}{k}})$ edges, and $\Omega(n^{1+\frac{1}{k}})$ weight.

Next the algorithm introduces r. Consider the spanner $S = (X', E_S)$ consisting of $n-1$ edges with r as a center. Note that the maximum distance in S is $2k-1$, and hence S is a $2k-1$ spanner as required. Note that S contains $n-1$ edges of weight $\frac{2k-1}{2}$ each, and thus have total weight of $O(nk)$. We conclude

$$\begin{aligned}
|E_{H_n}| &\geq |E_{H_{n-1}}| = \Omega(\tfrac{1}{k} \cdot n^{1+\frac{1}{k}}) = \Omega(\tfrac{1}{k} \cdot n^{\frac{1}{k}}) \cdot |E_S| \ . \\
w(E_{H_n}) &\geq w(E_{H_{n-1}}) = \Omega(n^{1+\frac{1}{k}}) = \Omega(\tfrac{1}{k} \cdot n^{\frac{1}{k}}) \cdot w(S) \ .
\end{aligned}$$

□

4.3 Ultrametrics

4.3.1 Definition

An ultrametric (X, d) is a metric space satisfying a strong form of the triangle inequality, that is, for all $x, y, z \in X$, $d(x, z) \leq \max\{d(x, y), d(y, z)\}$. A related notion is a k-hierarchical well-separated tree (k-HST).

Definition 4.7 (α-HST)**.** A metric (X, d_X) is a α-hierarchical well-separated tree (α-HST) if there exists a bijection φ from X to leaves of a rooted tree T in which:

- Each node $v \in T$ is associated with a label $\ell(v)$ such that $\ell(v) = 0$ if v is a leaf and $\ell(v) \geq \alpha\ell(u)$ if v is an internal node and u is any child of v.
- $d_X(x, y) = \ell(\mathrm{lca}(\varphi(x), \varphi(y)))$ where $\mathrm{lca}(u, v)$ is the least common ancestor of any two given nodes u, v in T.

It is well known that any ultrametric is a 1-HST, and any k-HST is an ultrametric [6].

4.3.2 Spanner Construction

Suppose that we are given an HST in the online model. Construct a spanner H using the following algorithm: for every arriving vertex v, let u be the first vertex in the order of arrival among all the nearest neighbors of v. We add the edge $\{u, v\}$ to the spanner H. Note that H is a spanning tree at all times (we will later argue that it is actually an MST).

We show that for general ultrametrics, the online algorithm can maintain a spanner of lightness arbitrarily close to 1 (with constant stretch).

4.3.3 Analysis

Lemma 4.8. *If U is an α-HST, then the spanner H has distortion $2 \cdot \frac{\alpha}{\alpha-1}$.*

Proof. Think of the representation of the HST as a tree with labeled internal nodes. For every internal node χ, we call the first descendent in the order of arrival the *center* of χ. Consider a vertex v at the time of its arrival, let χ be an internal node which is an ancestor of v, and let u be the center of χ. We argue that $d_H(v, u) \leq t \cdot d_U(v, u)$ for $t = \frac{\alpha}{\alpha-1}$. The proof is by induction. The induction step is immediate if the edge $\{u, v\}$ was added to H. Otherwise, let χ' be the highest internal node which is an ancestor of v but has a center other than u. Let x be the center of χ'. At the time when x arrives, it was the only descendent of χ'. In particular, the closest neighbors of x at this time is u (as otherwise, there must be an internal vertex χ'' between χ' and χ with center other than u). As u is the center of χ, it is the first arriving descendent of χ. In other words, u is the first vertex in the order of arrival among all the nearest neighbors of u. We conclude that $\{x, u\} \in H$. As U is an

α-HST $\ell(\chi') \leq \frac{1}{\alpha}\ell(\chi)$. By the induction hypothesis, $d_H(v,x) \leq t \cdot d_U(v,x)$. We conclude

$$\begin{aligned} d_H(v,u) &\leq d_H(v,x) + d_H(x,u) \leq t \cdot d_U(v,x) + d_U(x,u) \\ &= t \cdot \ell(\chi') + \ell(\chi) \leq \left(\frac{t}{\alpha} + 1\right) \cdot \ell(\chi) = t \cdot \ell(\chi) = t \cdot d_U(v,u)\,. \end{aligned}$$

For two arbitrary vertices u, v, let $\chi = \text{lca}(u,v)$, and let x be the center of χ. By the definition of HST, $d_U(v,x), d_U(x,u) \leq d_U(v,u)$. Using the previous argument,

$$d_H(v,u) \leq d_H(v,x) + d_H(x,u) \leq t \cdot (d_U(v,x) + d_U(x,u)) = 2t \cdot d_U(v,u)\,.$$

□

Lemma 4.9. *The spanner H is an MST of U.*

Proof. Assume for contradiction otherwise. Then $w\left(\text{MST}(U)\right) < w\left(H\right)$. Let T be an MST of U containing the maximum number edges of H. Let $\{u,v\} = e \in H \setminus T$ be some edge. Assume, without loss of generality, that u arrived before v, and let $\chi = \text{lca}(u,v)$. As the algorithm added edge $\{u,v\}$ to H, necessarily u is the center of χ. Further, there is a child node χ_v of χ, where v is a unique descendent of χ_v (at the time of arrival). Let S_v be the set of all descendants of χ_v in U. Then T contains at least one edge from the vertices of S_v to a vertex outside of S_v. Let $e' \in T$ be such an edge that is on the unique uv-path in T. Then $w(e') \geq \ell(\chi) = w(e)$, and $T \cup \{e\} \setminus \{e'\}$ is a spanning tree of U, of weight at most $w(T)$. A contradiction to the maximality of T. □

Theorem 4.10. *Given an ultrametric U, for every $\alpha \geq 1$, an online algorithm can maintain a $\frac{2\alpha^2}{\alpha-1}$-spanner of wieght $\alpha \cdot w(\text{MST})$. Alternatively, for every $\varepsilon > 0$, it can maintain a spanner of weigth $(1+\varepsilon) \cdot w(\text{MST})$ and stretch $\frac{2(1+\varepsilon)^2}{\varepsilon} = O(\varepsilon^{-1})$.*

Proof. Let U_α be the α-HST for U where we round every distance up to the next integer power of α. That is, $d_{U_\alpha}(u,v) = \alpha^{\lceil \log_\alpha d_U(u,v) \rceil}$. Note that $d_U(u,v) \leq d_{U_\alpha}(u,v) < \alpha \cdot d_U(u,v)$.

In particular, the weight of the MST in U_α is larger than the MST of U by at most a factor α. We run the online algorithm above on U_α instead of U. As a result, we get a spanner H_α of U_α with stretch $2 \cdot \frac{\alpha}{\alpha-1}$ and lightness 1 (w.r.t. U_α). Let H be the same spanner with the original weights. Then for every pair of vertices u, v

$$d_H(u,v) \le d_{H_\alpha}(u,v) \le \frac{2\alpha}{\alpha-1} \cdot d_{U_\alpha}(u,v) \le \frac{2\alpha^2}{\alpha-1} \cdot d_U(u,v)\,.$$

The weight of H is bounded by $w(H) \le w(H_\alpha) = w(\mathrm{MST}(U_\alpha)) \le \alpha \cdot w(\mathrm{MST}(U))$. □

Remark 4.11. *The minimal possible stretch in the Theorem 4.10 above is* 8*, which is obtained for lightness* $\alpha = 2$*. This stretch is the best possible stretch obtained by a spanning tree. Indeed, consider the metric induced on the leaves of the full binary tree. One can observe that this is an ultrametric. Chan et al. [28] showed that for every* $\varepsilon > 0$*, there is a full binary tree large enough such that every tree over its set of leaves has stretch greater than* $8 - \varepsilon$*. (see [45, 46, 48, 63] for further details on the Steiner point removal problem.)*

4.3.4 Establishing the Trade-off

Note that a spanner with stretch smaller than 2 might require $\Omega(n^2)$ edges. Indeed, the uniform metric (where all distances are 1) is an ultrametric, and every spanner with a missing edge has stretch at least 2. Similarly, it follows that every such spanner will have lightness $\Omega(n)$. In the next theorem we show that an online algorithm can get arbitrarily close to stretch 2.

Theorem 4.12. *Given an ultrametric* U*, for every* $\varepsilon \in (0, \frac{1}{2})$*, an online algorithm can maintain an* $(2+\varepsilon)$*-spanner with* $O(\varepsilon^{-1}\log\varepsilon^{-1}) \cdot n$ *edges and* $O(\varepsilon^{-2}) \cdot w(\mathrm{MST})$ *weight.*

Proof. For every $i \in \{0, 1, \dots, \kappa\}$ with $\kappa = \left\lfloor \log_{1+\varepsilon} \varepsilon^{-1} \right\rfloor$, let U_i be the ultrametric U, where for every pair of vertices, $d_{U_i}(u,v)$ is defined to be the $(1+\varepsilon)^i \cdot \varepsilon^{-j}$ for the minimal index j

such that $d_{U_i}(u,v) \le (1+\varepsilon)^i \cdot \varepsilon^{-j}$. We construct a spanner H_i for U_i using the algorithm above. The final spanner will be $H = \bigcup_i H_i$ (with the original weights).

The sparsity is straightforward, as we have $\kappa+1 = O(\log_{1+\varepsilon} \varepsilon^{-1}) = O(\varepsilon^{-1} \log \varepsilon^{-1})$ trees. For the lightness, let T be an MST of the ultrametric X. Denote by T_i the MST of U_i. For every edge $e \in T$, it holds that

$$\sum_{i=0}^{\kappa} w_{U_i}(e) \le w_U(e) \sum_{i=0}^{\kappa} (1+\varepsilon)^i = \frac{(1+\varepsilon)^{\kappa+1} - 1}{\varepsilon} \cdot w_U(e) = O(\varepsilon^{-2}) \cdot w_U(e).$$

We now can bound the weight of H as follows:

$$\begin{aligned} w_U(H) &\le \sum_{i=0}^{\kappa} w_U(H_i) \le \sum_{i=0}^{\kappa} w_{U_i}(H_i) = \sum_{i=0}^{\kappa} w_{U_i}(T_i) \le \sum_{i=0}^{\kappa} w_{U_i}(T) \\ &= \sum_{i=0}^{\kappa} \sum_{e\in T} w_{U_i}(e) = \sum_{e\in T} \sum_{i=0}^{\kappa} w_{U_i}(e) = \sum_{e\in T} O(\varepsilon^{-2})\, w_U(e) = O(\varepsilon^{-2})\, w_U(T). \end{aligned}$$

It remains to analyze the stretch of H. For every pair of vertices $u, v \in U$, there are unique indices i, j such that $(1+\varepsilon)^{i-1} \cdot \varepsilon^{-j} < d_U(u,v) \le (1+\varepsilon)^i \cdot \varepsilon^{-j}$. Hence in U_i it holds that $d_{U_i}(u,v) \le (1+\varepsilon) \cdot d_U(u,v)$. As U_i is an ε^{-1}-HST, it holds that

$$d_H(u,v) \le d_{H_i}(u,v) \le \frac{2\varepsilon^{-1}}{\varepsilon^{-1} - 1} \cdot d_{U_i}(u,v) \le \frac{2}{1-\varepsilon} \cdot (1+\varepsilon) \cdot d_U(u,v) \le 2(1+3\varepsilon) \cdot d_U(u,v)\,.$$

One can obtain the stretch factor $2+\varepsilon$, stated in the theorem, by scaling ε accordingly. $\square$

Chapter 5

Conclusions and future work

We studied online spanners for points in metric spaces. In the Euclidean d-space, we presented an online $(1+\varepsilon)$-spanner algorithm with competitive ratio $O(\varepsilon^{1-d}\log n)$, improving the previous bound of $O_d(\varepsilon^{-(d+1)}\log n)$ from [17]. In fact, the spanner maintained by the algorithm has $O_d(\varepsilon^{1-d}\log\varepsilon^{-1})\cdot n$ edges, almost matching the (offline) optimal bound of $O_d(\varepsilon^{1-d})\cdot n$. Moreover, in the plane, a tighter analysis of the same algorithm provides an almost quadratic improvement of the competitive ratio to $O(\varepsilon^{-3/2}\log\varepsilon^{-1}\log n)$, by comparing the online spanner with an instance-optimal spanner directly, circumventing the comparison to an MST (i.e., lightness). Note that, the logarithmic dependence on n is unavoidable due to a $\Omega((\varepsilon^{-1}/\log\varepsilon^{-1})\log n)$ lower bound in the real line [17]. However, our lower bound $\Omega(\varepsilon^{-d})$ under L_1-norm in $\mathbb{R}^d$ shows a dependence on the dimension. This leads to the following question.

Question 5.1. Does the competitive ratio of an online $(1+\varepsilon)$-spanning algorithm for n points in $\mathbb{R}^d$ necessarily grow proportionally with $\varepsilon^{-f(d)}\cdot\log n$, where $\lim_{d\to\infty} f(d)=\infty$?

For $t\in[(1+\varepsilon)\sqrt{2},(1-\varepsilon)2]$, we showed that every online t-spanner algorithm in $\mathbb{R}^d$ must have competitive ratio $2^{\Omega(\varepsilon^2 d)}$. Next, we studied online spanners in general metrics. We showed that the *ordered greedy* algorithm maintains a spanner with $O(\varepsilon^{-1}\log\frac{k}{\varepsilon})\cdot n^{1+\frac{1}{k}}$ edges and $O(\varepsilon^{-1}n^{\frac{1}{k}}\log^2 n)$ lightness, with stretch factor $t=(2k-1)(1+\varepsilon)$ for $k\geq 2$ and $\varepsilon\in(0,1)$, for a sequence of n points in a metric space. Moreover, we show that these bounds cannot be significantly improved, by introducing an instance that achieves an $\Omega(\frac{1}{k}\cdot n^{1/k})$ competitive ratio on both sparsity and lightness. Finally, we established the trade-off among stretch, number of edges and lightness for points in ultrametrics, showing that one can maintain a $(2+\varepsilon)$-spanner for ultrametrics with $O(n\cdot\varepsilon^{-1}\log\varepsilon^{-1})$ edges and $O(\varepsilon^{-2})$ lightness.

Bibliography

[1] N. Alon, B. Awerbuch, Y. Azar, N. Buchbinder, and J. Naor. A general approach to online network optimization problems. *ACM Transactions on Algorithms (TALG)*, 2(4):640–660, 2006.

[2] N. Alon and Y. Azar. On-line Steiner trees in the Euclidean plane. *Discrete & Computational Geometry*, 10:113–121, 1993.

[3] S. Alstrup, S. Dahlgaard, A. Filtser, M. Stöckel, and C. Wulff-Nilsen. Constructing light spanners deterministically in near-linear time. *Theoretical Computer Science*, 2022.

[4] I. Althöfer, G. Das, D. Dobkin, D. Joseph, and J. Soares. On sparse spanners of weighted graphs. *Discrete & Computational Geometry*, 9(1):81–100, 1993.

[5] B. Awerbuch, Y. Azar, and Y. Bartal. On-line generalized Steiner problem. *Theoretical Computer Science*, 324(2-3):313–324, 2004.

[6] Y. Bartal, N. Linial, M. Mendel, and A. Naor. Some low distortion metric Ramsey problems. *Discret. Comput. Geom.*, 33(1):27–41, 2005.

[7] S. Baswana. Streaming algorithm for graph spanners—single pass and constant processing time per edge. *Inf. Process. Lett.*, 106(3):110–114, 2008.

[8] S. Baswana, S. Khurana, and S. Sarkar. Fully dynamic randomized algorithms for graph spanners. *ACM Trans. Algorithms*, 8(4):35:1–35:51, 2012.

[9] R. Becker, S. Forster, A. Karrenbauer, and C. Lenzen. Near-optimal approximate shortest paths and transshipment in distributed and streaming models. *SIAM J. Comput.*, 50(3):815–856, 2021.

[10] M. d. Berg, O. Cheong, M. J. Kreveld, and M. H. Overmars. *Computational geometry: algorithms and applications, 3rd Edition.* Springer, 2008. see here.

[11] T. Bergamaschi, M. Henzinger, M. P. Gutenberg, V. V. Williams, and N. Wein. New techniques and fine-grained hardness for dynamic near-additive spanners. In *Proc. 32nd ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1836–1855, 2021.

[12] P. Berman and C. Coulston. On-line algorithms for Steiner tree problems. In *Proc. 29th ACM Symposium on Theory of Computing (STOC)*, pages 344–353, 1997.

[13] A. Bernstein, S. Forster, and M. Henzinger. A deamortization approach for dynamic spanner and dynamic maximal matching. In *Proc. 30th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1899–1918, 2019.

[14] S. Bhore, A. Filtser, H. Khodabandeh, and C. D. Tóth. Online spanners in metric spaces. *arXiv preprint arXiv:2202.09991*, 2022.

[15] S. Bhore and C. D. Tóth. Light Euclidean Steiner spanners in the plane. In *Proc. 37th International Symposium on Computational Geometry (SoCG)*, volume 189 of *LIPIcs*, pages 31:1–17. Schloss Dagstuhl, 2021.

[16] S. Bhore and C. D. Tóth. On Euclidean Steiner $(1+\varepsilon)$-spanners. In *Proc. 38th Symposium on Theoretical Aspects of Computer Science (STACS)*, volume 187 of *LIPIcs*, pages 13:1–13:16. Schloss Dagstuhl, 2021.

[17] S. Bhore and C. D. Tóth. Online Euclidean spanners. In *Proc. 29th European Symposium on Algorithms (ESA)*, volume 204 of *LIPIcs*, pages 116:1–16:19. Schloss Dagstuhl, 2021.

[18] G. Bodwin, M. Dinitz, and Y. Nazari. Vertex Fault-Tolerant Emulators. In M. Braverman, editor, *13th Innovations in Theoretical Computer Science Conference (ITCS 2022)*, volume 215 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 25:1–25:22, Dagstuhl, Germany, 2022. Schloss Dagstuhl – Leibniz-Zentrum für Informatik.

[19] G. Bodwin and S. Krinninger. Fully dynamic spanners with worst-case update time. In *Proc. 24th Annual European Symposium on Algorithms (ESA)*, volume 57 of *LIPIcs*, pages 17:1–17:18. Schloss Dagstuhl, 2016.

[20] B. Bollobás. *Extremal Graph Theory.* Academic Press, London, 1978.

[21] A. Borodin and R. El-Yaniv. *Online Computation and Competitive Analysis*. Cambridge University Press, 1998. see here.

[22] G. Borradaile, E. W. Chambers, D. Eppstein, W. Maxwell, and A. Nayyeri. Low-stretch spanning trees of graphs with bounded width. *arXiv preprint arXiv:2004.08375*, 2020.

[23] G. Borradaile, H. Le, and C. Wulff-Nilsen. Greedy spanners are optimal in doubling metrics. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, San Diego, California, USA, January 6-9, 2019*, pages 2371–2379, 2019.

[24] P. Bose, P. Carmi, M. Farshi, A. Maheshwari, and M. Smid. Computing the greedy spanner in near-quadratic time. *Algorithmica*, 58(3):711–729, 2010.

[25] P. Bose, J. D. Carufel, P. Morin, A. van Renssen, and S. Verdonschot. Towards tight bounds on theta-graphs: More is not always better. *Theor. Comput. Sci.*, 616:70–93, 2016.

[26] P. Bose, J. Gudmundsson, and P. Morin. Ordered theta graphs. *Computational Geometry*, 28(1):11–18, 2004.

[27] P. B. Callahan and S. R. Kosaraju. Faster algorithms for some geometric graph problems in higher dimensions. In V. Ramachandran, editor, *Proceedings of the Fourth Annual ACM/SIGACT-SIAM Symposium on Discrete Algorithms, 25-27 January 1993, Austin, Texas, USA*, pages 291–300. ACM/SIAM, 1993. see here.

[28] T.-H. H. Chan, D. Xia, G. Konjevod, and A. Richa. A tight lower bound for the Steiner point removal problem on trees. In *Proc. 9th Conference on Approximation Algorithms for Combinatorial Optimization Problems, and 10th Conference on Randomization and Computation (APPROX/RANDOM)*, pages 70–81, Berlin, 2006. Springer-Verlag.

[29] S. Chechik and C. Wulff-Nilsen. Near-optimal light spanners. *ACM Trans. Algorithms*, 14(3):33:1–33:15, 2018.

[30] L. P. Chew. There are planar graphs almost as good as the complete graph. *J. Comput. Syst. Sci.*, 39(2):205–219, 1989.

[31] K. L. Clarkson. Approximation algorithms for shortest path motion planning. In *Proc. 19th ACM Symposium on Theory of Computing (STOC)*, pages 56–65, 1987.

[32] G. Das and G. Narasimhan. A fast algorithm for constructing sparse Euclidean spanners. *International Journal of Computational Geometry & Applications*, 7(04):297–315, 1997.

[33] M. J. Demmer and M. P. Herlihy. The arrow distributed directory protocol. In *Proc. 12th Symposium on Distributed Computing (DISC)*, volume 1499 of *LNCS*, pages 119–133. Springer, 1998.

[34] F. Ebrahimnejad and J. R. Lee. Non-existence of annular separators in geometric graphs. *arXiv preprint arXiv:2107.09790*, 2021.

[35] M. Elkin. Streaming and fully dynamic centralized algorithms for constructing and maintaining sparse spanners. *ACM Trans. Algorithms*, 7(2):20:1–20:17, 2011.

[36] M. Elkin and S. Solomon. Fast constructions of lightweight spanners for general graphs. *ACM Trans. Algorithms*, 12(3):29:1–29:21, 2016.

[37] D. Eppstein, M. T. Goodrich, and N. Mamano. Reactive proximity data structures for graphs. In *Latin American Symposium on Theoretical Informatics*, pages 777–789. Springer, 2018.

[38] D. Eppstein, M. T. Goodrich, and D. Strash. Linear-time algorithms for geometric graphs with sublinearly many edge crossings. *SIAM Journal on Computing*, 39(8):3814–3829, 2010.

[39] D. Eppstein and E. Havvaei. Parameterized leaf power recognition via embedding into graph products. *Algorithmica*, 82(8):2337–2359, 2020.

[40] D. Eppstein and H. Khodabandeh. On the edge crossings of the greedy spanner. In *37th International Symposium on Computational Geometry*, volume 12, page 37, 2021.

[41] D. Eppstein and H. Khodabandeh. Optimal spanners for unit ball graphs in doubling metrics. *arXiv preprint arXiv:2106.15234*, 2021.

[42] P. Erdős. Extremal problems in graph theory. *Theory of Graphs and Its Applications (Proc. Sympos. Smolenice)*, pages 29–36, 1964. see here.

[43] M. Farshi and J. Gudmundsson. Experimental study of geometric t-spanners. *ACM Journal of Experimental Algorithmics*, 14:1.3:1–1.3:29, 2009.

[44] J. Feigenbaum, S. Kannan, A. McGregor, S. Suri, and J. Zhang. Graph distances in the data-stream model. *SIAM J. Comput.*, 38(5):1709–1727, 2008.

[45] A. Filtser. Steiner point removal with distortion $O(\log k)$ using the relaxed-Voronoi algorithm. *SIAM J. Comput.*, 48(2):249–278, 2019.

[46] A. Filtser. Scattering and sparse partitions, and their applications. In *Proc. 47th International Colloquium on Automata, Languages, and Programming (ICALP)*, volume 168 of *LIPIcs*, pages 47:1–47:20. Schloss Dagstuhl, 2020.

[47] A. Filtser, M. Kapralov, and N. Nouri. Graph spanners by sketching in dynamic streams and the simultaneous communication model. In *Proc. 32nd ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1894–1913, 2021.

[48] A. Filtser, R. Krauthgamer, and O. Trabelsi. Relaxed Voronoi: A simple framework for terminal-clustering problems. In *Proc. 2nd Symposium on Simplicity in Algorithms (SOSA)*, volume 69 of *OASICS*, pages 10:1–10:14. Schloss Dagstuhl, 2019.

[49] A. Filtser and S. Solomon. The greedy spanner is existentially optimal. In *Proceedings of the 35th ACM SIGACT-SIGOPS Symposium on Principles of Distributed Computing (PODC)*, pages 9–17. ACM, 2016.

[50] A. Filtser and S. Solomon. The greedy spanner is existentially optimal. *SIAM J. Comput.*, 49(2):429–447, 2020.

[51] J. Fischer and S. Har-Peled. Dynamic well-separated pair decomposition made easy. In *Proc. 17th Canadian Conference on Computational Geometry (CCCG)*, pages 235–238, 2005. see here.

[52] J. Gao, L. J. Guibas, and A. Nguyen. Deformable spanners and applications. *Comput. Geom.*, 35(1-2):2–19, 2006.

[53] M. J. Golin, H. Khodabande, and B. Qin. Non-approximability and polylogarithmic approximations of the single-sink unsplittable and confluent dynamic flow problems. *arXiv preprint arXiv:1709.10307*, 2017.

[54] M. T. Goodrich, S. Gupta, H. Khodabandeh, and P. Matias. How to catch marathon cheaters: New approximation algorithms for tracking paths. In *Workshop on Algorithms and Data Structures*, pages 442–456. Springer, 2021.

[55] L. Gottlieb and L. Roditty. An optimal dynamic spanner for doubling metric spaces. In *Proc. 16th Annual European Symposium on Algorithms (ESA)*, volume 5193 of *LNCS*, pages 478–489. Springer, 2008.

[56] L.-A. Gottlieb. A light metric spanner. In *Proc. 56th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 759–772, 2015.

[57] L.-A. Gottlieb, A. Kontorovich, and R. Krauthgamer. Efficient regression in metric spaces via approximate Lipschitz extension. *IEEE Transactions on Information Theory*, 63(8):4838–4849, 2017.

[58] L.-A. Gottlieb and L. Roditty. An optimal dynamic spanner for doubling metric spaces. In *Proc. 16th European Symposium on Algorithms (ESA)*, volume 5193 of *LNCS*, pages 478–489. Springer, 2008.

[59] E. Grigorescu, Y. Lin, and K. Quanrud. Online directed spanners and Steiner forests. In *Proc. Approximation, Randomization, and Combinatorial Optimization Algorithms and Techniques (APPROX/RANDOM)*, volume 207 of *LIPIcs*, pages 5:1–5:25. Schloss Dagstuhl, 2021.

[60] J. Gudmundsson and C. Knauer. Dilation and detours in geometric networks. In *Handbook of Approximation Algorithms and Metaheuristics*, volume 2. Chapman and Hall/CRC, 2nd edition, 2018. see here.

[61] J. Gudmundsson, C. Levcopoulos, and G. Narasimhan. Fast greedy algorithms for constructing sparse geometric spanners. *SIAM Journal on Computing*, 31(5):1479–1500, 2002.

[62] J. Gudmundsson, C. Levcopoulos, G. Narasimhan, and M. Smid. Approximate distance oracles for geometric spanners. *ACM Transactions on Algorithms (TALG)*, 4(1):1–34, 2008.

[63] A. Gupta. Steiner points in tree metrics don't (really) help. In *Proc. 12th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 220–227, 2001. see here.

[64] A. Gupta, R. Ravi, K. Talwar, and S. W. Umboh. LAST but not least: Online spanners for buy-at-bulk. In P. N. Klein, editor, *Proc. 28th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 589–599, 2017.

[65] M. Hajiaghayi, V. Liaghat, and D. Panigrahi. Online node-weighted steiner forest and extensions via disk paintings. *SIAM J. Comput.*, 46(3):911–935, 2017.

[66] S. Har-Peled. *Geometric Approximation Algorithms*, volume 173 of *Mathematical Surveys and Monographs.* AMS, Providence, RI, 2011. see here.

[67] S. Har-Peled and M. Mendel. Fast construction of nets in low-dimensional metrics and their applications. *SIAM J. Comput.*, 35(5):1148–1184, 2006.

[68] M. Herlihy, S. Tirthapura, and R. Wattenhofer. Competitive concurrent distributed queuing. In *Proc. 20th ACM Symposium on Principles of Distributed Computing (PODC)*, pages 127–133, 2001.

[69] M. Imase and B. M. Waxman. Dynamic Steiner tree problem. *SIAM Journal on Discrete Mathematics*, 4(3):369–384, 1991.

[70] M. Imase and B. M. Waxman. Dynamic steiner tree problem. *SIAM J. Discret. Math.*, 4(3):369–384, 1991.

[71] J. M. Keil. Approximating the complete Euclidean graph. In *Proc. 1st Scandinavian Workshop on Algorithm Theory (SWAT)*, volume 318 of *LNCS*, pages 208–213. Springer, 1988.

[72] R. Krauthgamer and J. R. Lee. Navigating nets: Simple algorithms for proximity search. In *Proc 15th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 798–807, 2004. see here.

[73] H. Le and S. Solomon. Truly optimal Euclidean spanners. In *Proc. 60th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 1078–1100, 2019.

[74] H. Le and S. Solomon. Light Euclidean spanners with Steiner points. In *Proc. 28th European Symposium on Algorithms (ESA)*, volume 173 of *LIPIcs*, pages 67:1–67:22. Schloss Dagstuhl, 2020.

[75] H. Le and S. Solomon. Near-optimal spanners for general graphs in (nearly) linear time. In *Proceedings of the 2022 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 3332–3361. SIAM, 2022.

[76] H. Le and C. Than. Greedy spanners in euclidean spaces admit sublinear separators. In *Proceedings of the 2022 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 3287–3310. SIAM, 2022.

[77] N. Mamano, A. Efrat, D. Eppstein, D. Frishberg, M. Goodrich, S. Kobourov, P. Matias, and V. Polishchuk. Euclidean tsp, motorcycle graphs, and other new applications of nearest-neighbor chains. *arXiv preprint arXiv:1902.06875*, 2019.

[78] A. McGregor. Graph stream algorithms: a survey. *SIGMOD Rec.*, 43(1):9–20, 2014.

[79] J. Naor, D. Panigrahi, and M. Singh. Online node-weighted Steiner tree and related problems. In *Proc. 52nd IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 210–219, 2011.

[80] G. Narasimhan and M. Smid. *Geometric Spanner Networks.* Cambridge University Press, 2007.

[81] D. Peleg and A. A. Schäffer. Graph spanners. *Journal of Graph Theory*, 13(1):99–116, 1989.

[82] D. Peleg and J. D. Ullman. An optimal synchronizer for the hypercube. *SIAM J. Comput.*, 18(4):740–747, 1989.

[83] D. Peleg and E. Upfal. A trade-off between space and efficiency for routing tables. *Journal of the ACM (JACM)*, 36(3):510–530, 1989.

[84] L. Roditty. Fully dynamic geometric spanners. *Algorithmica*, 62(3-4):1073–1087, 2012.

[85] J. Ruppert and R. Seidel. Approximating the d-dimensional complete Euclidean graph. In *Proc. 3rd Canadian Conference on Computational Geometry (CCCG)*, pages 207–210, 1991.

[86] C. Schindelhauer, K. Volbert, and M. Ziegler. Geometric spanners with applications in wireless networks. *Computational Geometry*, 36(3):197–214, 2007.

[87] M. H. Smid. The well-separated pair decomposition and its applications., 2018.

[88] M. H. M. Smid. The well-separated pair decomposition and its applications. In *Handbook of Approximation Algorithms and Metaheuristics*, volume 2. CRC Press, 2nd edition, 2018. see here.

[89] S. W. Umboh. Personal communication, October 2021.

[90] A. C. Yao. Space-time tradeoff for answering range queries (extended abstract). In *Proc. 14th Annual ACM Symposium on Theory of Computing (STOC)*, pages 128–136, 1982.

[91] A. H. A. Zargari, S. A. H. Aqajari, H. Khodabandeh, A. M. Rahmani, and F. Kurdahi. An accurate non-accelerometer-based ppg motion artifact removal technique using cyclegan. *arXiv preprint arXiv:2106.11512*, 2021.