

Towards Instance-Optimal Euclidean Spanners

Hung Le^{*} Shay Solomon[†] Cuong Than[‡] Csaba D. Tóth[§] Tianyi Zhang[¶]

Abstract

Euclidean spanners are important geometric objects that have been extensively studied since the 1980s. The two most basic “compactness” measures of a Euclidean spanner E ¹ are the size (number of edges) $|E|$ and the weight (sum of edge weights) $\|E\|$. The state-of-the-art constructions of Euclidean $(1 + \epsilon)$ -spanners in $\mathbb{R}^d$ have $O_d(n \cdot \epsilon^{-d+1})$ edges (or *sparsity* $O_d(\epsilon^{-d+1})$) and weight $O_d(\epsilon^{-d} \log \epsilon^{-1}) \cdot \|E_{\text{mst}}\|$ (or *lightness* $O_d(\epsilon^{-d} \log \epsilon^{-1})$); here O_d suppresses a factor of $d^{O(d)}$ and $\|E_{\text{mst}}\|$ denotes the weight of a minimum spanning tree of the input point set.

Importantly, these two upper bounds are (near-)optimal (up to the $d^{O(d)}$ factor and disregarding the factor of $\log(\epsilon^{-1})$ in the lightness bound) for some *extremal instances* [Le and Solomon, 2019], and therefore they are (near-)*optimal in an existential sense*. Moreover, both these upper bounds are attained by the same construction—the classic *greedy spanner*, whose sparsity and lightness are not only existentially optimal, but they also significantly outperform those of any other Euclidean spanner construction studied in an experimental study by [Farshi-Gudmundsson, 2009] for various practical point sets in the plane. This raises the natural question of whether the greedy spanner is (near-) optimal for *any point set instance*?

Motivated by this question, we initiate the study of *instance optimal Euclidean spanners*. Our results are two-fold.

- Rather surprisingly (given the aforementioned experimental study), we demonstrate that the *greedy spanner is far from being instance optimal*, even when allowing its stretch to grow. More concretely, we design two hard instances of point sets in the plane, where the greedy $(1 + x\epsilon)$ -spanner (for basically any parameter $x \geq 1$) has $\Omega_x(\epsilon^{-1/2}) \cdot |E_{\text{spa}}|$ edges and weight $\Omega_x(\epsilon^{-1}) \cdot \|E_{\text{light}}\|$, where E_{spa} and E_{light} denote the per-instance sparsest and lightest $(1 + \epsilon)$ -spanners, respectively, and the Ω_x notation suppresses a polynomial dependence on $1/x$.
- As our main contribution, we design a new construction of Euclidean spanners, which is inherently different from known constructions, achieving the following bounds: a stretch of $1 + \epsilon \cdot 2^{O(\log^*(d/\epsilon))}$ with $O(1) \cdot |E_{\text{spa}}|$ edges and weight $O(1) \cdot \|E_{\text{light}}\|$. In other words, we show that a slight increase to the stretch suffices for obtaining *instance optimality up to an absolute constant for both sparsity and lightness*. Remarkably, there is only a log-star dependence on the dimension in the stretch, and there is no dependence on it whatsoever in the number of edges and weight. In general, for any integer $k \geq 1$, we can construct a Euclidean spanner in $\mathbb{R}^d$ of stretch $1 + \epsilon \cdot 2^{O(k)}$ with $O(\log^{(k)}(\epsilon^{-1}) + \log^{(k-1)}(d)) \cdot |E_{\text{spa}}|$ edges and weight $O(\log^{(k)}(\epsilon^{-1}) + \log^{(k-1)}(d)) \cdot \|E_{\text{light}}\|$, where $\log^{(k)}$ denotes the k -iterated logarithm.

^{*}University of Massachusetts Amherst, hungle@cs.umass.edu

[†]Tel Aviv University, solo.shay@gmail.com

[‡]University of Massachusetts Amherst, cthan@umass.edu

[§]California State University Northridge and Tufts University, csaba.toth@csun.edu

[¶]Tel Aviv University, tianyiz21@tauex.tau.ac.il

¹We shall identify a graph $H = (X, E)$ with its edge set E . All edge weights are given by the Euclidean distances.

Contents

1	Introduction	1
1.1	Our Contribution	4
1.2	Related Work on Approximate Spanners in General Graphs	6
2	Technical Overview	6
3	Preliminaries	12
4	Instance-Optimal Euclidean Spanners	13
4.1	A Greedy-Pruning Algorithm	13
4.2	Stretch Analysis	15
4.3	Sparsity Analysis	17
4.4	Lightness Analysis	22
5	Fast Implementation	32
5.1	Net-Tree Spanner	32
5.2	Fast Implementation	32
5.3	Analysis	34
5.3.1	Stretch	34
5.3.2	Sparsity	35
5.3.3	Lightness	37
5.3.4	Running Time	38
6	Lower Bounds for the Greedy Spanner	40
6.1	Sparsity Lower Bounds	40
6.2	Lightness Lower Bounds	46

1 Introduction

A *Euclidean t -spanner* of a set of point $X \subset \mathbb{R}^d$ is an edge-weighted graph $H = (X, E, \mathbf{w})$ having X as the vertex set, such that the weight $\mathbf{w}(x, y)$ of any edge $xy \in E$ is given by the Euclidean distance $\|xy\|$ between the endpoints x and y , and for any $x, y \in X$, there is a path in H of *weight* at most $t \cdot \|xy\|$; such a path is called a *t -spanner path*. The parameter t is called the *stretch* of the spanner. The most important “compactness” measure of a Euclidean spanner is its number of edges or its *sparsity*, which is the ratio of the number of edges of the spanner to the number of points. Another basic compactness measure of a Euclidean spanner is its weight (i.e., total edge weight) or its *lightness*, which is the ratio of its weight to the weight of a Euclidean minimum spanning tree (MST).

The pioneering work by Chew [Che86] showed that one could construct a spanner with *constant* stretch and *constant* sparsity for any point set in the plane. Since then, Euclidean spanners have been extensively and intensively studied. Over more than three decades, Chew’s result has been strengthened and generalized, culminating with the following result: For any parameter $\epsilon \in (0, 1)$ and for any finite point set in $\mathbb{R}^d$, one can construct a $(1 + \epsilon)$ -spanner with sparsity $O_d(\epsilon^{-d+1})$ and/or lightness $O_d(\epsilon^{-d} \log(\epsilon^{-1}))$; here O_d suppresses a factor of $d^{O(d)}$. The sparsity upper bound of $O_d(\epsilon^{-d+1})$ is realized by various classic constructions, such as the Theta-graph [Cla87, Kei88, RS91], Yao graph [Yao82], and the greedy spanner [ADD⁺93]; the upper bound arguments, which were given already in the 90s, are short and simple. (We refer the readers to the book by Narasimhan and Smid [NS07] for a comprehensive coverage of spanner constructions that achieve a similar sparsity bound.) On the other hand, the lightness upper bound of $O_d(\epsilon^{-d} \log(\epsilon^{-1}))$ is much more complex, and it was proven rather recently by Le and Solomon [LS22]; in contrast to the sparsity upper bound, the lightness upper bound was proved only for the greedy spanner. In the same paper [LS22], it was shown that, for any $d \geq 1$ and for any $n = \Omega_d(\epsilon^{-d} \log(\epsilon^{-1}))$, there is a set X of n points in $\mathbb{R}^d$ such that any $(1 + \epsilon)$ -spanner for X must have sparsity $\Omega_d(\epsilon^{-d+1})$ and lightness $\Omega_d(\epsilon^{-d})$. Their results imply that the greedy spanner is *existentially optimal* for *both* sparsity and lightness and the Theta- and Yao-graphs are existentially optimal just for sparsity; by existentially optimal we mean (near-)optimal in an existential sense to be formally defined below.

Existential optimality. Let A be a polynomial-time algorithm that takes a point set $X \in \mathbb{R}^d$ and $\epsilon \in (0, 1)$ as input, and outputs a Euclidean $(1 + \epsilon)$ -spanner for X , denoted by $A(X, \epsilon)$. Let $\mathbf{C}(G)$ be a cost function imposed on a graph G . For our purposes, function $\mathbf{C}$ either counts the number of edges (corresponding to sparsity) or the total edge weight (corresponding to lightness). We say that algorithm A is *existentially optimal for $\mathbf{C}$ with optimality ratio κ* (for the Euclidean space $\mathbb{R}^d$) if, for every positive integer n , there exists an n -point set P_n in $\mathbb{R}^d$ such that

$$\text{for any } n\text{-point set } X_n \in \mathbb{R}^d, \text{ we have } \mathbf{C}(A(X_n, \epsilon)) \leq \kappa \cdot \text{opt}_{\mathbf{C}}(P_n, \epsilon),$$

where $\text{opt}_{\mathbf{C}}(P_n, \epsilon)$ is the cost of the optimal $(1 + \epsilon)$ -spanner for P_n under $\mathbf{C}$. Here the point set $P = P_n$ serves as a “hard” or “extreme” instance; that is, for existential optimality of algorithm A , it suffices to show the existence of a single hard instance (for any n), where the cost of A on any n -point set is no worse (by more than a factor of κ) than the *optimal* cost on the hard instance P . The notion of existential optimality in the context of graph spanners was explicitly formulated by Filtser and Solomon [FS20], though the general idea of existential optimality was implicitly used long before. Here, we tailor their definition to the Euclidean space $\mathbb{R}^d$.

Ideally, we would like to design an existentially optimal algorithm A with $\kappa = 1$, but this is too much to ask for: No known (polynomial time) algorithm in the spanner literature is existentially optimal with $\kappa = 1$ (or even close to that). The known constructions are existentially optimal with

optimality ratio $\kappa = d^{O(d)}$ or $\kappa = 2^{O(d)} \cdot O_\epsilon(1)$, where $O_\epsilon(1)$ is a constant that depends only on ϵ ; thus the optimality ratio of all known constructions is a constant (and typically a large one) that depends at least exponentially on the dimension d . We shall henceforth say that algorithm A is *existentially optimal for $\mathbf{C}$* if the optimality ratio κ is a constant that depends only on the dimension d ; importantly, κ *must not* depend on ϵ and on the size n of the input point sets.

The hard instance used by Le and Solomon [LS23], denoted here by S_n , is basically a set of n evenly spaced points on the boundary of a d -dimensional unit sphere.² Let us start with the sparsity (so that the function $\mathbf{C}$ counts the number of edges). Le and Solomon [LS23] showed that $\text{opt}_{\mathbf{C}}(S_n, \epsilon) \geq \alpha_d \cdot \epsilon^{-d+1} |S_n|$ for some $\alpha_d = 2^{-O(d)}$. For the Theta-graph construction, denoted here by Θ , it is known [Cla87, Kei88, RS91] that for every n -point set $X \in \mathbb{R}^d$, we have $\mathbf{C}(\Theta(X_n, \epsilon)) \leq \beta_d \cdot \epsilon^{-d+1} |X_n|$ for some $\beta_d = 2^{O(d)}$. Thus, for every n -point set $X \in \mathbb{R}^d$:

$$\mathbf{C}(\Theta(X_n, \epsilon)) \leq \beta_d \cdot (1/\alpha_d) \cdot \text{opt}_{\mathbf{C}}(P_n, \epsilon) = 2^{O(d)} \text{opt}_{\mathbf{C}}(P_n, \epsilon),$$

implying that Theta-graphs are existentially optimal for sparsity. By the same argument, both Yao-graphs and greedy spanners are existentially optimal for sparsity [ADD⁺93]. Next, for lightness (now the function $\mathbf{C}$ counts the weight), Le and Solomon [LS23] showed that $\text{opt}_{\mathbf{C}}(S_n, \epsilon) \geq \alpha_d \cdot \epsilon^{-d} \|\text{MST}(S_n)\|$ for some $\alpha_d = 2^{-O(d)}$, where $\text{MST}(S_n)$ is a Euclidean minimum spanning tree of S_n . Combining this lower bound with the lightness upper bound of the greedy spanner by [LS23], it follows that the greedy $(1+\epsilon)$ -spanner algorithm is existentially optimal for lightness with *optimality ratio* $2^{O(d)} \log(\epsilon^{-1})$. It is worth noting that removing the $\log(\epsilon^{-1})$ factor in the lightness optimality ratio of the greedy algorithm remains an open problem.

Thus, the hard instance by Le and Solomon [LS23] allows one to “declare” that their spanner construction, if achieving $O_d(\epsilon^{-d+1})$ sparsity and/or $O_d(\epsilon^{-d})$ lightness, is (existentially) optimal and hence they could redirect their effort on optimizing other properties. However, existential optimality, while interesting in its own right, is a rather weak notion of optimality: it only requires that the algorithm has to perform as well as the optimal spanners (up to a factor of κ) for *the hard(est) instance*. The hard instance might be impractical; indeed, this is the case with the aforementioned hard instance by Le and Solomon [LS23] (basically a set of evenly spaced points on the boundary of a sphere), which is very unlikely to appear in practice. On the other hand, on *practical point sets*, an existentially optimal algorithm A might perform poorly compared to the optimal spanner. In fact, it is conceivable that for *a wide range of point sets*, including various random distributions of points, the spanners produced by an existentially optimal algorithm A have more edges than the *instance optimal spanners* by a factor of $\Omega_d(\epsilon^{-d+1})$. In extreme cases where ϵ is chosen so that $\epsilon^{-d+1} = \Theta_d(n)$, A could end up having $\Theta_d(n^2)$ edges while an optimal spanner has only $O_d(n)$ edges. Indeed, the experimental work by Farshi and Gudmundsson [FG09] showed that several existentially optimal algorithms (with respect to sparsity), including the Theta- and Yao-graphs, produce spanners with a much larger number of edges than the optimal spanners³. This naturally calls for a focus on a stronger notion of optimality, namely instance optimal spanners.

Instance optimality. The notion of instance optimality was introduced by Fargin, Lotem, and Naor [FLN03] in the problem of choosing top k items in sorted lists. We adapt this notion in our context as follows. We say that a polynomial-time algorithm A is *instance optimal for $\mathbf{C}$ with optimality ratio κ* if for every finite point set $X \in \mathbb{R}^d$, it holds that

²For sparsity, one has to use multiple vertex-disjoint copies of the sphere that are well-separated from each other.

³In [FG09], the optimal spanners (in terms of sparsity and other cost functions) were not computed explicitly; however for every instance, one can take the best spanners over the collection of algorithms studied by Farshi and Gudmundsson as an upper bound for the cost of the optimal spanners.

$$\mathbf{C}(A(X, \epsilon)) \leq \kappa \cdot \text{opt}_{\mathbf{C}}(X, \epsilon),$$

where $\text{opt}_{\mathbf{C}}(X, \epsilon)$ is the cost of the optimal $(1 + \epsilon)$ -spanner for X under $\mathbf{C}$.

An instance optimal algorithm with ratio κ is, in fact, an *approximation algorithm* with an approximation factor κ . The reason we chose to use the terminology of instance optimality rather than that of approximation is that the existential bounds in low-dimensional Euclidean spaces are already “good” in the sense that they are independent of the metric size, namely $\epsilon^{-O(d)}$ for stretch $1 + \epsilon$ and dimension d . This stands in contrast to the existential bounds in general graphs that depend on the graph size, and for which there is a long line of influential work on approximation algorithms from both the upper and lower bounds sides; see Section 1.2 for more detail. We stress that improving bounds that are independent of the metric size is a completely different challenge, requiring new tools and techniques. By using the “instance-optimality” terminology, we wish to deviate from the line of work on approximation algorithms in general graphs and put the main focus on (1) the side of the upper bound (i.e., the “optimality” aspect) and (2) how the instance-optimal bounds compare to the existential-optimal bounds. Moreover, we hope that our work will initiate a systematic study on instance-optimal spanners in other graph families for which the existential bounds are independent of the graph size, such as unit-disk graphs, planar graphs, and bounded treewidth graphs.

By definition, an instance optimal algorithm is also existentially optimal, but the converse is of course not true; indeed, designing an instance optimal algorithm appears to be significantly harder. First, designing an instance optimal algorithm with ratio $\kappa = 1$ is NP-hard for both sparsity [KK06, GKK⁺10] and lightness [CCY13]. Thus, to get a polynomial time algorithm, we shall allow the ratio κ to grow beyond 1, to any constant that depends only on the dimension d ; we refer to such an algorithm as *instance optimal*. Even under this relaxation, no such algorithm is known to date, even in the plane! This leads naturally to a fundamental question in Euclidean spanners, summarized below.

Question 1.1. *Can one design a polynomial-time instance optimal $(1 + \epsilon)$ -spanner algorithm with a ratio $\kappa = O_d(1)$ for sparsity and/or lightness?*

The greedy spanner. A natural candidate of an instance optimal algorithm is the (path) greedy spanner [ADD⁺93]. As mentioned above, in the Euclidean space $\mathbb{R}^d$, it was shown to be *existentially optimal* for sparsity, and also for lightness up to a factor of $\log(\epsilon^{-1})$ [LS22]. In addition, Filtser and Solomon [FS20] showed that greedy spanners are existentially optimal for very broad classes of graphs: those that are closed under edge deletions, which include general graphs and minor-closed families. They also showed that greedy spanners (and also an approximate version of the greedy spanner [DN97, GLN02]) are existentially (near-)optimal for both sparsity and lightness in the family of *doubling metrics*, which is wider than that of Euclidean spaces. Experimental results [SZ04, FG09, CS22] also showed that the greedy spanners achieve the best quality in multiple aspects.

For general graphs, it was known that the approximation factor of the greedy spanners is $\Theta(n)$ for stretch $t < 3$ [FS20, ABS⁺20] while an algorithm with a better approximation was known [KP94]. However, for higher stretch ($t \geq 5$), the greedy algorithm achieves the best-known approximation ratio. Furthermore, there are works showing that the greedy spanner provides rather good *bicriteria approximation* algorithms in general graphs [GW22, Won23, BBGW24] for a related problem, called *minimum dilation graph augmentation*: augmenting a graph as few edges as possible to reduce the dilation (a.k.a. stretch).

Consequently, it appears that a preponderance of work on the greedy spanners all point in the same direction, i.e., to a *conjecture that the greedy algorithm gives a good bicriteria instance optimal spanner*. In this work, we demonstrate that the *greedy spanner is far from being bicriteria instance optimal* for points in $\mathbb{R}^d$. More concretely, we design two hard instances of point sets in the plane, where the greedy $(1 + x\epsilon)$ -spanner (for basically any parameter $x \geq 1$) has $\Omega_x(\epsilon^{-1/2}) \cdot |E_{\text{spa}}|$ edges and weight $\Omega_x(\epsilon^{-1}) \cdot \|E_{\text{light}}\|$, where E_{spa} and E_{light} denote the per-instance sparsest and lightest $(1 + \epsilon)$ -spanners, respectively, and the Ω_x notation suppresses a polynomial dependence on $1/x$.

Bicriteria instance optimality. Given that the greedy spanner is far from being instance optimal, even in the plane, and *even when allowing its stretch to grow from $1 + \epsilon$ to $1 + x\epsilon$ for $x \geq 1$* , and given that the greedy spanner appears to outperform any other known spanner construction in terms of both sparsity and lightness, the natural conclusion is that a *new spanner construction is in order*. In light of our hardness result for the greedy spanner, it seems acceptable to allow the stretch to increase from $1 + \epsilon$ to $1 + x\epsilon$, for some reasonably small x . This leads to the following question: Could standard techniques in geometric optimization, such as Arora’s technique [Aro98], be applied to construct instance optimal spanners? Arora’s technique has been instrumental in solving problems such as the Euclidean TSP and Steiner tree. In our problem, however, the major difficulty is that we aim at optimizing the cost of the spanner *while guaranteeing a stretch bound of $1 + \epsilon$* . Alas, Arora’s technique, as well as other known techniques, are not suitable for achieving *both criteria*. Thus, it appears that *a new technique for achieving both criteria is in order*.

The above discussion motivates us to consider *bicriteria instance optimal spanners*: We say that an algorithm A is *(c, κ) -instance optimal* if for every point set $X \in \mathbb{R}^d$, $A(X, \epsilon)$ is a $(1 + c \cdot \epsilon)$ -spanner for X , and $\mathbf{C}(A(X, \epsilon)) \leq \kappa \cdot \mathbf{opt}_{\mathbf{C}}(X, \epsilon)$.

Question 1.2. *Can one design a polynomial-time (c, κ) -instance optimal spanner algorithm with c and κ both bounded by $O_d(1)$ (independent of ϵ) for sparsity and/or lightness?*

While Question 1.2 asks for constants c and κ (depending only on the dimension d), what was previously known is embarrassingly little. Even in the basic setting of the Euclidean plane, the only positive result is a direct corollary of existentially optimal spanners, which gives a bicriteria (c, κ) -instance optimal spanner algorithm with $c = O(1)$ and $\kappa = O(\epsilon^{-1})$ for sparsity and $\kappa = O(\epsilon^{-2})$ for lightness⁴. To the best of our knowledge, no prior result achieves sublinear (respectively, subquadratic) dependence of κ on ϵ^{-1} for sparsity (resp., lightness) in the Euclidean plane. On the other hand, there is a lot of work on approximating spanners in general graphs, which we will review in more detail in Section 1.2. A short takeaway is that the approximation factors in this setting depend on n (the number of vertices), while in our setting, the approximation factor is independent of n (the number of points). Therefore, the techniques for general graphs do not seem applicable to $\mathbb{R}^d$.

1.1 Our Contribution

Hard instances. We first construct point sets in $\mathbb{R}^2$ for which the *greedy* $(1 + \epsilon)$ -spanner is far from being instance optimal in terms of sparsity or lightness, even if we relax the stretch from $1 + \epsilon$

⁴Note that in the regime of parameters derived from existentially optimal spanners, having $c = 1$ is the same as having $c = O(1)$ since one can apply the standard scaling trick: $\epsilon \leftarrow \epsilon/c$. Scaling reduces the stretch from $1 + c\epsilon$ to $1 + \epsilon$ while adding a factor of c for sparsity and c^2 for lightness to κ . The same trick, however, does not work for instance optimal spanners: if one scales ϵ to ϵ/c , then one now essentially compares against the optimal $(1 + \epsilon/c)$ -spanner instead of the optimal $(1 + \epsilon)$ -spanner.

to a larger value $1 + x\epsilon$.

Theorem 1.1 (Sparsity lower bound for greedy). *For every sufficiently small $\epsilon > 0$ and $1 \leq x \leq o(\epsilon^{-1/3})$, there exists a finite set $S \subset \mathbb{R}^2$ such that*

$$|E_{\text{gr}(x)}| \geq \Omega\left(\frac{\epsilon^{-1/2}}{x^{3/2}}\right) \cdot |E_{\text{spa}}|,$$

where $E_{\text{gr}(x)}$ is the edge set of the greedy $(1 + x\epsilon)$ -spanner, and E_{spa} is the edge set of a sparsest $(1 + \epsilon)$ -spanner for S .

Theorem 1.2 (Lightness lower bound for greedy). *For every sufficiently small $\epsilon > 0$ and $x \in [2, \epsilon^{-1/2}/48]$, there exists a finite set $S \subset \mathbb{R}^2$ such that*

$$\|E_{\text{gr}(x)}\| \geq \Omega\left(\frac{\epsilon^{-1}}{x^2 \cdot \log x}\right) \cdot \|E_{\text{light}}\|,$$

where $E_{\text{gr}(x)}$ is the edge set of the greedy $(1 + x\epsilon)$ -spanner, and E_{light} is the edge set of a minimum-weight $(1 + \epsilon)$ -spanner for S .

Main results. Our main contribution is on the algorithmic front: we design a bicriteria (c, κ) -instance optimal spanner algorithm with $c = 2^{O(\log^*(d/\epsilon))}$ and $\kappa = O(1)$, thus resolving Question 1.2 up to the exponential log-star term $2^{O(\log^*(d/\epsilon))}$, which is bounded by $O(\log^{(k)})(d/\epsilon)$ for *any constant* k ; here and throughout $\log^{(k)}$ denotes the k -iterated logarithm function (i.e., $\log^{(1)}(x) = \log(x)$ and $\log^{(k)}(x) = \log(\log^{(k-1)}(x))$ for any integer $k \geq 2$). This result is obtained as a direct corollary of the following general theorem.

Theorem 1.3 (General tradeoff upper bound). *Let $X \subset \mathbb{R}^d$ be any set of n points, and $k \geq 1$ an integer. For any $\epsilon \in (0, 1)$, there is an algorithm that returns an Euclidean $(1 + 2^{O(k)}\epsilon)$ -spanner $H = (X, E)$ for X , such that*

$$\begin{aligned} |E| &= O\left(\log^{(k)}(\epsilon^{-1}) + \log^{(k-1)}(d)\right) |E_{\text{spa}}| \quad \text{and} \\ \|E\| &= O\left(\log^{(k)}(\epsilon^{-1}) + \log^{(k-1)}(d)\right) \|E_{\text{light}}\|, \end{aligned}$$

where $E_{\text{spa}}, E_{\text{light}} \in \binom{X}{2}$ are the edge sets of the optimal $(1 + \epsilon)$ -spanners of X for sparsity and lightness, respectively. In other words, our algorithm is bicriteria $(2^{O(k)}, \log^{(k)}(\epsilon^{-1}) + \log^{(k-1)}(d))$ -instance optimal for both sparsity and lightness.

Furthermore, our algorithm can be implemented in $\epsilon^{-O(d)} n \log^2(n)$ time.

We now highlight the two extreme points on the tradeoff curve. First, by setting $k = O(1)$ (for any constant) in Theorem 1.3, we obtain an $(O(1), \log^{(k)}(\epsilon^{-1}) + \log^{(k-1)}(d))$ -instance optimal spanner algorithm. This result is also interesting from the perspective of *existential optimality*. By scaling $\epsilon \leftarrow \epsilon/c$, we get an existentially optimal algorithm for both sparsity and lightness with ratio $\kappa = O(\log^{(k)}(\epsilon^{-1}) + \log^{(k-1)}(d))$ for any constant k . Recall that all known existentially optimal spanner algorithms have an *exponential dependence on d* in the optimality ratio or the stretch blow-up, while we achieve a sublogarithmic dependence on d . In particular, recall that the state-of-the-art existential optimality ratio for *lightness* is $O_d(\log(\epsilon^{-1})) = d^{O(d)} \cdot \log(\epsilon^{-1})$; our result *improves exponentially* both the dependence on ϵ^{-1} and the dependence on d .

By setting $k = \log^*(d/\epsilon)$ in Theorem 1.3, we obtain the following main corollary of the general tradeoff. Remarkably, our optimality ratio κ in Corollary 1.1 does not depend on the dimension d , while our stretch blow-up c only depends *sublogarithmically* on d .

Corollary 1.1 (Almost instance optimality). *There is an algorithm for constructing spanners that are (c, κ) -instance optimal for both sparsity and lightness, where $c = 2^{O(\log^*(d/\epsilon))}$ and $\kappa = O(1)$. Furthermore, our algorithm can be implemented in $\epsilon^{-O(d)} n \log^2(n)$ time.*

Remark. An important feature of the classic greedy spanner algorithm is that it is existentially optimal with respect to both sparsity and lightness. Our spanner algorithms (provided by Theorem 1.3 and Corollary 1.1) also have this feature, but in the stronger sense of instance optimality: it *simultaneously approximates both the sparsest spanner and the lightest spanner*. We note that for the same point set X and stretch factor, the sparsest spanner for X and the lightest spanner could be completely different. In fact, the sparsest spanner could have a huge lightness while the lightest spanner could have a huge sparsity. Thus, a priori, it is unclear whether there exists an algorithm that is instance optimal with respect to *both* sparsity and lightness.

1.2 Related Work on Approximate Spanners in General Graphs

There is a long line of work on approximating minimum t -spanners of both undirected and directed graphs. We briefly review the current best results in these regimes. For undirected graphs, the greedy algorithm [ADD⁺93] provides a t -spanner with approximation ratio $n^{2/(t+1)}$ for odd t and $n^{2/t}$ for even t . Dinitz, Kortsarz, and Raz [DKR15] showed that for any $t \geq 3$ and for any constant $\epsilon > 0$, there is no polynomial-time algorithm approximating a t -spanner with ratio better than $2^{\log n^{1-\epsilon}/t}$ assuming $NP \not\subseteq BPTIME(2^{\text{polylog}(n)})$. For $t = 2$, Kortsarz and Peleg [KP94] (see also [EP01], [DK11]) designed an algorithm with approximation ratio $O(\log n)$, matching the lower bound given by Kortsarz [Kor01] (assuming $P \neq NP$). For $t = 3$, Berman, Bhattacharyya, Makarychev, Raskhodnikova and Yaroslavtsev [BBM⁺11] achieved approximation ratio $\tilde{O}(n^{1/3})$. Dinitz and Zhang [DZ16] obtained the same approximation ratio for stretch $t = 4$. For $t \geq 5$, the approximation ratio achieved by the greedy algorithm remains the state-of-the-art.

For directed graphs, Dinitz and Krauthgamer [DK11] gave an $\tilde{O}(\sqrt{n})$ -approximation algorithm when $t = 3$. The approximation ratio for $t = 3$ was later improved to $O(n^{1/3} \log n)$ for the unit-weight case [BBM⁺11]. For general t , there have been significant efforts to improve the approximation ratio. Bhattacharyya, Grigorescu, Jung, Raskhodnikova, and Woodruff [BGJ⁺12] provided a $\tilde{O}(n^{1-1/t})$ -approximation algorithm for directed graphs for $t > 2$. Berman, Raskhodnikova, and Ruan [BRR10] improved the approximation ratio to $\tilde{O}(t \cdot n^{1-\frac{1}{\lceil t/2 \rceil}})$. For $t > 3$, Dinitz and Krauthgamer [DK11] achieved the approximation ratio of $\tilde{O}(n^{2/3})$. Berman, Bhattacharyya, Makarychev, Raskhodnikova and Yaroslavtsev [BBM⁺11] later improved the approximation ratio to $O(\sqrt{n} \log n)$.

2 Technical Overview

Our construction and sparsity analysis. Let us begin with analyzing the sparsity of an arbitrary $(1 + \delta)$ -spanner (X, E) against a sparsest $(1 + \epsilon)$ -spanner (X, E_{spa}) ; one can think of the parameter δ as $O(\epsilon)$, though in practice it is a parameter used by our algorithm that starts at around ϵ and ultimately grows to $O(\epsilon)$ and even beyond $O(\epsilon)$. A basic approach is to *charge* the edges in E to edges in E_{spa} . Our first idea is to design a *fractional* charging scheme, rather than an integral one. Taking any edge $(s, t) \in E$, since (X, E_{spa}) is a $(1 + \epsilon)$ -spanner, there exists a path π in (X, E_{spa}) between s and t such that $\|\pi\| \leq (1 + \epsilon)\|st\|$. Thus, we can charge the edge $(s, t) \in E$ to each edge $e \in E_{\text{spa}}$ along the path π with a *fractional cost* of $\|e\|/\|st\|$, meaning that the sum of fractional costs of these edges amounts to at least 1 (and also at most $1 + \epsilon$), thus the total

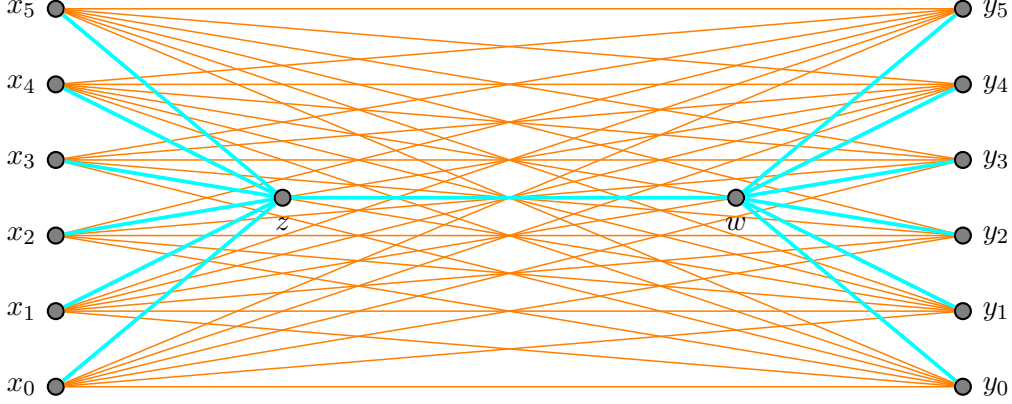


Figure 1: In this example, a spanner might include the orange edges, while the optimal spanner takes the blue edges. Then the middle blue edge would receive a large amount of charges from the orange edges.

fractional costs of all edges in E_{spa} is at least $|E|$. Therefore, if we could argue that the fractional costs received by any edge $e \in E_{\text{spa}}$ is at most λ , that would directly imply that $|E| \leq \lambda \cdot |E_{\text{spa}}|$.

However, such a charging scheme by itself is insufficient, unless it is accompanied with a “good” spanner. Alas, there are hard instances for which the known spanner constructions fail. As a (simplistic) example, consider a point set $X \in \mathbb{R}^2$ that contains two point sets $\{x_i\}_{0 \leq i \leq 1/\sqrt{\epsilon}}$, $\{y_i\}_{0 \leq i \leq 1/\sqrt{\epsilon}}$, where $x_i = (0, i \cdot \epsilon)$, $y_i = (10, i \cdot \epsilon)$, as well as a pair of middle points $z = (3, \frac{\sqrt{\epsilon}}{2})$, $w = (3, \frac{\sqrt{\epsilon}}{2})$. See Figure 1 for an illustration. Our spanner could wastefully include edges $x_i y_j$ for all i, j , creating a bi-clique between $\{x_i\}_{0 \leq i \leq 1/\sqrt{\epsilon}}$ and $\{y_i\}_{0 \leq i \leq 1/\sqrt{\epsilon}}$; we will show later on that the greedy spanner exhibits this kind of wasteful behavior (on more subtle instances). On the other hand, the instance optimal $(1 + \epsilon)$ -spanner for this point set will only include edges $x_i z, zw, w y_i, \forall i$. Getting back to the aforementioned charging scheme, we see that all bi-clique edges would charge to the same middle edge zw , and so this charging scheme would inevitably fail. The main problem here is not the charging scheme, but rather the wasteful behavior of known spanner constructions.

To outperform the known spanner constructions on hard instances, we design a new spanner algorithm, which deviates significantly from the known constructions. Our construction is guided by a novel charging scheme; we build on the basic idea of fractional charging as explained before, but in a much more nuanced way, which takes into account the angles formed by the edges, the locations of the endpoints, and other geometric parameters. To keep this technical overview simple, we will not go into most details of the charging scheme; refer to Section 4.3 for the full details.

Start with an arbitrary $(1 + \delta)$ -spanner. To improve sparsity (and lightness), our basic strategy is to look for *helper edges*, such as edges zw in the example above, add them to our spanner, and then *prune* unnecessary edges whose distances are already well-preserved by zw together with other existing shorter edges in the spanner. More specifically, we will go over all edges $st \in E$ of the original spanner in a non-decreasing order of their lengths (i.e., weights), and look for helper edges zw in the $(1 + \epsilon)\|st\|$ -ellipsoid around edge st that satisfy the following two properties:

1. $\|zw\| \geq \Omega(\|st\|)$; and
2. both z and w are bounded away from the endpoints s and t ; that is, $\|sz\|, \|wt\| \geq \Omega(\|st\|)$.

If such a helper edge zw exists, then we add it to E . The key observation is that any other edge $s't' \in E$ with roughly the same length as st and charging to common edges as st can now be pruned

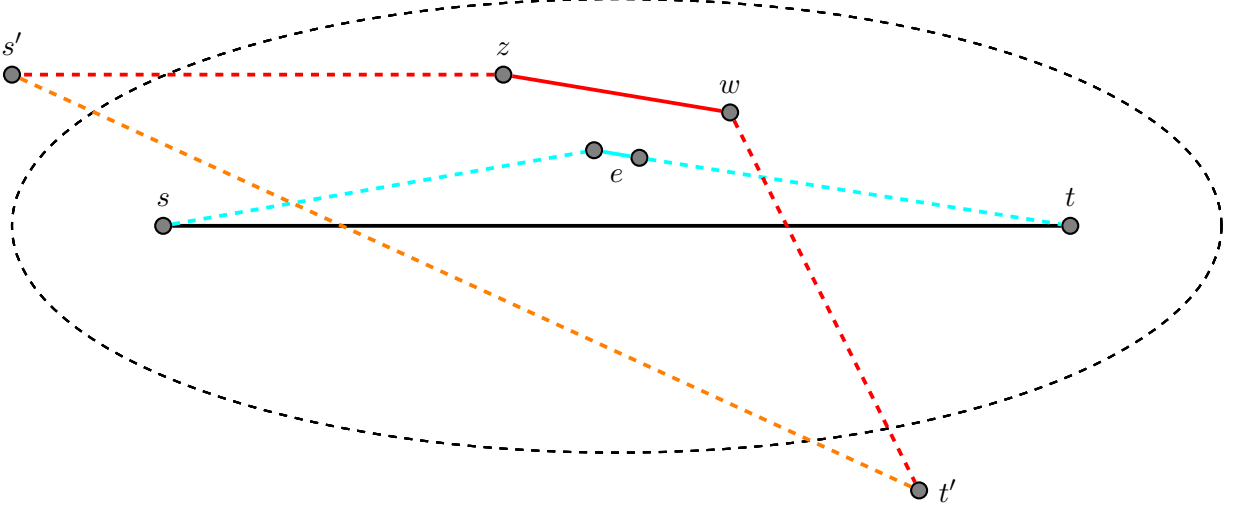


Figure 2: The blue path between s and t is a $(1 + \epsilon)$ -spanner path π in (X, E_{spa}) . The solid blue edge e is an edge of π , so it receives a fractional charge of $\|e\|/\|st\|$ from st . When processing edge st , we find a helper edge zw in the $(1 + \epsilon)\|st\|$ -ellipsoid around st . After adding the helper edge zw to E , we are able to prune other spanner edges $s't'$ which are charging to e , as well, because we can now connect s' and t' using the red path in (X, E) .

from E . Indeed, due to our charging scheme, if $s't'$ and st charge to the same edge, it means that these two edges should be similar in a strong geometric sense, which allows us to reason that edge zw could serve as a helper edge for $s't'$, as well. Consequently, by adding a single helper edge zw while processing edge st , we are able to prune away from our spanner all edges that are “similar” to edge st , which leads to a significant saving. See Figure 2 for an illustration. So far, we have only discussed the pruning of edges $s't'$ that are of roughly the same length as st . We then generalize the above insight to show that for *every length scale* and for every edge e charged by st , at most one edge from that length scale may charge to edge e . Since the fractional charge of edge $s't'$ to edge e , namely $\|e\|/\|s't'\|$, decays with the length of $s't'$, we can bound the total contribution of all such edges $s't'$ to the fractional cost of edge e , over all length scales, by a geometric sum.

This strategy of using helper edges zw as described above is effective when helper edges exist; *alas, they do not always exist*. When helper edges do not exist, we will show that there must exist an edge e along the $(1 + \epsilon)$ -spanner path π in E_{spa} such that $\|e\| \geq \Omega(\|st\|)$; that is, π is making at least one long stride at some point when connecting s and t . In this case, we will charge the edge st only to this single long edge e . However, since this long edge e does not qualify as a helper edge (property 2 of a helper edge does not hold), we cannot apply the aforementioned argument, and it is now possible that many different edges $s't' \in E$ of length $\Theta(\|e\|)$ will charge to the same edge e . In this case, we will apply a more aggressive pruning, which greedily finds all edges e whose addition to E could prune sufficiently many edges st with length $\Theta(\|e\|)$. Roughly speaking, this greedy pruning procedure works as follows: for any index $j \geq 0$, let $L_j \subseteq E$ be the set of edges whose lengths are in the range $[1.01^j, 1.01^{j+1})$. We start with an upper bound on the sparsity $\alpha \geq |E|/|E_{\text{spa}}|$, and for each length scale $[1.01^j, 1.01^{j+1})$, we will greedily add *substitute* edges $e = zw \in L_j$ to the spanner, whose inclusion in E could help in removing at least $\alpha/100$ existing edges from L_j without increasing the stretch by too much. After such a pruning procedure, we are able to decrease our upper bound α on the ratio $|E|/|E_{\text{spa}}|$ by a constant factor, ignoring the new substitute edges, whose sparsity is constant. By working carefully, we demonstrate that this

pruning procedure can be iterated until the sparsity upper bound has reduces from α to $O(\log \alpha)$, while the stretch increases only slightly.

Thus far we have presented the informal descriptions of two different *pruning phases*. Our spanner construction employs these two pruning phases on top of the original $(1+\delta)$ -spanner (X, E) that we start from, one after another, to reduce the sparsity upper bound from α to $O(\log \alpha)$ while increasing the stretch from $1+\delta$ to $1+O(\delta)$. We remark that there is a delicate interplay between the two pruning phases, which requires our algorithm to pay special care to additional subtleties, and in particular to explicitly distinguish between two types of edges; aiming for brevity, we will not discuss such subtleties in this high-level overview, but the details appear in Section 4.1 (refer to Definition 4.1 in particular for the definition of the two types of edges). Finally, to achieve the stretch and sparsity bounds claimed in Corollary 1.1, we apply the two pruning phases iteratively until the sparsity upper bound reduces to a constant. The full details of the construction, the stretch analysis and the sparsity analysis appear in Sections 4.1, 4.2 and 4.3, respectively.

Our spanner algorithm and the underlying charging scheme heavily rely on Euclidean geometry. However, surprisingly perhaps, both of them are completely unaffected by the *dimension* of the input point set. (Our algorithm runs in low polynomial time in any dimension; however, the fast implementation of our algorithm described below does depend on the dimension.) The dependencies on the dimension in Corollary 1.1 (ignoring the fast implementation running time) stem only from the sparsity and lightness dependencies on the dimension in the original spanner (X, E) .

Lightness analysis. It turns out that the same spanner construction algorithm described above also approximates the instance-optimal lightness. More specifically, relying on the fact that the sparsity and lightness bounds of the initial (e.g., greedy) spanner (X, E) are both $\epsilon^{-\Theta(d)}$, we can prove that the instance-optimal lightness bound in the resulting spanner is (basically) the same as the sparsity bound. While our lightness analysis builds on the sparsity analysis, it has to drill quite a bit deeper in order to overcome another significant technical challenge, which we highlight next. The full details are in Section 4.4. When comparing our $(1+\delta)$ -spanner (X, E) against the optimal (now in terms of weight) spanner (X, E_{light}) , for any edge $st \in E$, let π denote a $(1+\epsilon)$ -spanner path in (X, E_{light}) between s and t . While in the sparsity analysis we charged the edge $st \in E$ to each edge e along π with a fractional cost of $\frac{\|e\|}{\|st\|}$, so that the total fractional costs of these edges is at least 1, such a charging is not suitable for the lightness analysis. Instead, we shall charge edge st to each edge e along π with a cost of $\|e\|$, so that the total costs of these edges is at least $\|st\|$, thus the total costs of the edges in E_{light} is at least $\|E\|$. Therefore, if we could argue that the total costs received by any edge $e \in E_{\text{light}}$ is at most λ , that would directly imply that $\|E\| \leq \lambda \cdot \|E_{\text{light}}\|$.

By adapting our techniques from the sparsity analysis, we can bound by $O(1)$ the number of times any edge $e \in E_{\text{light}}$ gets charged from the same edge set L_j (i.e., the edges whose lengths are in the range $[1.01^j, 1.01^{j+1})$). However, the total amount of charges over all length scales (or indices j) increases by a factor of $O(\log \Phi)$, where Φ is the spread of the point set. Using a standard trick, this blowup can be reduced to $O(\log n)$, but this is still insufficient for achieving instance-optimality. This is how lightness is different from sparsity, as charges of sparsity are decreasing geometrically as j increases, while charges of lightness remain the same for all scales j .

To bypass the logarithmic blowup, let us go over all edges in E in a non-decreasing order of lengths and allocate their charges to edges in E_{light} in an “adaptive” manner. Consider any edge $st \in E$ as well as a $(1+\epsilon)$ -spanner path π between s and t in (X, E_{light}) . If most edges of π have only been charged a small number of times by edges in E shorter than st , then we can safely charge st proportionally only to those edges (we charge any such edge e a cost of say $10\|e\|$). Otherwise, there exists a set of edges $z_1w_1, z_2w_2, \dots, z_lw_l$ on π such that:

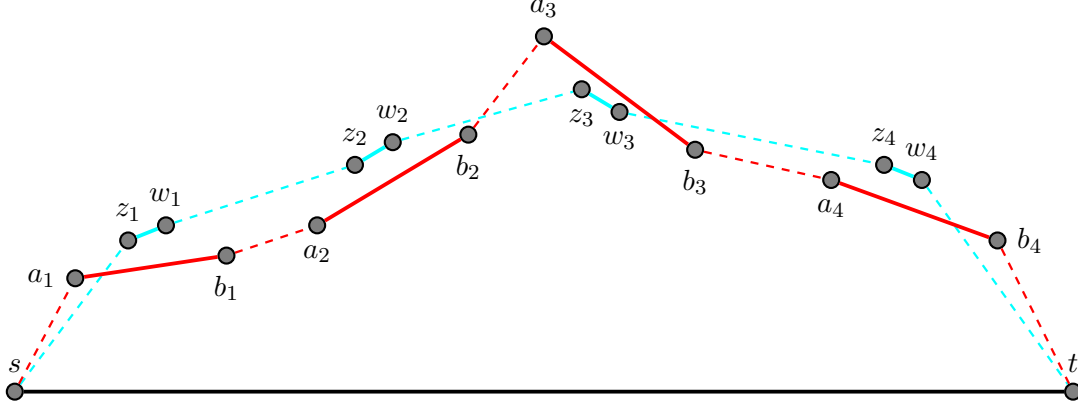


Figure 3: The blue path represents a $(1 + \epsilon)$ -spanner path π between s and t in (X, E_{light}) , and the blue solid edges are the ones already receiving heavy charges from shorter edges in E . Then, for each blue solid edge $z_i w_i$, we can find a longer red solid edge $a_i b_i \in E$ that charged to $z_i w_i$. Thus, we can stitch together all these red solid edges with existing edges in E to create a good spanner path between s and t , and so edge st does not need to stay in E anymore.

- $\sum_{i=1}^l \|z_i w_i\| \geq \Omega(\|st\|)$; and
- each edge $z_i w_i$ has been charged multiple times already.

In this case, we can prove that for each such edge $z_i w_i$ of π , there exists an edge $a_i b_i \in E$, which charges to $z_i w_i$ and is much longer than $\|z_i w_i\|$ yet much shorter than $\|st\|$. Using this key insight carefully, we prove that existing edges in E can be used to “stitch together” all the edges $a_1 b_1, a_2 b_2, \dots, a_l b_l$ into a good spanner path (of small stretch) between s and t in (X, E) , making the direct edge st unnecessary in E , and so we can show that st could not have been added to E . See Figure 3 for an illustration.

Fast implementation. Next, we highlight three main technical difficulties behind achieving a near-linear time implementation of our spanner construction (in low-dimensional spaces). Recall that our implementation takes $\epsilon^{-O(d)} n \log^2(n)$ time; see Section 5 for the full details.

The first difficulty is to efficiently locate all the substitute edges. (Recall that the substitute edges are needed if helper edges do not exist, and they are used for a more aggressive pruning of edges.) A straightforward implementation would enumerate all possible choices of a substitute edge xy , and count how many edges any such edge xy can prune from the current spanner if it is added to the spanner. Such an implementation would require quadratic time to find a substitute edge each time. Instead, we will build a *hierarchy of nets* (as in [CGMZ16]), and for each edge $st \in E$, its substitute edge will be restricted to $\epsilon\|st\|$ -net points nearby. Using the standard packing bound in $\mathbb{R}^d$, the total number of possible substitute edges for st is bounded by $\epsilon^{-O(d)}$. Then, we can initialize and maintain an efficient data structure in $\epsilon^{-O(d)} n \log(n)$ total time, which counts the number of edges assigned to each possible substitute edge; using this data structure, we can repeatedly select the best substitute edges in $\epsilon^{-O(d)} n \log(n)$ total time.

The second difficulty stems from the need to find a helper edge zw for any existing edge $st \in E$. A straightforward implementation of this task takes quadratic time by checking all possible helper edges zw . To narrow down our search space, we will again build an $\epsilon\|st\|$ -net and only look for net points around the segment st in search of a helper edge zw . Using the packing bound, we can show that the total number of candidate edges is at most $\epsilon^{-O(d)}$.

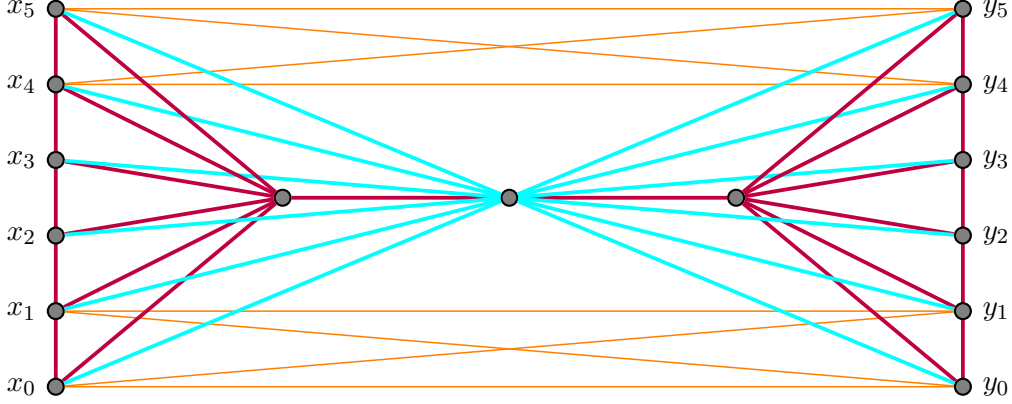


Figure 4: The red edges are in both the optimal and the greedy spanner; the blue edges are only in the optimal spanner, and the orange edges are only in the greedy spanner. This point set fools the greedy algorithm not to add the blue edges, so that it later has to add all orange edges, which form two bi-cliques, incurring quadratic sparsity.

As the third difficulty, we need to go over all edges in $st \in E$ in a non-decreasing order of lengths and, for each one, decide if we should keep it in the new spanner. Here, we need to quickly determine if the distance between the two endpoints s and t in the new spanner is already well-preserved, which entails an (approximate) shortest path computation. If we directly apply Dijkstra’s algorithm to compute shortest paths, then the total time for such computations over all edges $st \in E$ would be quadratic. To reduce the runtime, we will adopt the approach from [DN97] used for constructing an *approximate* version of the greedy spanner. Roughly speaking, to compute an (approximate) shortest path between s and t , we build a *cluster graph* that contracts clusters of radius $\epsilon \|st\|$, and then look at the cluster centers near s and t ; importantly, we only apply Dijkstra’s algorithm locally on these cluster centers. Relying again on the standard packing bound in $\mathbb{R}^d$, we can prove that the number of cluster centers around s and t is also at most $\epsilon^{-O(d)}$, which allows us to reduce the running time of an (approximate) shortest path computation to $\epsilon^{-O(d)}$.

Lower bounds for greedy spanners. Our hard instances for the poor performance of the greedy algorithm build on the dramatic difference between the sparsity of Euclidean spanners with or without Steiner points [BT22, LS22]: $O(\epsilon^{(-d+1)/2})$ versus $O(\epsilon^{-d+1})$. We modify the previously known worst-case instances for Euclidean Steiner spanners: Two large sets uniformly distributed along two parallel lines, and a small set of middle points between the two lines (a schematic example is in Figure 4; see Figures 9 and 10 for more accurate illustrations). In previous work [BT22, LS22], the middle points played the role of “Steiner points” (recall that a Steiner spanner does not have to maintain the stretch factor for paths to and from Steiner points). We include the middle points in the input. Importantly, the optimal (i.e., sparsest) spanner can take advantage of the middle points as convenient “via” points between the two large point sets (similarly to Steiner spanners), but the greedy algorithm misses their potential and is forced to add the complete bipartite graph between the two large point sets on the two lines; see Theorems 1.1 and 6.1 for details. The main challenge is the analysis of the greedy spanner. We heavily use geometry in the design and analysis of the lower bound instances to maintain a tight approximation of the stretch between point pairs during the greedy process.

Our lower bounds for *sparsity* use edges of comparable weight, and immediately give the same lower bounds for *lightness*. However, we can obtain stronger lower bounds for *lightness* with a

surprisingly simple point set: Uniformly distributed points along a circular arc (not the entire circle!). Due to the uniform distribution, we can easily analyze the greedy algorithm with stretch $(1 + x\epsilon)$ for any x , $1 \leq x \leq O(\epsilon^{1/2})$: The greedy algorithm includes a path along the circular arc; and then at a certain threshold, it adds a large number of heavy edges of equal weight. However, an optimum (i.e., lightest) spanner can use a much smaller number of “shortcut” edges instead (similarly to the *helper* edges in our upper bound construction). In our basic example (Theorem 6.2), a single shortcut edge of G_{light} trades off against $O(\epsilon^{-1})$ almost diametric edges of G_{gr} . For a lower bound for $(1 + x\epsilon)$ -spanners (Theorem 1.2), we use a hierarchy of shortcut edges.

3 Preliminaries

Let $\kappa = 10^4$ be a large but fixed constant independent of n , d , and ϵ ; and assume that $\epsilon > 0$ is relatively small compared to κ^{-1} and d^{-1} . More specifically, we assume the following relation:

$$\epsilon \cdot 2^{O(\log^*(d/\epsilon))} < \kappa^{-5}. \quad (1)$$

For any vector $e \in \mathbb{R}^d$, let $\|e\|$ denote the Euclidean length (i.e., ℓ_2 -norm) of e . For a graph H , let $V(H)$ and $E(H)$, resp., denote the vertex set and the edge set of H . For any set of edges $E \subseteq \binom{X}{2}$, let $\|E\|$ be the total length of the edges in E ; and the weight of a graph $H = (X, E)$ is defined as $\|H\| = \|E\|$.

For any pair of points $s, t \in \mathbb{R}^d$, let st denote the segment that connects s and t , and let $\vec{st}$ or $t - s$ be the vector directed from s to t . We will sometimes use the notation $s \rightsquigarrow t$ for the shortest (spanner) path between s and t in a graph H when H is clear from context.

For any pair of vectors $e_1, e_2 \in \mathbb{R}^d$, their angle $\angle(e_1, e_2)$ is defined as:

$$\angle(e_1, e_2) \stackrel{\text{def}}{=} \arccos \left(\frac{|e_1 \cdot e_2|}{\|e_1\| \cdot \|e_2\|} \right)$$

For a polygonal path π and two vertices p, q of π , let $\pi[p, q]$ be the sub-path of π between p and q . For an edge e and a line st , let $\text{proj}_{st}(e)$ be the orthogonal projection of e onto the line st . We generalize [BT22, Lemma 4], originally stated for a polygonal path π , to the setting where π is a sequence of edges; we include the proof for completeness.

Lemma 3.1 (Lemma 4 in [BT22]). *Consider a sequence of edges π (not necessarily a polygonal path) whose projection on a line is segment ab . Let $E(\pi, ab, \theta)$ be the set of edges $e \in \pi$ such that $\angle(e, ab) \leq \theta$. If $\|\pi\| \leq (1 + \epsilon)\|ab\|$, then $\|E(\pi, ab, 2\sqrt{\epsilon})\| \geq 0.5\|ab\|$.*

Proof. Assume otherwise that $\|E(\pi, ab, 2\sqrt{\epsilon})\| < 0.5\|ab\|$. Then, since the projection of π on line ab is equal to the segment ab , we have:

$$\sum_{e \in \pi} \|\text{proj}_{ab}(e)\| \geq \|ab\|$$

which implies

$$\begin{aligned} \sum_{e \in \pi \setminus E(\pi, ab, 2\sqrt{\epsilon})} \|\text{proj}_{ab}(e)\| &\geq \|ab\| - \sum_{e \in E(\pi, ab, 2\sqrt{\epsilon})} \|\text{proj}_{ab}(e)\| \\ &\geq \|ab\| - \sum_{e \in E(\pi, ab, 2\sqrt{\epsilon})} \|e\| \\ &= \|ab\| - \|E(\pi, ab, 2\sqrt{\epsilon})\|. \end{aligned}$$

Recall that for every edge $e \in \pi \setminus E(\pi, ab, 2\sqrt{\epsilon})$, we have $\angle(e, ab) \geq 2\sqrt{\epsilon}$. Using Taylor estimate $1/\cos(x) \geq 1 + x^2/2$, and thus for any edge $e \in \pi \setminus E(\pi, ab, 2\sqrt{\epsilon})$, we have

$$\|e\| \geq \frac{\|\text{proj}_{ab}(e)\|}{\cos(2\sqrt{\epsilon})} \geq \|\text{proj}_{ab}(e)\| \cdot (1 + 2\epsilon).$$

Combined with the previous inequality, we obtain

$$\begin{aligned} \|\pi\| &= \sum_{e \in E(\pi, ab, 2\sqrt{\epsilon})} \|e\| + \sum_{e \in \pi \setminus E(\pi, ab, 2\sqrt{\epsilon})} \|e\| \\ &\geq \|E(\pi, ab, 2\sqrt{\epsilon})\| + (1 + 2\epsilon) \cdot (\|ab\| - \|E(\pi, ab, 2\sqrt{\epsilon})\|) \\ &\geq (1 + 2\epsilon)\|ab\| - 2\epsilon\|E(\pi, ab, 2\sqrt{\epsilon})\| \\ &> (1 + \epsilon)\|ab\|, \end{aligned}$$

which is a contradiction. $\square$

Given a finite point set $X \subset \mathbb{R}^d$, let $G_{\text{spa}} = (X, E_{\text{spa}})$ and $G_{\text{light}} = (X, E_{\text{light}})$ be the $(1 + \epsilon)$ -stretch Euclidean spanners of X with the minimum number of edges and the minimum weight, respectively. We will use the following statement, which bounds the sparsity and lightness of the greedy spanner; although more precise bounds are known (which explicate the constants in the O -notation), as mentioned, the bounds in the following statement will suffice for our purposes.

Lemma 3.2 ([CDNS95, RS98, NS07]). *If $H = (X, E)$ is the greedy $(1 + \epsilon)$ -spanner for $X \subset \mathbb{R}^d$, then $|E| \leq n \cdot \epsilon^{-O(d)}$, and $\|E\| \leq \|\text{MST}(X)\| \cdot \epsilon^{-O(d)}$, where $\text{MST}(X)$ is a Euclidean minimum spanning tree of X .*

4 Instance-Optimal Euclidean Spanners

4.1 A Greedy-Pruning Algorithm

We are given a set $X \subset \mathbb{R}^d$ of n points and a sufficiently small $\epsilon > 0$ satisfying Equation (1). Initially, let $H = (X, E)$ be an arbitrary $(1 + \epsilon)$ -spanner on X (for example, the greedy spanner). According to Lemma 3.2, we may assume that H has $n \cdot \epsilon^{-O(d)}$ edges and weight at most $\|\text{MST}(X)\| \cdot \epsilon^{-O(d)}$. Next, we will successively modify H to improve its sparsity and lightness while keeping the stretch under control. In each iteration, we will prune some of the edges in H and add new edges. In each iteration, we maintain an upper bound on the current stretch of H and its approximation ratio compared to optimal $(1 + \epsilon)$ -spanners w.r.t. sparsity and lightness. Specifically, we assume that H is a $(1 + \delta)$ -spanner for some $\delta \geq \epsilon$; the assumption $\delta \geq \epsilon$ (or $\delta = \Omega(\epsilon)$) will be crucial in the analysis of stretch and lightness later on. Furthermore, we assume that $\alpha \geq \max\{|E|/|E_{\text{spa}}|, \|E\|/\|E_{\text{light}}\|\}$; initially when E is a greedy spanner, we have $\delta = \epsilon$ and $\alpha = \epsilon^{-O(d)}$. The total number of iterations will be $\log^*(d/\epsilon) + O(1)$.

One iteration consists of two phases which construct two spanners: First $H_1 = (X, E_1)$ and then $H_2 = (X, E_2)$. At the end of an iteration, we will reassign $E \leftarrow E_2$. During the process, edges in E_1 and E_2 that come from E will be called *old* edges, and all other edges in E_1 and E_2 will be called *new* edges.

Classification of edges in E . As a preliminary step, the algorithm will distinguish between two types of edges in E . For an edge $st \in E$ and a point $x \in X$, let $\text{proj}(x)$ denote the orthogonal

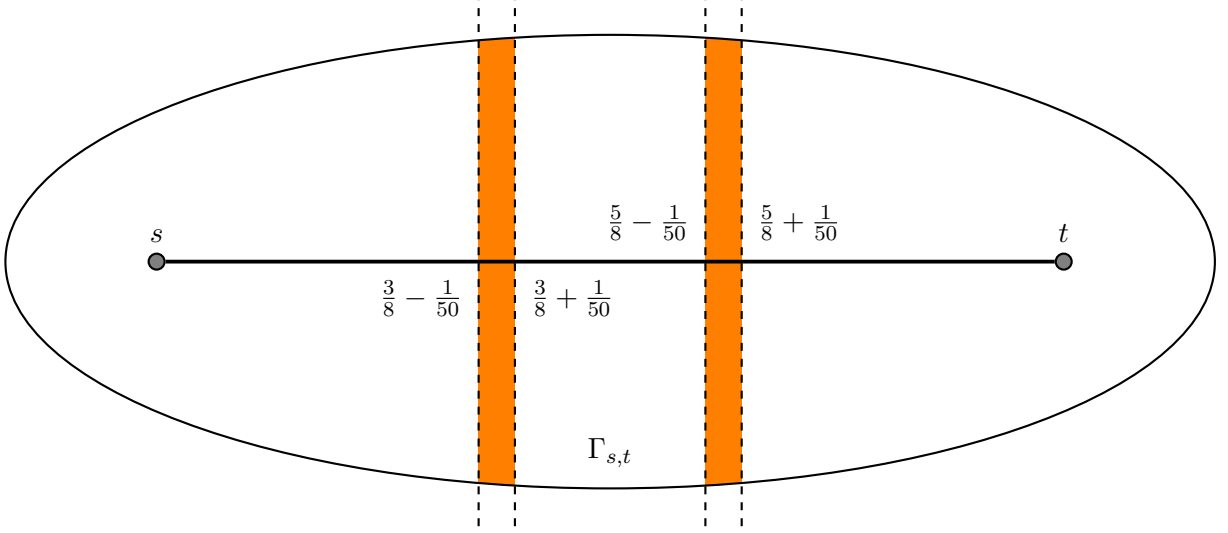


Figure 5: The two sets $A_{s,t}$ and $B_{s,t}$ are drawn as two orange regions in the ellipsoid around st . For simplicity, in this figure we assume that s and t lie on the x -axis with coordinate 0 and 1, respectively (and all the numbers shown designate x -coordinates).

projection of x on the straight line passing through s and t . Let $\Gamma_{s,t}$ be the ellipsoid defined by

$$\Gamma_{s,t} = \left\{ x \in \mathbb{R}^d : \|sx\| + \|xt\| \leq (1 + \epsilon)\|st\| \right\}$$

with foci s and t . We define the following two regions (see Figure 5 for an illustration):

$$A_{s,t} \stackrel{\text{def}}{=} \Gamma_{s,t} \cap \left\{ x \in \mathbb{R}^d : \frac{\|s - \text{proj}_{st}(x)\|}{\|st\|} \in \left[\frac{3}{8} - \frac{1}{50}, \frac{3}{8} + \frac{1}{50} \right], \frac{\|t - \text{proj}_{st}(x)\|}{\|st\|} \in \left[\frac{5}{8} - \frac{1}{50}, \frac{5}{8} + \frac{1}{50} \right] \right\},$$

$$B_{s,t} \stackrel{\text{def}}{=} \Gamma_{s,t} \cap \left\{ x \in \mathbb{R}^d : \frac{\|s - \text{proj}_{st}(x)\|}{\|st\|} \in \left[\frac{5}{8} - \frac{1}{50}, \frac{5}{8} + \frac{1}{50} \right], \frac{\|t - \text{proj}_{st}(x)\|}{\|st\|} \in \left[\frac{3}{8} - \frac{1}{50}, \frac{3}{8} + \frac{1}{50} \right] \right\}.$$

Definition 4.1. If $A_{s,t} \cap X$ or $B_{s,t} \cap X$ is empty for an edge $st \in E$, then st is called a **type-(i)** edge; otherwise, it is called a **type-(ii)** edge. Let $E^{(i)}$ and $E^{(ii)}$, resp., denote the set of type-(i) and type-(ii) edges in E .

First pruning phase. Initially, we set $E_1 \leftarrow E$ and $\beta \stackrel{\text{def}}{=} 1.01$. Without loss of generality, assume that all pairwise distances in X are at least 1. Next, we perform $O(\log \alpha)$ sub-iterations of pruning. In the i -th sub-iteration, go over all indices $j = 0, 1, 2, \dots$. For each index j , let $L_j \subseteq E$ be the set of old edges whose lengths are in the range $[\beta^j, \beta^{j+1})$. In the j -th iteration, enumerate all pairs of vertices $\{x, y\} \in \binom{X}{2}$ such that $\|xy\| \geq \beta^j/25$, and define a set of type-(i) edges

$$P_{x,y} = \left\{ st \in L_j \cap E_1 \cap E^{(i)} : \|sx\| + \|xy\| + \|yt\| \leq (1 + \epsilon) \cdot \|st\| \right\}.$$

If $|P_{x,y}| \geq \frac{\alpha}{2^{i_K}}$, then add xy to E_1 as a new edge, and remove all type-(i) edges in $P_{x,y}$ from E_1 .

Second pruning phase. To construct E_2 , initially set $E_2 \leftarrow E_1 \setminus E^{(ii)}$. Then, enumerate all type-(ii) edges of E_1 in an increasing order of edge weights. For each such edge $st \in E_1 \cap E^{(ii)}$, first check whether the stretch between s and t is approximately preserved in the graph (X, E_2) ; that is, whether

$$\text{dist}_{H_2}(s, t) \leq (1 + \kappa^2 \delta) \cdot \|st\|.$$

If so, move on to the next old edge in E_1 . Otherwise, since st is type-(ii), there exists a pair of vertices $a \in A_{s,t}, b \in B_{s,t}$. Then add one such edge ab to E_2 as a new edge, and st to E_2 as an old edge; we call ab the *helper* edge associated with st . After these procedures, move on to the next type-(ii) edge in E_1 .

Updating the parameters. After the two pruning phases, reassign $E \leftarrow E_2$. Before moving on to the next iteration, we need to update the upper bound on the stretch δ and approximation ratio α . Specifically, update $\delta \leftarrow \Delta(\kappa, \delta) \stackrel{\text{def}}{=} (1 + \delta) \cdot (1 + \kappa \delta) \cdot (1 + \kappa^2 \delta) - 1$, and $\alpha \leftarrow O(\log \alpha)$. The whole algorithm is summarized in Algorithm 1.

Algorithm 1: GreedyPrune(X, ϵ)

```

1  $\kappa \leftarrow 5000, \beta \leftarrow 1.01, \alpha \leftarrow (1/\epsilon)^{O(d)}, \delta \leftarrow \epsilon;$ 
2 let  $H = (X, E)$  be a greedy  $(1 + \epsilon)$ -spanner on the point set  $X \subseteq \mathbb{R}^d;$ 
3 for  $k = 1, 2, \dots, O(\log^*(d/\epsilon))$  do
    /* the first pruning phase */
4      $E_1 \leftarrow E;$ 
5     for  $i = 1, 2, \dots, O(\log \alpha)$  do
6         for  $j = 0, 1, 2, \dots$  do
7             define  $L_j = \{e \in E : \|e\| \in [\beta^j, \beta^{j+1})\};$ 
8             while there exists  $xy \in \binom{X}{2}$  such that  $\|xy\| \geq \beta^j/25$ , and  $|P_{x,y}| \geq \frac{\alpha}{2^i \kappa}$ , where
                 $P_{x,y} = \{st \in L_j \cap E_1 \cap E^{(i)} : \|sx\| + \|xy\| + \|yt\| \leq (1 + \epsilon) \cdot \|st\|\}$  (see
                Definition 4.1);
9             add  $xy$  to  $E_1$  as a new edge, and remove  $P_{x,y}$  from  $E_1;$ 
        /* the second pruning phase */
10     $E_2 \leftarrow E_1 \setminus E^{(ii)};$ 
11    for edge  $st \in E_1 \cap E^{(ii)}$  in non-decreasing order in terms of norm do
12        if  $\text{dist}_{H_2}(s, t) > (1 + \kappa^2 \delta) \|st\|$  then
13             $E_2 \leftarrow E_2 \cup \{st\};$ 
14            find  $a \in A_{s,t}, b \in B_{s,t}$  and add  $ab$  to  $E_2$  as a new helper edge;
15     $E \leftarrow E_2, \delta \leftarrow \Delta(\kappa, \delta), \alpha \leftarrow O(\log \alpha);$ 
16 return  $E;$ 
```

4.2 Stretch Analysis

Before analyzing the stretch of our spanner, we first need to bound the value of δ throughout all $O(\log^*(d/\epsilon))$ iterations of pruning.

Claim 4.1. *Throughout all $O(\log^*(d/\epsilon))$ iterations of pruning, we have $\delta < \kappa^{-5}$ and $\Delta(\kappa, \delta) < (\kappa + 1)^2 \delta$.*

Proof. We show by induction on i that right before the i -th iteration of pruning, we have $\delta \leq (\kappa + 1)^{2(i-1)}\epsilon$. For the basis step, recall that at the beginning when E was initialized as a greedy $(1 + \epsilon)$ -spanner, we have $\delta = \epsilon$. For the induction step, suppose that $\delta \leq (\kappa + 1)^{2(i-2)}\epsilon < \kappa^{-5}$ (by Equation (1)) right before the $(i - 1)$ -st iteration. Then, at the end of the $(i - 1)$ -st iteration, we have updated δ as:

$$\begin{aligned}
\Delta(\kappa, \delta) &= (1 + \delta) \cdot (1 + \kappa\delta) \cdot (1 + \kappa^2\delta) - 1 \\
&= (1 + (\kappa + 1)\delta + \kappa\delta^2) \cdot (1 + \kappa^2\delta) - 1 \\
&< (1 + (\kappa + 2)\delta) \cdot (1 + \kappa^2\delta) - 1 \\
&= (\kappa^2 + \kappa + 2)\delta + \kappa^2(\kappa + 2)\delta^2 \\
&< (\kappa + 1)^2\delta \\
&\leq (\kappa + 1)^2 \cdot (\kappa + 1)^{2(i-2)}\epsilon \\
&= (\kappa + 1)^{2(i-1)}\epsilon.
\end{aligned}$$

Therefore, at the end of the $(i - 1)$ -st iteration, and so also at the beginning of the i th iteration, we have $\delta < (\kappa + 1)^{2(i-1)}\epsilon$, which completes the induction step. It follows that throughout all $O(\log^*(d/\epsilon))$ iterations we have $\Delta(\kappa, \delta) \leq (\kappa + 1)^{O(\log^*(d/\epsilon))}\epsilon < \kappa^{-5}$, where the last inequality follows by employing Equation (1) again. $\square$

Next, let us show an upper bound on the stretch during the execution of the first pruning phase.

Claim 4.2. *At the end of the first pruning phase, for any edge $st \in E$, we are guaranteed that $\text{dist}_{H_1}(s, t) \leq (1 + \kappa\delta) \cdot \|st\|$. Also, at the end of the second phase, for any edge $st \in E_1 \cap E$, we have that $\text{dist}_{H_2}(s, t) \leq (1 + \kappa^2\delta) \cdot \|st\|$.*

Proof. The first assertion holds trivially for all edges that remain in E_1 until the end of the first phase; this includes all type-(ii) edges and possibly some type-(i) edges. We henceforth restrict the attention only to edges that get pruned from E_1 during the first phase, and for each such edge e we consider the index j such that $e \in L_j \cap E_1 \cup E^{(i)}$. We will prove that the stretch of each edge $e = st$ that is pruned from E_1 is in check (i.e., $\text{dist}_{H_1}(s, t) \leq (1 + \kappa\delta) \cdot \|st\|$) by induction on j , $j \geq 0$. We stress that the induction is applied on the final edge set E_1 at the end of the first pruning phase (when the last sub-iteration ends).

As the basis when $j = 0$, all edges in $E^{(i)} \cap L_0$ would be added to E_1 and never pruned (otherwise their stretch would be at least 2), so the stretch between the endpoints of any type-(i) edge in L_0 is 1. Now, for $j \geq 1$, let us assume $\text{dist}_{H_1}(s, t) \leq (1 + \kappa\delta) \cdot \|st\|$ for every edge $st \in L_k$ for every $k < j$. Consider any type-(i) edge $st \in L_j$ which was removed from E_1 during the first phase. Then, by the algorithm, there must exist an edge $xy \in E_1$ such that $\|xy\| \geq \beta^j/25$ and

$$\|sx\| + \|xy\| + \|yt\| \leq (1 + \epsilon)\|st\|.$$

Note that this edge will not be removed later on during the first phase since we only prune old edges. As $\|xy\| \geq \beta^j/25$ and $\|st\| < \beta^{j+1}$, we have

$$\max\{\|sx\|, \|yt\|\} \leq (1 + \epsilon)\|st\| - \|xt\| \leq \left(1 + \epsilon - \frac{1}{25\beta}\right)\|st\| < \frac{\|st\|}{\beta^2}.$$

Since we assumed (X, E) is a $(1 + \delta)$ -spanner for X , there exist two paths γ_1 and γ_2 in E connecting s, x and y, t such that

$$\|\gamma_1\| \leq (1 + \delta)\|sx\|,$$

$$\|\gamma_2\| \leq (1 + \delta)\|yt\|.$$

Therefore, every edge $e = s't'$ on the paths γ_1 or γ_2 has length at most

$$\|e\| \leq (1 + \delta) \max\{\|sx\|, \|yt\|\} \leq \frac{1 + \delta}{\beta^2} \|st\| < \frac{\|st\|}{\beta}.$$

In other words, e belongs to some set L_k for $k < j$. Using the inductive hypothesis, $\text{dist}_{H_1}(s', t') \leq (1 + \kappa\delta)\|s't'\|$, and consequently we have the following inequalities which conclude the induction

$$\begin{aligned} \text{dist}_{H_1}(s, t) &\leq \text{dist}_{H_1}(s, x) + \|xy\| + \text{dist}_{H_1}(y, t) \\ &\leq (1 + \kappa\delta)((1 + \epsilon)\|st\| - \|xy\|) + \|xy\| \\ &= (1 + \epsilon)(1 + \kappa\delta)\|st\| - \kappa\delta\|xy\| \\ &\leq \left((1 + \epsilon)(1 + \kappa\delta) - \frac{\kappa\delta}{25\beta} \right) \|st\| < (1 + \kappa\delta)\|st\|, \end{aligned}$$

where the penultimate inequality holds as $\|xy\| \geq \frac{1}{25\beta}\|st\|$ and the last inequality follows from Equation (1) and Claim 4.1, which yield $\epsilon(1 + \kappa\delta) < \frac{\kappa\delta}{25\beta}$.

During the second pruning phase, if an old edge $st \in E_1$ was not added to E_2 , there must exist a path π between s and t consisting of edges in E_2 such that $\|\pi\| \leq (1 + \kappa^2\delta) \cdot \|st\|$. Since the set E_2 grows monotonically, the path π is preserved for the remainder of the phase. $\square$

Corollary 4.1. *For every $st \in \binom{X}{2}$, we have:*

$$\text{dist}_{H_2}(s, t) \leq (1 + \Delta(\kappa, \delta)) \cdot \|st\| < (1 + (\kappa + 1)^2\delta) \cdot \|st\|.$$

Note that the second inequality in the corollary above holds as $\delta < \kappa^{-5}$ by Claim 4.1.

4.3 Sparsity Analysis

We first analyze how the number of edges in E changes in one iteration, and then consider k consecutive iterations at the end of this subsection. To analyze sparsity, we devise a charging scheme Ψ_0 that maps (fractionally) edges in E to edges in E_{spa} . In fact, the charging scheme charges (possibly fractionally) edges from E to edges in a subdivision of E_{spa} . We also stress that the total number of edges in the subdivision of E_{spa} to which we charge is at most $\kappa \cdot |E_{\text{spa}}| = O(|E_{\text{spa}}|)$.

Charging scheme from E to E_{spa} . For each edge $st \in E$, find a path $\pi_{s,t}$ in E_{spa} between s and t such that $\|\pi_{s,t}\| \leq (1 + \epsilon) \cdot \|st\|$. If $st \in E^{(i)}$, then by definition $A_{s,t}$ or $B_{s,t}$ is empty. Since $\pi_{s,t}$ is a path connecting s, t with total weight at most $(1 + \epsilon)\|st\|$, the entire path $\pi_{s,t}$ should lie within the ellipsoid $\Gamma_{s,t}$ and thus there must be a single edge e on $\pi_{s,t}$ that crosses the region $A_{s,t}$ or $B_{s,t}$. In this case, Ψ_0 charges the edge st to edge $e \in E_{\text{spa}}$.

Next, assume $st \in E^{(ii)}$. To charge type-(ii) edges to E_{spa} , we subdivide each edge $e \in E_{\text{spa}}$ evenly into κ sub-segments with at most $\kappa - 1$ Steiner points. Let $Y \supseteq X$ be the point set containing all original points and Steiner points, and let $E_{\text{spa}}^Y \subseteq \binom{Y}{2}$ denote the set of subdivided edges (clearly, $|E_{\text{spa}}^Y| = O(|E_{\text{spa}}^Y|)$). Our charging scheme will be from type-(ii) edges to edges in E_{spa}^Y . We distinguish between two cases, depending on the path $\pi_{s,t}$.

- (a) Suppose $\pi_{s,t} \cap A_{s,t}$ or $\pi_{s,t} \cap B_{s,t}$ is empty; that is, the polygonal path $\pi_{s,t}$ does not contain vertices in $A_{s,t}$ or $B_{s,t}$. Then, since $\pi_{s,t}$ lies in $\Gamma_{s,t}$ entirely, there must be an edge $e = s't'$ in $\pi_{s,t}$ that crosses $A_{s,t}$ or $B_{s,t}$. If e only crosses one of the two regions (say $A_{s,t}$), then we have

$$\|s - \text{proj}_{st}(s')\| < \left(\frac{3}{8} - \frac{1}{50}\right) \cdot \|st\|,$$

$$\left(\frac{3}{8} - \frac{1}{50}\right) \cdot \|st\| < \|\text{proj}_{st}(t') - t\| < \left(\frac{5}{8} - \frac{1}{50}\right) \cdot \|st\|.$$

Let $z \in Y \cap e$ be the Steiner point in $A_{s,t}$ on the segment e that is closest to s' ; such a point z must exist since each sub-segment of e has length at most $\frac{\|e\|}{\kappa} < \frac{\|st\|}{25}$. Then, Ψ_0 charges st to segment zt' which has length at least $\left(\frac{1}{25} - \frac{1}{\kappa}\right) \|st\| > \frac{\|st\|}{26}$.

If e crosses both regions $A_{s,t}$ and $B_{s,t}$, then let $z_1 \in Y \cap e$ be the Steiner point in $A_{s,t}$ that is closest to s' , and let $z_2 \in Y \cap e$ be the Steiner point in $B_{s,t}$ that is closest to t' . Then, Ψ_0 charges st to segment $z_1 z_2$ which has length at least $\left(\frac{1}{4} + \frac{1}{25} - \frac{2}{\kappa}\right) \|st\| > \frac{\|st\|}{4}$.

- (b) Otherwise, we will charge st *fractionally* to a set of edges in E_{spa}^Y . Move along $\pi_{s,t}$ from s to t and let p be the last vertex in $A_{s,t}$ and let q be the first vertex in $B_{s,t}$. As $\|\pi_{s,t}\| \leq (1 + \epsilon) \cdot \|st\|$ and $\|\text{proj}_{st}(p) - \text{proj}_{st}(q)\| \geq (1/4 - 1/25)\|st\|$, we know that:

$$\|\pi_{s,t}[p, q]\| \leq \|\text{proj}_{st}(p) - \text{proj}_{st}(q)\| + \epsilon \cdot \|st\| \leq (1 + 10\epsilon) \cdot \|\text{proj}_{st}(p) - \text{proj}_{st}(q)\|.$$

Therefore, applying Lemma 3.1, we know that

$$\|E(\pi_{s,t}[p, q], st, 2\sqrt{10\epsilon})\| > 0.5 \cdot \|\text{proj}_{st}(p) - \text{proj}_{st}(q)\|.$$

Then, for each edge $e \in E(\pi_{s,t}[p, q], st, 2\sqrt{10\epsilon})$, Ψ_0 charges a fraction of $\frac{2 \cdot \|e\|}{\|\text{proj}_{st}(p) - \text{proj}_{st}(q)\|}$ of edge st to edge e .

By design of our charging scheme Ψ_0 , we can upper bound the angle between any type-(ii) edge st and the edge e it charges to.

Claim 4.3. *If a type-(ii) edge st charges to an edge $xy \in E_{\text{spa}}^Y$, then angle $\angle(st, xy)$ is at most $15\sqrt{\epsilon}$. Furthermore, for all $z \in \{x, y\}$, the projection $\text{proj}_{st}(z)$ of z onto line st lies on the segment st and satisfies $\frac{\|s - \text{proj}_{st}(z)\|}{\|st\|} \in \left[\frac{3}{8} - \frac{1}{50}, \frac{5}{8} + \frac{1}{50}\right]$.*

Proof. The second assertion of the statement holds by the design of our charging scheme, so let us focus on the first assertion. If st charges to xy as in case-(b), then the assertion holds by design. Otherwise, assume st charges to xy as in case-(a). Define $\theta = \angle(st, xy)$. Since both x and y are on the path $\pi_{s,t}$, let us assume w.l.o.g. that x lies between s and y on $\pi_{s,t}$, and then we have

$$\begin{aligned} \epsilon \|st\| &\geq \|\pi_{s,t}\| - \|st\| \geq \|sx\| + \|xy\| + \|yt\| - \|st\| \\ &\geq \|xy\| - \|\text{proj}_{st}(xy)\| = (1 - \cos \theta) \|xy\| \\ &\geq \frac{\theta^2}{8} \cdot \|xy\| > \frac{\theta^2}{208} \cdot \|st\|. \end{aligned}$$

Here we have used the fact that $1 - \cos \theta = 2 \sin^2(\theta/2) > \theta^2/8$ and $\|xy\| > \frac{\|st\|}{26}$. Therefore, $\theta \leq 15\sqrt{\epsilon}$. $\square$

We need to argue that Ψ_0 is a valid charging scheme from E to $E_{\text{spa}} \cup E_{\text{spa}}^Y$.

Claim 4.4. *Every edge in E is fully charged to edges in a subdivision of E_{spa} ; that is, the total charges produced by any edge in E is at least 1.*

Proof. According to the charging scheme, every type-(i) edge is fully charged to an edge in E_{spa} . As for type-(ii) edges, in case (a), we are also using an integral charging (i.e., charging to a single edge) to edges in E_{spa}^Y . In case (b), the fractions of st charged to various edges in $E(\pi_{s,t}[p, q], 2\sqrt{10}\epsilon)$ sum to at least $2 \cdot \frac{1}{2} \frac{\|pq\|}{\|pq\|} = 1$ by Lemma 3.1, and so st is also fully charged fractionally to some Steiner edges in E_{spa} . $\square$

Next, let us analyze the sparsity of the first pruning phase.

Claim 4.5. *During the first pruning phase, the number of new edges added to E_1 is at most $O(|E_{\text{spa}}| \log \alpha)$. After the first pruning phase, the number of type-(i) edges in E_1 is at most $O(|E_{\text{spa}}|)$.*

Proof. During the first pruning phase, we show by induction on i that at the beginning of the i -th sub-iteration, $|E_1 \cap E^{(i)}|$ is at most $|E_{\text{spa}}| \alpha / 2^{i-1}$. For the basis when $i = 1$, this bound holds as $\alpha \geq |E|/|E_{\text{spa}}|$.

For the inductive step, to bound $|E_1 \cap E^{(i)}|$ after the i -th sub-iteration, we need to utilize our charging scheme for type-(i) edges. Suppose, for the sake of contradiction, that there are more than $|E_{\text{spa}}| \alpha / 2^{i-1}$ type-(i) edges remaining in $E_1 \cap E^{(i)}$ after the i -th sub-iteration. Then, by the pigeonhole principle, there exists a set F of more than $\alpha / 2^{i-1}$ type-(i) edges currently in E_1 charging to the same edge $xy \in E_{\text{spa}}$. By our charging scheme, for each such edge $e \in F$, we have

$$\|xy\| \leq (1 + \epsilon)\|e\| \leq (1 + \epsilon)25 \cdot \|xy\| \leq 25\beta \cdot \|xy\|.$$

Therefore, by the pigeonhole principle there exists an index $j \geq 0$ such that

$$|F \cap L_j| \geq \frac{\alpha}{2^{i-1} \cdot \log_\beta 25\beta} > \frac{\alpha}{2^i \kappa}.$$

The last inequality holds since $\kappa = 10^4 > \log_\beta(25\beta)$. Now, consider the sub-iteration during the first pruning phase when we were processing edges in L_j .

To reach a contradiction, it suffices to show that at that time, it must be that $P_{x,y} \supseteq F \cap L_j$. In fact, for any edge $st \in F \cap L_j$, by the charging scheme, we know that xy is on a $(1 + \epsilon)$ -spanning path of E_{spa} between s and t , hence the triangle inequality yields

$$\|sx\| + \|xy\| + \|yt\| \leq (1 + \epsilon)\|st\|.$$

Note that for each edge $e = st \in F \cap L_j$, we have $\|xy\| \geq \beta^j / 25$. Therefore, the algorithm could have added xy to E_1 as a new edge and remove the entire set $P_{x,y}$, leading to a contradiction. This completes the proof of the induction step.

It remains to bound the number of new edges added to E_1 . In the i -th sub-iteration, each time we add a new edge to E_1 , we decrease $|E_1 \cap E^{(i)}|$ by at least $\frac{\alpha}{2^i \kappa}$. Since $|E_1 \cap E^{(i)}|$ was at most $|E_{\text{spa}}| \alpha / 2^{i-1}$ at the beginning of the i -th sub-iteration, we could add at most $O(|E_{\text{spa}}|)$ edges to E_1 in this sub-iteration. It follows that at most $O(|E_{\text{spa}}| \log \alpha)$ new edges are added to E_1 in total. $\square$

Next, let us analyze the sparsity of the second pruning phase. Let us begin with a basic observation which follows directly from the algorithm description.

Observation 4.1. *Every type-(ii) edge $st \in E_2 \cap E^{(ii)}$ must have a helper edge ab when it was processed and added to E_2 during the second pruning phase.*

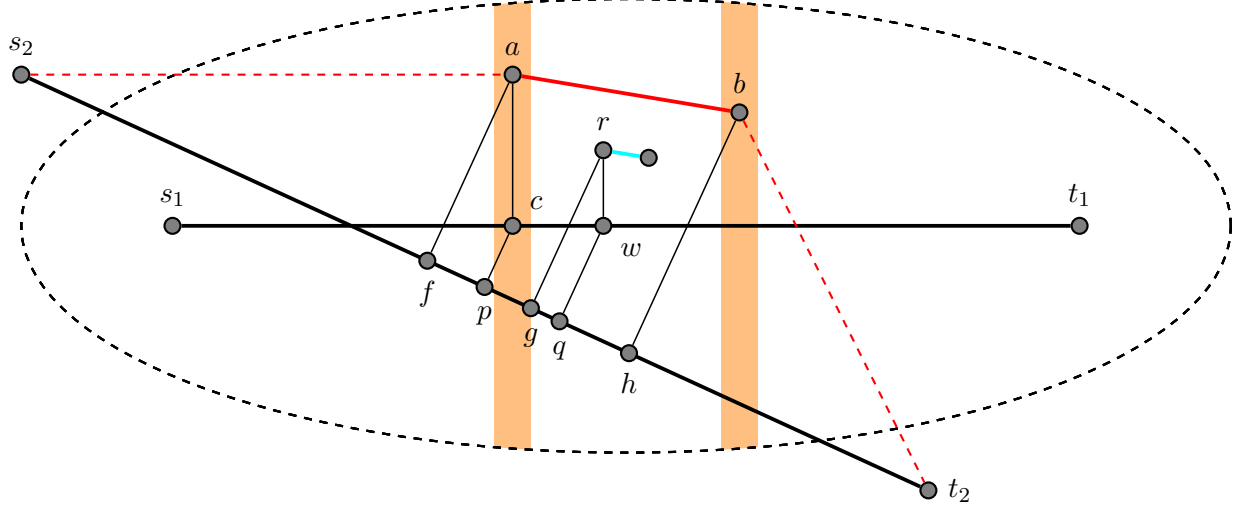


Figure 6: Both s_1t_1 and s_2t_2 are charging to e which is drawn as the cyan edge. If s_1t_1 was added to E_2 together with a helper edge ab , then s_2t_2 could not be added to E_2 later on since $(s_2 \rightsquigarrow a) \circ ab \circ (b \rightsquigarrow t_2)$ will be a good path in H_2 .

The following lemma is the key behind the charging argument.

Lemma 4.1. *Fix any edge $e \in E_{\text{spa}}^Y$ and level index $j \geq 0$. Then, after the second pruning phase, there is at most one type-(ii) edge in $E_2 \cap L_j$ that is charged to e .*

Proof. Assume for contradiction that there are two distinct type-(ii) edges $s_1t_1, s_2t_2 \in E_2 \cap L_j$ charging to the same edge $e \in E_{\text{spa}}^Y$. Let r be an arbitrary endpoint of e . Without loss of generality, assume $\|s_1t_1\| \leq \|s_2t_2\|$, and so edge s_1t_1 was processed before edge s_2t_2 . By Observation 4.1, a helper edge of s_1t_1 , say edge ab , was added to E_2 .

Write $D = \|s_1t_1\|$. Let c and w be the orthogonal projections of a and r on the line s_1t_1 ; let f, g , and h be the projections of a, r , and b on line s_2t_2 ; and let p and q be the projections of c and w on line s_2t_2 ; see Figure 6 for an illustration in the 2-dimensional case. By the design of our charging scheme and Claim 4.3, we know that g should land on segment s_2t_2 and $\|cw\| \leq 0.29 \cdot D$, and $\angle(s_1t_1, s_2t_2) \leq \angle(s_1t_1, e) + \angle(e, s_2t_2) \leq 30\sqrt{\epsilon}$. As s_2t_2 is also charged to e , we know that $\|s_2g\|/\|s_2t_2\| \in [\frac{3}{8} - \frac{1}{50}, \frac{5}{8} + \frac{1}{50}]$. Thus, by the triangle inequality, we obtain

$$\begin{aligned}
\|s_2p\| &\geq \|s_2g\| - \|gq\| - \|pq\| \\
&\geq 0.355D - \sin(\angle(s_1t_1, s_2t_2)) \cdot \|rw\| - \|cw\| \\
&\geq (0.355 - 2\sqrt{\epsilon} \cdot \sin(30\sqrt{\epsilon}) - 0.29) D \\
&> 0.06D.
\end{aligned}$$

Note that the last inequality holds for sufficiently small ϵ . Thus, $\|s_2f\| \geq \|s_2p\| - \|pf\| > 0.06D - \sin(\angle(s_1t_1, s_2t_2)) \cdot 2\sqrt{\epsilon}D > 0.06D - 60\epsilon D > 0.05D$. This also shows that both f and p should land on the segment s_2t_2 .

On the other hand, using the triangle inequality for the projections of segments cp, cw, wq on

the hyperplane orthogonal to line s_2t_2 , we have:

$$\begin{aligned}
\|cp\| &\leq \|wq\| + \|cw\| \cdot \sin(30\sqrt{\epsilon}) \\
&< \|wq\| + 30\sqrt{\epsilon}\|cw\| \\
&< \|wq\| + 8.7\sqrt{\epsilon}D \\
&\leq \|rg\| + \|rw\| + 8.7\sqrt{\epsilon}D \\
&\leq 12.7\sqrt{\epsilon}D.
\end{aligned}$$

Here, we have used the fact that $\|cw\| \leq 0.29D$ and $\|rg\|, \|rw\| \leq 2\sqrt{\epsilon}D$. Therefore, $\|af\| \leq \|ac\| + \|cp\| \leq 14.7\sqrt{\epsilon}D$. Symmetrically, we can show that $\|bh\| \leq 14.7\sqrt{\epsilon}D$ and $\|t_2h\| > 0.05D$.

Finally, let us show that $(s_2 \rightsquigarrow a) \circ ab \circ (b \rightsquigarrow t_2)$ can make a good path from s_2 to t_2 in H_2 . By the above calculations, as $\|s_2f\| > 0.05D$ and $\|af\| \leq 14.7\sqrt{\epsilon}D$, we have

$$\begin{aligned}
\|s_2a\| - \|s_2f\| &= \sqrt{\|s_2f\|^2 + \|af\|^2} - \|s_2f\| = \frac{\|af\|^2}{\sqrt{\|s_2f\|^2 + \|af\|^2} + \|s_2f\|} \\
&< 217\epsilon D^2 / 0.1D = 2170\epsilon D.
\end{aligned} \tag{2}$$

Symmetrically, we can show that h also lands on the segment s_2t_2 and

$$\|bt_2\| - \|ht_2\| \leq 2170\epsilon D. \tag{3}$$

Also, since $\angle(ab, fh) \leq 15\sqrt{\epsilon}$ by Claim 4.3, we have

$$\begin{aligned}
\|ab\| &\leq \frac{\|fh\|}{\cos(15\sqrt{\epsilon})} \leq (1 + 112.5\epsilon + O(\epsilon^2)) \cdot \|fh\| \\
&< (1 + 115\epsilon)\|fh\| < \|fh\| + 120\epsilon D.
\end{aligned}$$

The last inequality is because $\|fh\| \leq \|s_2t_2\| < \beta D$.

By Corollary 4.1, we have

$$\begin{aligned}
\text{dist}_{H_2}(s_2, a) &\leq (1 + (\kappa + 1)^2\delta) \cdot \|s_2a\|, \\
\text{dist}_{H_2}(b, t_2) &\leq (1 + (\kappa + 1)^2\delta) \cdot \|bt_2\|.
\end{aligned}$$

Therefore, using Equations (2) and (3) combined with Claim 4.1 (i.e., $\delta < k^{-5}$), at the time when edge s_2t_2 was being processed by the second pruning phase, we have

$$\begin{aligned}
\text{dist}_{H_2}(s_2, t_2) &\leq \text{dist}_{H_2}(s_2, a) + \|ab\| + \text{dist}_{H_2}(b, t_2) \\
&\leq (1 + (\kappa + 1)^2\delta) \cdot \|s_2a\| + (1 + (\kappa + 1)^2\delta) \cdot \|bt_2\| + \|ab\| \\
&\leq (1 + (\kappa + 1)^2\delta) \cdot (\|s_2t_2\| + 2860\epsilon D - \|ab\|) + \|ab\| \\
&< (1 + \kappa^2\delta)\|s_2t_2\| + ((2\kappa + 1)\delta\|s_2t_2\| + (1 + (\kappa + 1)^2\delta) \cdot 2860\epsilon D - (\kappa + 1)^2\delta\|ab\|) \\
&< (1 + \kappa^2\delta)\|s_2t_2\| + (2\kappa + 3000 - 0.2(\kappa + 1)^2) \delta\|s_2t_2\| \\
&< (1 + \kappa^2\delta)\|s_2t_2\|.
\end{aligned}$$

The penultimate inequality holds as $\|ab\| \geq 0.21\|s_1t_1\| > 0.21\beta^{-1}\|s_2t_2\| > 0.2\|s_2t_2\|$, $\delta \geq \epsilon$ and $\kappa = 10^4$. Therefore, s_2t_2 could not have been added to E_2 , leading to a contradiction. $\square$

Corollary 4.2. *The total number of type-(ii) edges added to E_2 is at most $O(|E_{\text{spa}}|)$.*

Proof. Consider any subdivided edge $e \in E_{\text{spa}}^Y$ within an edge in E_{spa} . By Lemma 4.1, for each $j \geq 0$, at most one type-(ii) edge in $L_j \cap E_2$ charged to e . By the fractional charging scheme, the total amount of charges that e receives across all $j \geq 0$ is a geometric sum which is bounded by a constant. Note that we analyze integral (case (a)) and fractional (case (b)) charges separately: Integral charges are incurred by edges on a constant number of levels, since by the charging scheme every edge charged to e must have weight $\Theta(\|e\|)$; the fractional charges to e could be incurred by edges on possibly many levels, but our charging scheme guarantees that the fractional charge to e decays geometrically with the level, hence the sum of fractional charges is bounded by a geometric sum. $\square$

Putting it all together. Corollary 4.2, together with Claim 4.5, yields $|E_2| = O(|E_{\text{spa}}| \log \alpha)$. Also, by Corollary 4.1, the stretch of H_2 is at most $1 + (\kappa + 1)^2 \delta$. Therefore, starting with (X, E) being a $(1 + \epsilon)$ -spanner with $\delta = \epsilon$ and $\alpha = \epsilon^{-O(d)}$, if we iterate these two pruning phases k times, we end up with a spanner with $(1 + \epsilon \cdot 2^{O(k)})$ -stretch and $O\left(\log^{(k)}(1/\epsilon) + \log^{(k-1)}(d)\right) \cdot |E_{\text{spa}}|$ edges, which concludes the sparsity bound of Theorem 1.3. Note that Algorithm 1 (Algorithm GreedyPrune) repeats the two pruning phases for $k = O(\log^*(d/\epsilon))$ iterations, the result of which provides a spanner with the stretch and sparsity bounds of Corollary 1.1.

4.4 Lightness Analysis

Similarly to the analysis of sparsity, we devise a charging scheme from edges in E to edges in E_{light} .

Preparation. Since we deal with edge lengths, we use more refined geometric properties. For each edge $st \in E$, let $\pi_{s,t}$ be a path in E_{light} between s and t such that $\|\pi_{s,t}\| \leq (1 + \epsilon) \cdot \|st\|$. For technical reasons, we wish that all points on the path $\pi_{s,t}$ are monotonically increasing in terms of their projections on the directed line $\overrightarrow{st}$, which is not necessarily the case in general. Therefore, as a preliminary step, we construct a sub-sequence of edges $\rho_{s,t} \subseteq \pi_{s,t}$ from s to t in $\mathbb{R}^d$ such that

- (1) the projection of $\rho_{s,t}$ on the line st is a segment that covers the entire segment st ;
- (2) every point on segment st is covered at most twice by the projection of $\rho_{s,t}$ on line st .

To define this sub-sequence $\rho_{s,t}$, consider the following iterative procedure which uses a cursor variable z starting at $z \leftarrow s$, as well as a partially constructed sub-sequence $\rho \subseteq \pi_{s,t}$ from s to z . In each iteration, let xy be the last edge on $\pi_{s,t}$ whose projection on st contains the projection of z on st ; namely $\text{proj}_{st}(z) \in \text{proj}_{st}(xy)$. Then, extend $\rho_{s,t}$ by $\rho_{s,t} \leftarrow \rho_{s,t} \cup \{xy\}$ and reassign $z \leftarrow y$. By the construction, it is easy to see that both requirements (1)–(2) on $\rho_{s,t}$ are met. Using $\rho_{s,t}$, we describe a charging scheme Ψ_1 that discharges the weight of the edges of E to edges in E_{light} .

Charging scheme from E to E_{light} . The charging scheme Ψ_1 is almost the same as Ψ_0 , except that we are using $\rho_{s,t}$ in lieu of $\pi_{s,t}$ for type-(ii) edges.

If $st \in E^{(i)}$, then by definition $A_{s,t}$ or $B_{s,t}$ is empty. Since $\pi_{s,t}$ is a path connecting s, t with total weight at most $(1 + \epsilon)\|st\|$, the entire path $\pi_{s,t}$ should lie within the ellipsoid and thus there must be a single edge e on $\pi_{s,t}$ that crosses the region $A_{s,t}$ or $B_{s,t}$. In this case, Ψ_1 charges weight $\|st\|$ to such an edge $e \in E_{\text{light}}$.

Next, assume that $st \in E^{(ii)}$. To charge type-(ii) edges to E_{light} , we subdivide each edge $e \in E_{\text{light}}$ evenly into κ sub-segments by adding at most $\kappa - 1$ Steiner points on e . Let $Y \supseteq X$ be the point set containing all original points and Steiner points, and let $E_{\text{light}}^Y \subseteq \binom{Y}{2}$ denote the set

of subdivided edges (clearly, $\|E_{\text{light}}\| = \|E_{\text{light}}^Y\|$). Our charging scheme will be from type-(ii) edges to edges in E_{light}^Y . We distinguish between two cases, depending on the path $\rho_{s,t}$.

- (a) Suppose that $\rho_{s,t} \cap A_{s,t}$ or $\rho_{s,t} \cap B_{s,t}$ is empty; that is, the edges in $\rho_{s,t}$ do not have any endpoints in $A_{s,t}$ or $B_{s,t}$. Then, since $\rho_{s,t}$ lies in $\Gamma_{s,t}$ entirely and $st \subseteq \text{proj}_{st}(\rho_{s,t})$, there must be an edge $e = s't'$ in $\rho_{s,t} \subseteq \pi_{s,t}$ that crosses $A_{s,t}$ or $B_{s,t}$. If e only crosses one of the two regions (say $A_{s,t}$), then we have:

$$\|s - \text{proj}_{st}(s')\| < \left(\frac{3}{8} - \frac{1}{50}\right) \cdot \|st\|,$$

$$\left(\frac{3}{8} - \frac{1}{50}\right) \cdot \|st\| < \|\text{proj}_{st}(t') - t\| < \left(\frac{5}{8} - \frac{1}{50}\right) \cdot \|st\|.$$

Let $z \in Y \cap e$ be the Steiner points on segment e which is in $A_{s,t}$ but the closest one from s' ; such a point z must exist since each sub-segment of e has length at most $\frac{\|e\|}{\kappa} < \frac{\|st\|}{25}$. Then, Ψ_1 charges the weight $\|st\|$ to segment zt' which has length at least $(\frac{1}{25} - \frac{1}{\kappa}) \|st\| > \frac{\|st\|}{26}$.

If e crosses both regions $A_{s,t}$ and $B_{s,t}$, then let $z_1 \in Y \cap e$ be the Steiner point in $A_{s,t}$ which is the closest one from s' , and let $z_2 \in Y \cap e$ be the Steiner point in $B_{s,t}$ which is the closest one from t' . Then, Ψ_1 charges weight $\|st\|$ of st to segment $z_1 z_2$ which has length at least $(\frac{1}{4} + \frac{1}{25} - \frac{2}{\kappa}) \|st\| > \frac{\|st\|}{4}$.

- (b) Otherwise, we will distribute the weight $\|st\|$ among multiple edges in E_{light} . Move along $\rho_{s,t}$ from s to t and let p be the last vertex in $A_{s,t}$ and let q be the first vertex in $B_{s,t}$. As $\|\pi_{s,t}\| \leq (1 + \epsilon) \cdot \|st\|$, we know that

$$\|\rho_{s,t}[p, q]\| \leq \|\text{proj}_{st}(p) - \text{proj}_{st}(q)\| + \epsilon \cdot \|st\| \leq (1 + 10\epsilon) \cdot \|\text{proj}_{st}(p) - \text{proj}_{st}(q)\|,$$

where $\rho_{s,t}[p, q] \stackrel{\text{def}}{=} \rho_{s,t} \cap \pi_{s,t}[p, q]$ refers to the sub-sequence of $\rho_{s,t}$ between p and q . Therefore, Lemma 3.1 yields

$$\|E(\rho_{s,t}[p, q], st, 2\sqrt{10\epsilon})\| > 0.5 \cdot \|\text{proj}_{st}(p) - \text{proj}_{st}(q)\|.$$

Then, for each edge $e \in E(\rho_{s,t}[p, q], st, 2\sqrt{10\epsilon})$, let Ψ_1 charge an amount of $\frac{\|e\|}{\|\rho_{s,t}[p, q]\|} \cdot \|st\|$ from st to edge e .

For some technical reason, we need to be more formal about the properties of charging schemes.

Definition 4.2 (weight charging). A **replete** weight charging scheme Ψ from an edge set A to edge set B is a mapping $\Psi : A \times B \rightarrow \mathbb{R}^+ \cup \{0\}$ such that:

- for any $f \in B$, if $\Psi(e, f) > 0$, then $\Psi(e, f) \geq \min\{\|f\|, \|e\|\}$;
- $\sum_{f \in B} \Psi(e, f) = \|e\|$ for any $e \in A$.

We say that e charges to f under Ψ if $\Psi(e, f) > 0$.

We can show that $\Psi_1 : E \times (E_{\text{light}} \cup E_{\text{light}}^Y) \rightarrow \mathbb{R}^+ \cup \{0\}$ is replete under Definition 4.2 for $A = E$ and $B = E_{\text{light}} \cup E_{\text{light}}^Y$.

Claim 4.6. *The charging scheme Ψ_1 is replete, and every edge e receives $O(\|e\|)$ amount of charges from any edge $st \in E$; furthermore, both endpoints of e are in X if st is type-(ii) and case-(b).*

Proof. If st is a type-(i) edge, then by the design of Ψ_1 we know that $\|st\| > \|e\| \geq \frac{\|st\|}{25\beta}$. So the amount of charges e receives from st is between $\|e\|$ and $25\beta \cdot \|e\|$. If st is a type-(ii) edge in case-(a), then we have $\|st\| > \|e\| \geq \frac{\|st\|}{30}$. Finally, if st is a type-(ii) edge in case-(b), then on one hand notice that $\|\rho_{s,t}[p, q]\| \leq \|\pi_{s,t}\| - \|sp\| \leq (1 + \epsilon - (\frac{3}{8} - \frac{1}{50}))\|st\| < \|st\|$. Consequently, e receives at least $\|e\|$ charges from st under Ψ_1 . On the other hand, by the design of Ψ_1 , the amount of charges e has received from st is at most

$$\frac{\|e\|}{\|\rho_{s,t}[p, q]\|} \cdot \|st\| \leq \frac{2\|e\|}{\frac{1}{4} - \frac{1}{25}} < 10\|e\|.$$

The second half of the statement holds since the charging scheme in case-(b) does not involve any Steiner points in the super-set Y . $\square$

Similar to the sparsity analysis, we show that each edge e in E_{light} receives $O(\|e\|)$ amount of charges from type-(i) edges after the first pruning phase, and so this phase increases the weight by $O(\|E_{\text{light}}\| \log \alpha)$.

Claim 4.7. *After the first pruning phase, the total weight of new edges added to E_1 is $O(\|E_{\text{light}}\| \log \alpha)$. Furthermore, the total weight of type-(i) edges in E_1 is $O(\|E_{\text{light}}\| \log \alpha)$.*

Proof. During the first pruning phase, we show by induction on i that at the beginning of the i -th sub-iteration, $\|E_1 \cap E^{(i)}\|$ is at most $\|E_{\text{light}}\| \alpha / 2^{i-1}$. For the basis when $i = 1$, this bound holds as $\alpha \geq \|E\| / \|E_{\text{light}}\|$.

For the inductive step, to bound $\|E_1 \cap E^{(i)}\|$ after the i -th sub-iteration, we need to utilize our charging scheme Ψ_1 of type-(i) edges. Suppose otherwise that the total weight of all type-(i) edges remaining in $E_1 \cap E^{(i)}$ is at least $\|E_{\text{light}}\| \alpha / 2^{i-1}$ after the i -th sub-iteration. Then, by the pigeon-hole principle, there exists a set F of more than $25\beta \cdot \alpha / 2^{i-1}$ type-(i) edges currently in E_1 charging to the same edge $xy \in E_{\text{light}}$. By our charging scheme, for each such edge $e \in F$, we have

$$\|xy\| \leq \|e\| \leq 25\beta \cdot \|xy\|.$$

Therefore, there exists an index $j \geq 0$ such that

$$|F \cap L_j| \geq \frac{\alpha}{2^{i-1} \cdot \log_{\beta} 25\beta} > \frac{\alpha}{2^i \kappa}.$$

Now, consider the time during the first pruning phase when we were processing all edges in L_j .

To reach a contradiction, it suffices to claim that at the moment, it must be that $P_{x,y} \supseteq F \cap L_j$. In fact, for any edge $st \in F \cap L_j$, by the charging scheme, we know that

$$\|sx\| + \|xy\| + \|yt\| \leq (1 + \epsilon)\|st\|.$$

Therefore, the algorithm could have added xy to E_1 as a new edge and remove the entire set $P_{x,y}$ leading to a contradiction. This completes the proof of the induction step.

It remains to bound the total weight of new edges added to E_1 . In the i -th sub-iteration, each time we add a new edge e to E_1 , we have decreased the weight $\|E_1 \cap E^{(i)}\|$ by at least $\frac{\alpha}{2^i \kappa} \cdot \|e\|$. Since $\|E_1 \cap E^{(i)}\|$ was at most $\|E_{\text{light}}\| \alpha / 2^{i-1}$ at the beginning, we could increase the weight of E by at most $O(\|E_{\text{light}}\|)$ in this sub-iteration, and thus by at most $O(\|E_{\text{light}}\| \log \alpha)$ overall. $\square$

Now, let us analyze the second pruning phase. Using the same calculation as in Lemma 4.1, we can prove the following claim for type-(b) edges.

Claim 4.8. *Fix an edge $e \in E_{\text{light}}^Y$ and a level $j \geq 0$. After the second pruning phase, Ψ_1 charged the weight of at most one type-(ii) edge in $E_2 \cap L_j$ to e .*

Charging scheme from $E_2 \cap E$ to E_{light}^Y . However, Claim 4.8 is not enough to establish a lightness bound because an edge in E_{light}^Y could be charged multiple times across all different levels $j \geq 0$, and the charges need not form a geometric sum as in the case of sparsity. To bound the lightness of E_2 , we describe another charging scheme Ψ_2 , from $E_2 \cap E^{(ii)}$ to E_{light}^Y , that works simultaneously with the second pruning phase. During the construction of the charging scheme Ψ_2 , we will maintain the following invariant.

Invariant 4.1. *The charging scheme Ψ_2 is replete, and it charges every edge $e \in E_{\text{light}}^Y$ at most κ times, and each time e receives at most $20 \|e\|$ amount of charges.*

At the beginning when $E_2 \cap E^{(ii)} = \emptyset$, no edges have been charged to E_{light} under the new charging scheme. The algorithm goes over all edges $st \in E_1 \cap E$ and decides whether st should stay in E_2 or not. For an edge st of type-(b), let $F \subseteq E_{\text{light}}^Y$ be the set of edges to which the charging scheme Ψ_1 distributed a positive portion of the weight $\|st\|$, that is, $F = \{e \in E_{\text{light}}^Y : \Psi_1(st, e) > 0\}$.

Claim 4.9. $\|F\| > \frac{1}{10} \|st\|$.

Proof. Since st is type-(b), by definition, F contains all edges in the sub-sequence $\rho_{s,t}$ from the last vertex $p \in A_{s,t}$ to the first vertex $q \in B_{s,t}$. Therefore, the orthogonal projection of F on st has length

$$\|\text{proj}_{st}(F)\| = \frac{1}{2} \|\text{proj}_{st}(p) - \text{proj}_{st}(q)\| \geq \frac{1}{2} \cdot \left(\frac{1}{4} - \frac{1}{25} \right) \cdot \|st\| > \frac{1}{10} \cdot \|st\|.$$

Therefore, $\|F\| \geq \|\text{proj}_{st}(F)\| > \frac{1}{10} \|st\|$. □

Let $S \subseteq F$ be the subset of edges that Ψ_2 has already charged κ times by edges prior to processing edge st . If $\|S\| \leq \frac{1}{2} \|F\|$, then the charging scheme Ψ_2 distributes the weight $\|st\|$ among $F \setminus S$ proportionally to their weight. Formally, for each edge $e \in F \setminus S$, if Ψ_1 charged λ of the weight $\|st\|$ to e , then let Ψ_2 charge an amount of $\|F\|/\|F \setminus S\| \cdot \lambda \leq 2\lambda$ to e ; that is,

$$\Psi_2(st, e) \leftarrow \begin{cases} \frac{\|F\|}{\|F \setminus S\|} \cdot \Psi_1(st, e) & e \in F \setminus S \\ 0 & e \notin F \setminus S. \end{cases}$$

Note that $\frac{\|F\|}{\|F \setminus S\|} \cdot \Psi_1(st, e) \leq 2\Psi_1(st, e)$. By Claim 4.6, this maintains Invariant 4.1.

Next, let us consider the harder case where $\|S\| > \frac{1}{2} \|F\|$. In fact, we will show that st could not have been added to E_1 ; that is, this case never occurs.

Lemma 4.2. *If $\|S\| > \frac{1}{2} \|F\|$, then edge st would not be added to E_2 ; that is, we already have $\text{dist}_{H_2}(s, t) \leq (1 + \kappa^2 \delta) \cdot \|st\|$.*

Putting it all together. Assuming Lemma 4.2, we can show that every edge $e \in E_{\text{light}}^Y$ receives at most $20\kappa\|e\|$ charges under Ψ_2 (cf. Invariant 4.1). Together with Claim 4.7, we can show that $\|E_2\| = O(\|E_{\text{light}}\| \log \alpha)$. Also, by Corollary 4.1, the stretch of H_2 is at most $1 + (\kappa + 1)^2\delta$. Therefore, starting with (X, E) being a $(1 + \epsilon)$ -spanner with $\delta = \epsilon$ and $\alpha = \epsilon^{-O(d)}$, if we iterate these two pruning phases k times, we end up with a spanner with $(1 + \epsilon \cdot 2^{O(k)})$ -stretch and $O\left(\log^{(k)}(1/\epsilon) + \log^{(k-1)}(d)\right) \cdot \|E_{\text{light}}\|$ total weight, which concludes the lightness bound of Theorem 1.3. Note that Algorithm 1 (Algorithm GreedyPrune) repeats the two pruning phases for $k = O(\log^*(d/\epsilon))$ iterations, the result of which provides a spanner with the stretch and sparsity bounds of Corollary 1.1.

Proof of Lemma 4.2. We break down the proof of Lemma 4.2 into a sequence of claims.

Claim 4.10. *For each $e \in E_{\text{light}}^Y$ that Ψ_2 has charged exactly κ times, there exists an edge $\chi(e) \in E_2$ that Ψ_1 has already charged to e such that $\kappa\|e\| \leq \|\chi(e)\| \leq \frac{1}{\kappa}\|st\|$; furthermore, the angle between $\chi(e)$ and e is bounded by $\angle(\chi(e), e) \leq 2\sqrt{10}\epsilon$.*

Proof. According to Claim 4.8, all edges in E_2 that charge to e under Ψ_1 are on different levels L_i . Therefore, by our construction of Ψ_2 , the total number of levels that cover the range $[\frac{1}{\kappa}\|st\|, \kappa\|e\|] \subseteq [\frac{1}{\kappa}\|st\|, \|st\|] \cup [\|e\|, \kappa\|e\|]$ is at most $2\lceil \log_\beta \kappa \rceil < \kappa$. By the pigeonhole principle, there exists an edge $\chi(e)$, as claimed.

For the second half of the statement, since $\chi(e)$ charges to e under Ψ_1 and $\|\chi(e)\| \geq \kappa\|e\|$, it must be of type-(ii) in case-(b). Hence, we have $\angle(\chi(e), e) \leq 2\sqrt{10}\epsilon$ by design of Ψ_1 . $\square$

Let $P = \{\chi(e) : e \in S\}$. By Definition 4.2 and Invariant 4.1, we know that $\|P\| \geq \|S\|$. Next, we will prove that st is not added to E_2 in the second pruning phase. The proof consists of two steps. At a high level, in the first step, we will select a subset of edges $Q \subseteq P$ such that $\|Q\| \geq \Omega(\|st\|)$; in the second step, we will stitch these edges into an st -path in E_2 of total length at most $(1 + \kappa^2\delta) \cdot \|st\|$, thus implying that st would be excluded from E_2 .

To construct Q starting with $Q = \emptyset$, iterate over all indices $i = O(\log \alpha), \dots, 0$ decrementally while adding edges to Q . In the i -th iteration, scan the segment st from s to t . Whenever we hit the projection $\text{proj}_{st}(\chi(e))$ of some edge $\chi(e) \in P \cap L_i$, add $\chi(e)$ to Q , and remove all edges from P that lie in the *buffer region* of $\chi(e)$; that is, all edges $\chi(f) \in P$ whose projections $\text{proj}_{st}(\chi(f))$ on st are at distance at most $2\beta^{i+1}$ from $\text{proj}_{st}(\chi(e))$.

From the construction process of Q , we see that the edges in Q have pairwise disjoint projections on the line st . After we have constructed the edge set Q , we can order the edges in Q according to their projections from s to t as $Q = \{a_1b_1, a_2b_2, \dots, a_\ell b_\ell\}$. Next, let us define a candidate path γ between s and t in the current H_2 (that does not yet include st).

Definition 4.3 (path stitching). *We define the st -path*

$$\gamma = (s \rightsquigarrow a_1) \circ a_1b_1 \circ (b_1 \rightsquigarrow a_2) \circ a_2b_2 \circ \dots \circ (b_{\ell-1} \rightsquigarrow a_\ell) \circ a_\ell b_\ell \circ (b_\ell \rightsquigarrow t),$$

which contains the edges $a_i b_i$ for all $1 \leq i \leq \ell - 1$, and where $(x \rightsquigarrow y)$ denotes the shortest path from x to y in the current graph H_2 .

To reach a contradiction, our goal is to show that $\|\gamma\| \leq (1 + \kappa^2\delta) \cdot \|st\|$. First, we show that the total length of the edges in Q is $\Omega(\|st\|)$.

Claim 4.11. $\|Q\| \geq \frac{1}{320}\|st\|$.

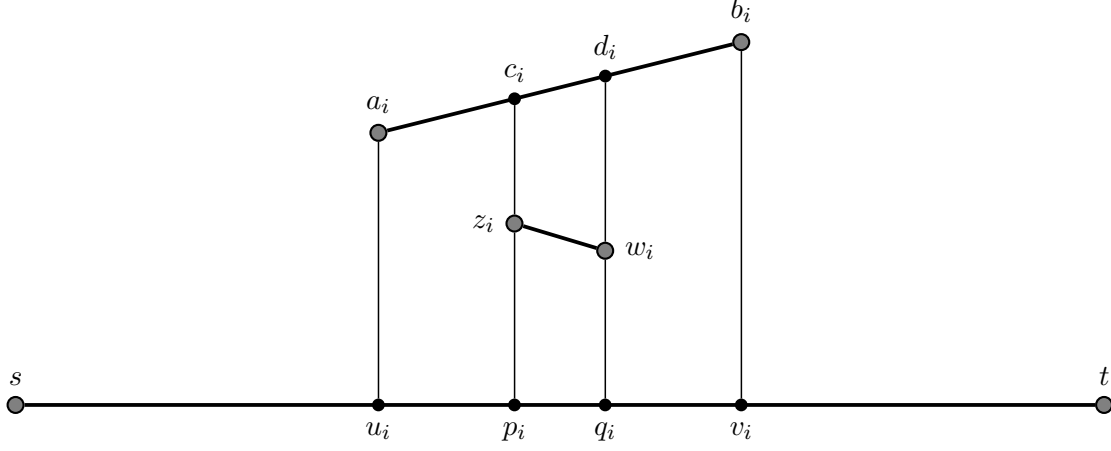


Figure 7: Auxiliary lines and points c_i, d_i, p_i, q_i, u_i , and v_i to assist our analysis.

Proof. By Claim 4.9 and our assumption in Lemma 4.2, we have $\|S\| \geq \frac{1}{2} \|F\| \geq \frac{1}{20} \|st\|$. So it suffices to show that $\|Q\| \geq \frac{1}{16} \|S\|$. According to the construction of Q , for every edge $e \in S$, either $\chi(e) \in P$ was added to Q , or $\chi(e)$ was removed from P because its projection $\text{proj}_{st}(\chi(e))$ on line st is at most $2\beta^{i+1}$ far away from the projection $\text{proj}_{st}(\chi(f))$ of another edge $\chi(f) \in Q \cap (L_i \cup L_{i+1} \cup \dots)$ for some level $i \in \{0, \dots, O(\log \alpha)\}$. In the latter case, let us associate this edge $\chi(e)$ with $\chi(f)$, and let P_f be the set of all edges associated with $\chi(f)$ under this definition. Since the construction procedure of Q enumerates all indices i from large to small, then $\|\chi(e)\| < \beta \cdot \|\chi(f)\|$ for all $\chi(e) \in P_f$. Since each point on segment st is covered by projections of at most two edges in F , the total length $\sum_{\chi(e) \in P_f} \|\text{proj}_{st}(\chi(e))\|$ is bounded by $14\beta^{i+1}$.

Noticing that $\angle(e, \chi(e)), \angle(e, st) \leq 2\sqrt{10}\epsilon$, we have $\angle(\chi(e), st) \leq 4\sqrt{10}\epsilon$. Therefore, we have

$$\|\text{proj}_{st}(\chi(e))\| \geq \cos(4\sqrt{10}\epsilon) \cdot \|\chi(e)\| = (1 - 80\epsilon + O(\epsilon^2)) \cdot \|\chi(e)\| \geq \beta^{-1} \|\chi(e)\|.$$

The last inequality holds when ϵ is below a certain constant threshold. Therefore, we have $14\beta^{i+1} \geq \sum_{\chi(e) \in P_f} \|\text{proj}_{st}(\chi(e))\| \geq \beta^{-1} \sum_{\chi(e) \in P_f} \|\chi(e)\|$, or $\sum_{\chi(e) \in P_f} \|\chi(e)\| \leq 14\beta^{i+2} < 15\|\chi(f)\|$. In other words, each edge $\chi(f)$ added to Q precludes a set of edges in P from joining Q which have total weight at most $15\|\chi(f)\|$, that is,

$$\begin{aligned} P &= Q \cup \bigcup_{\chi(e) \in Q} P_{\chi(e)} = \bigcup_{\chi(e) \in Q} (\{\chi(e)\} \cup P_{\chi(e)}), \\ \|P\| &= \sum_{\chi(e) \in Q} (\|\chi(e)\| + \|P_{\chi(e)}\|) \leq \sum_{\chi(e) \in Q} 16\|\chi(e)\| = 16\|Q\|. \end{aligned}$$

We can conclude that

$$\|Q\| \geq \frac{1}{16} \|P\| \geq \frac{1}{16} \|S\| \geq \frac{1}{320} \|st\|. \quad \square$$

To analyze the total length of γ , let us subdivide the path into multiple parts and analyze each part separately. For each edge $a_i b_i \in Q$, assume that $e_i = z_i w_i$ is the edge in S such that $\chi(e_i) = a_i b_i$. Draw two $(d-1)$ -dimensional hyper-planes perpendicular to line st through z_i and w_i which intersects lines st and $a_i b_i$ at c_i, d_i and p_i, q_i , respectively. See Figure 7 for an illustration.

Claim 4.12. *Both c_i and d_i lie on the segment $a_i b_i$; furthermore, $\min\{\|a_i c_i\|, \|d_i b_i\|\} \geq 0.25 \|a_i b_i\|$.*

Proof. It suffices to focus only on the inequality $\|a_i c_i\| \geq 0.25 \|a_i b_i\|$. Let $c'_i = \text{proj}_{a_i b_i}(z_i)$ be the projection of z_i on line $a_i b_i$. As $a_i b_i$ is charging to $z_i w_i$, point c'_i should land on the segment $a_i b_i$.

Since $\angle(a_i b_i, st) \leq 4\sqrt{10}\epsilon$ and $z_i c_i$ lies in the hyperplane perpendicular to st , we have

$$\|c_i c'_i\| = \left| \overrightarrow{c_i z_i} \cdot \frac{\overrightarrow{a_i b_i}}{\|a_i b_i\|} \right| \leq \sin(4\sqrt{10}\epsilon) \cdot \|c_i z_i\| < 4\sqrt{10}\epsilon \cdot \|c_i z_i\|,$$

which yields

$$\|c_i c'_i\| \leq \frac{4\sqrt{10}\epsilon}{\sqrt{1-160\epsilon}} \cdot \|a_i b_i\|.$$

As z_i lies in the ellipsoid $\{z \in \mathbb{R}^d : \|a_i z\| + \|b_i z\| \leq (1+\epsilon)\|a_i b_i\|\}$, we have $\|z_i c'_i\| < 2\sqrt{\epsilon} \cdot \|a_i b_i\|$. Therefore, $\|c_i c'_i\| < \frac{4\sqrt{10}\epsilon}{\sqrt{1-160\epsilon}} \|a_i b_i\| < 20\epsilon \|a_i b_i\|$. As $\|a_i c'_i\|/\|a_i b_i\| \in [0.355, 0.665]$, we have

$$\|a_i c_i\| \geq \|a_i c'_i\| - \|c_i c'_i\| > 0.355 \|a_i b_i\| - 20\epsilon \|a_i b_i\| > 0.25 \|a_i b_i\|. \quad \square$$

Claim 4.13. *Let $u_i = \text{proj}_{st}(a_i)$ and $v_i = \text{proj}_{st}(b_i)$; see Figure 7. Then, both u_i and v_i lie on segment st . Furthermore, we have $\|su_i\|, \|vt_i\| > \frac{1}{3}\|st\|$ and $\|a_i u_i\|, \|b_i v_i\| \leq 5\sqrt{\epsilon} \cdot \|st\|$.*

Proof. By the charging scheme Ψ_1 , we know that p_i lies on st and $\|sp_i\| \geq (\frac{3}{8} - \frac{1}{50})\|st\|$. Therefore, we have

$$\|su_i\| \geq \|sp_i\| - \|a_i c_i\| > \left(\frac{3}{8} - \frac{1}{50}\right)\|st\| - \frac{1}{\kappa}\|st\| > \frac{1}{3}\|st\|.$$

Similarly we can prove that $\|v_i b\| > \frac{1}{3}\|st\|$, and so both u_i and v_i lie on segment st .

As for the length of $a_i u_i$, let M be the projection matrix onto the hyperplane perpendicular to st , then the triangle inequality combined with $\|a_i b_i\| \leq \frac{1}{\kappa}\|st\|$ yields

$$\begin{aligned} \|a_i u_i\| &= \|M(\overrightarrow{a_i c_i} + \overrightarrow{c_i z_i} + \overrightarrow{z_i p_i} + \overrightarrow{p_i u_i})\| \leq \|M \cdot \overrightarrow{a_i c_i}\| + \|c_i z_i\| + \|z_i p_i\| \\ &\leq 4\sqrt{10}\epsilon \cdot \|a_i b_i\| + \sqrt{\epsilon} \cdot \|a_i b_i\| + \sqrt{\epsilon} \cdot \|st\| < 5\sqrt{\epsilon} \cdot \|st\|. \end{aligned} \quad \square$$

Next, we are going to show that stitching edges $a_i b_i$ and $a_{i+1} b_{i+1}$ via a shortest path $b_i \rightsquigarrow a_{i+1}$ in H_2 does not incur too much error compared to $\pi_{s,t}[b_i, a_{i+1}]$ in E_{light} . Refer to Figure 8.

Claim 4.14. *For any index i , we have*

$$\|c_i d_i\| - \|p_i q_i\| \leq 100\epsilon \|p_i q_i\| \leq (\|z_i w_i\| - \|p_i q_i\|) + 100\epsilon \|p_i q_i\|.$$

Proof. As $\angle(c_i d_i, p_i q_i) \leq 4\sqrt{10}\epsilon$, we have

$$\|c_i d_i\| \leq \frac{\|p_i q_i\|}{\cos(4\sqrt{10}\epsilon)} \leq (1 + 80\epsilon + O(\epsilon^2)) \|p_i q_i\| < (1 + 100\epsilon) \|p_i q_i\|.$$

Hence, we have

$$\|c_i d_i\| - \|p_i q_i\| \leq 100\epsilon \|p_i q_i\| \leq (\|z_i w_i\| - \|p_i q_i\|) + 100\epsilon \|p_i q_i\|. \quad \square$$

Claim 4.15. *For any index i , we have*

$$\|d_i b_i \circ b_i a_{i+1} \circ a_{i+1} c_{i+1}\| - \|q_i p_{i+1}\| \leq 4 \cdot (\|w_i z_{i+1}\| - \|q_i p_{i+1}\|) + 195\epsilon \cdot \|q_i p_{i+1}\|.$$

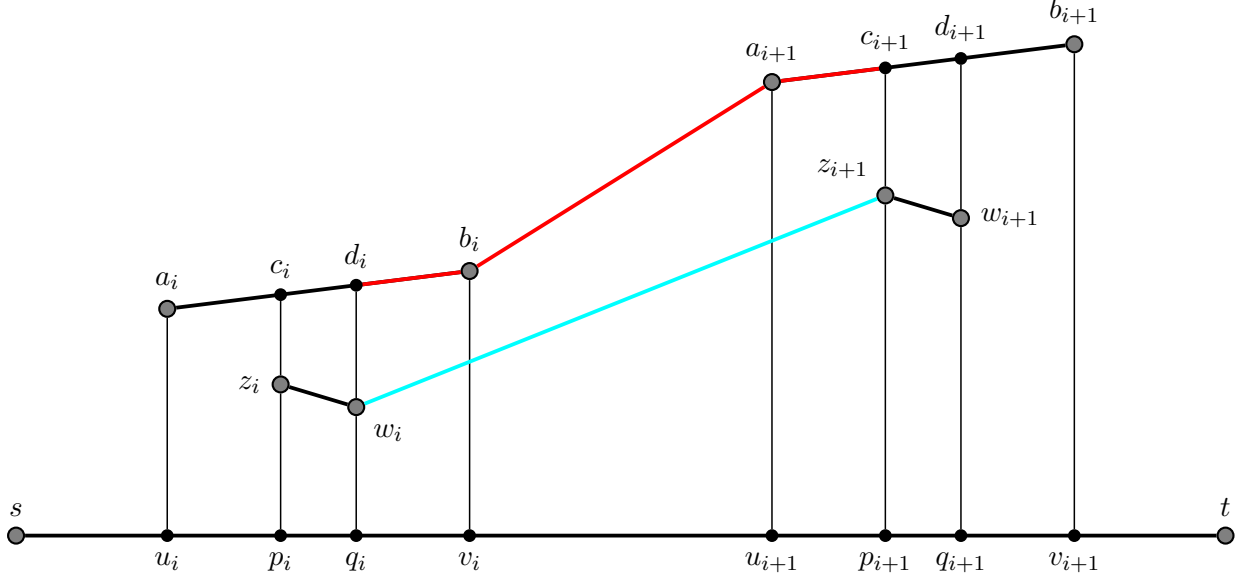


Figure 8: Comparing the error of the stitched path with the error of $\rho_{s,t}$.

Proof. Without loss of generality, assume that $\|a_i b_i\| \geq \|a_{i+1} b_{i+1}\|$. Then, by the design of buffer regions, we know that $\|v_i u_{i+1}\| \geq 2\|a_i b_i\|$, and $\|v_i u_{i+1}\| \leq \|q_i p_{i+1}\| < 2\|v_i u_{i+1}\|$.

Since $\angle(a_i b_i, st)$ and $\angle(a_{i+1} b_{i+1}, st)$ are bounded by $4\sqrt{10}\epsilon$, it follows that

$$\begin{aligned} \|d_i b_i\| - \|q_i v_i\| &\leq \left(\frac{1}{\cos(4\sqrt{10}\epsilon)} - 1 \right) \cdot \|q_i v_i\| \leq (80\epsilon + O(\epsilon^2)) \cdot \|q_i v_i\| \leq 41\epsilon \|q_i p_{i+1}\|, \quad (4) \\ \|a_{i+1} c_{i+1}\| - \|u_{i+1} p_{i+1}\| &\leq \left(\frac{1}{\cos(4\sqrt{10}\epsilon)} - 1 \right) \cdot \|u_{i+1} p_{i+1}\| \\ &\leq (80\epsilon + O(\epsilon^2)) \cdot \|u_{i+1} p_{i+1}\| < 41\epsilon \|q_i p_{i+1}\|. \end{aligned} \quad (5)$$

It remains to compare $\|b_i a_{i+1}\| - \|v_i u_{i+1}\|$ against $\|w_i z_{i+1}\| - \|q_i p_{i+1}\|$. Let M be the matrix of the orthogonal projection onto the hyperplane perpendicular to line st . Define $h_1 = \|M \cdot \overrightarrow{b_i a_{i+1}}\|$ and $h_2 = \|M \cdot \overrightarrow{w_i z_{i+1}}\|$. First, we bound the difference between h_1 and h_2 using the triangle inequality

$$\begin{aligned} |h_1 - h_2| &\leq \left| M \cdot (\overrightarrow{b_i a_{i+1}} - \overrightarrow{w_i z_{i+1}}) \right| \\ &= \left| M \cdot (\overrightarrow{w_i d_i} + \overrightarrow{d_i b_i} + \overrightarrow{a_{i+1} c_{i+1}} + \overrightarrow{c_{i+1} z_{i+1}}) \right| \\ &\leq \sqrt{\epsilon} \cdot \|a_i b_i\| + 4\sqrt{10}\epsilon \cdot \|a_i b_i\| + 4\sqrt{10}\epsilon \cdot \|a_{i+1} b_{i+1}\| + \sqrt{\epsilon} \cdot \|a_{i+1} b_{i+1}\| \\ &< 15\sqrt{\epsilon} \|v_i u_{i+1}\|. \end{aligned}$$

Next, to compare $\|b_i a_{i+1}\| - \|v_i u_{i+1}\|$ against $\|w_i z_{i+1}\| - \|q_i p_{i+1}\|$, we use the Pythagorean theorem

combined with the identity $\sqrt{x} - \sqrt{y} = \frac{x-y}{\sqrt{x}+\sqrt{y}}$ and the definition of h_1 and h_2 :

$$\begin{aligned}
\|b_i a_{i+1}\| - \|v_i u_{i+1}\| &= \sqrt{\|v_i u_{i+1}\|^2 + h_1^2} - \|v_i u_{i+1}\| \\
&\leq \sqrt{\|v_i u_{i+1}\|^2 + 2h_2^2 + 2(h_1 - h_2)^2} - \|v_i u_{i+1}\| \\
&= \frac{2h_2^2 + 2(h_1 - h_2)^2}{\sqrt{\|v_i u_{i+1}\|^2 + 2h_2^2 + 2(h_1 - h_2)^2} + \|v_i u_{i+1}\|} \\
&\leq \frac{2h_2^2 + 225\epsilon \cdot \|v_i u_{i+1}\|^2}{\sqrt{\|v_i u_{i+1}\|^2 + 2h_2^2 + 225\epsilon \cdot \|v_i u_{i+1}\|^2} + \|v_i u_{i+1}\|} \\
&< \frac{2h_2^2}{\sqrt{\frac{1}{4}\|q_i p_{i+1}\|^2 + h_2^2 + \frac{1}{2}\|q_i p_{i+1}\|}} + 112.5\epsilon \cdot \|v_i u_{i+1}\| \\
&< \frac{4h_2^2}{\sqrt{\|q_i p_{i+1}\|^2 + h_2^2} + \|q_i p_{i+1}\|} + 112.5\epsilon \cdot \|v_i u_{i+1}\| \\
&< 4 \cdot (\|w_i z_{i+1}\| - \|q_i p_{i+1}\|) + 113 \cdot \epsilon \|v_i u_{i+1}\|. \tag{6}
\end{aligned}$$

Taking a summation of Equations (4) to (6), we can conclude the proof; see Figure 8 for an illustration. $\square$

Similarly, we can prove an upper bound on the lengths of the prefix and suffix of the path γ .

Claim 4.16. *The following inequalities hold:*

$$\begin{aligned}
\|sa_1 \circ a_1 c_1\| &\leq (1 + 113\epsilon) \|sp_1\|, \\
\|d_\ell b_\ell \circ b_\ell t\| &\leq (1 + 113\epsilon) \|qt\|.
\end{aligned}$$

Proof. Let us focus on the first inequality; the second one can be proved in a symmetric manner. Since $\angle(a_1 b_1, st) \leq 4\sqrt{10}\epsilon$, then we can show that

$$\|a_1 c_1\| - \|u_1 p_1\| \leq \left(\frac{1}{\cos(4\sqrt{10}\epsilon)} - 1 \right) \cdot \|u_1 p_1\| \leq (80\epsilon + O(\epsilon^2)) \|u_1 p_1\| < 100\epsilon \cdot \|u_1 p_1\|. \tag{7}$$

According to Claim 4.13, we have $\|a_1 u_1\| \leq 5\sqrt{\epsilon} \cdot \|st\| \leq 15\sqrt{\epsilon} \|su_1\|$. Therefore, we obtain

$$\|sa_1\| - \|su_1\| < 113\epsilon \cdot \|su_1\|. \tag{8}$$

Adding Equations (7) and (8) finishes the proof. $\square$

Let θ be the polygonal path defined as:

$$\theta = sa_1 \circ a_1 b_1 \circ b_1 a_2 \circ a_2 b_2 \circ \dots \circ b_{\ell-1} a_\ell \circ a_\ell b_\ell \circ b_\ell t$$

from s to t , passing through all edges in $Q = \{a_1 b_1, \dots, a_\ell b_\ell\}$. Taking a summation over all indices $i \in \{1, \dots, \ell\}$ and using Claim 4.14, Claim 4.15 and Claim 4.16, we have

$$\|\theta\| - \|st\| \leq 4(\|\pi_{s,t}\| - \|st\|) + 195\epsilon \cdot \|st\| < 200\epsilon \cdot \|st\|,$$

or equivalently,

$$\|\theta\| \leq (1 + 200\epsilon) \cdot \|st\|. \tag{9}$$

To compare the weight of paths θ and γ , we need to argue that each edge of θ is closely approximated by a shortest path in the current graph H_2 . (Note that we cannot directly apply Corollary 4.1 here, because we are analyzing the state of H_2 during the execution of the second pruning phase, not at the end.)

Claim 4.17. *Any edge e of θ from $\{sa_1, a_1b_1, \dots, a_\ell b_\ell, b_\ell t\}$ has length at most*

$$\|e\| \leq 0.9 \|st\| < \frac{1}{1 + \Delta(\kappa, \delta)} \cdot \|st\|.$$

Proof. By the design of our charging schemes, Ψ_1 and Ψ_2 , the projections of all p_i and q_i are on the segment st ; and more importantly

$$\|sp_i\|, \|tq_i\| \geq \left(\frac{3}{8} - \frac{1}{50}\right) \cdot \|st\| > 0.3 \|st\|.$$

Therefore, we have

$$\|sa_1\| \geq \|sp_1\| - \|u_1p_1\| \geq \|sp_1\| - \|a_1b_1\| \geq 0.3 \|st\| - \frac{1}{\kappa} \|st\| > 0.2 \|st\|.$$

Similarly, we also have $\|b_\ell t\| > 0.2 \|st\|$. Since $\|\theta\| < (1 + 200\epsilon)\|st\|$ by Equation (9), the length of any edge $e \in \{sa_1, a_1b_1, \dots, a_\ell b_\ell, b_\ell t\}$ is bounded by

$$\|e\| < (1 + 200\epsilon - 0.2)\|st\| < 0.9 \|st\| < \frac{1}{1 + \Delta(\kappa, \delta)} \cdot \|st\| \quad ,$$

as claimed. $\square$

According to Corollary 4.1, all edges in the path θ will be preserved up to a stretch factor of $(1 + \delta)(1 + \kappa\delta)(1 + \kappa^2\delta) = 1 + \Delta(\kappa, \delta)$ in H_2 when the second pruning phase finishes. Since the second phase adds edges to E_2 in non-decreasing order of length and by Claim 4.17, each edge in θ is already preserved in H_2 at the time when the algorithm examines edge st .

Therefore, using Claim 4.11 and recalling the definition of γ (cf. Definition 4.3), we have

$$\begin{aligned} \|\gamma\| &= \|\theta\| + (\text{dist}_{H_2}(s, a_1) - \|sa_1\|) + (\text{dist}_{H_2}(t, b_\ell) - \|tb_\ell\|) + \sum_{i=1}^{\ell-1} (\text{dist}_{H_2}(b_i, a_{i+1}) - \|b_i a_{i+1}\|) \\ &\leq \|\theta\| + \Delta(\kappa, \delta) \cdot \left(\|sa_1\| + \|tb_\ell\| + \sum_{i=1}^{\ell-1} \|b_i a_{i+1}\| \right) \\ &= \|\theta\| + \Delta(\kappa, \delta) \cdot \left(\|\theta\| - \sum_{i=1}^{\ell} \|a_i b_i\| \right) \\ &\leq \|\theta\| + \Delta(\kappa, \delta) \cdot \left(\|\theta\| - \frac{1}{320} \|st\| \right) \\ &\leq \left((1 + 200\epsilon)(1 + \Delta(\kappa, \delta)) - \frac{\Delta(\kappa, \delta)}{320} \right) \|st\| \\ &= \left(1 + 200\epsilon + \left(1 + 200\epsilon - \frac{1}{320} \right) \cdot \Delta(\kappa, \delta) \right) \|st\| \\ &\leq \left(1 + \left(1 + 200\epsilon + \frac{200}{(\kappa + 1)^2} - \frac{1}{320} \right) \cdot (\kappa + 1)^2 \delta \right) \|st\| \\ &< (1 + \kappa^2 \delta) \|st\|. \end{aligned}$$

Here we have used $\kappa = 10^4$, Claim 4.1, Equation (9), and the inequalities $\epsilon \leq \delta < \kappa^{-5}$ and $\Delta(\kappa, \delta) < (\kappa + 1)^2 \delta$. This concludes the proof of Lemma 4.2.

5 Fast Implementation

In this section, we provide a fast implementation of our algorithm. For the input spanner, we use the construction of Gudmundson et al. [GLN02]. We follow the original two pruning phases with some modifications. In the first pruning phase, we only add edges in the net-tree spanner of Chan et al. [CGMZ16]. In the second pruning phase, we modify the technique of Das and Narasimhan in [DN97] when checking each type-(ii) edge in E_1 and use the hierarchical net structure in [CGMZ16] to find a helper edge.

5.1 Net-Tree Spanner

An *r -net* of a metric (X, δ) is a subset N of X such that the distance between any two points in N is greater than r and for every point in X , its distance to the closest point in N is at most r . Let Φ be the aspect ratio of X and $r_i = 2^i$ with i being any positive integer. A hierarchy $X = N_0 \supseteq N_1 \supseteq N_2 \supseteq \dots \supseteq N_{\log \Phi}$ is a *hierarchical net* of X if N_{i+1} is a r_{i+1} -net of N_i .

Consider a hierarchical net $X = N_0 \supseteq N_1 \supseteq N_2 \supseteq \dots \supseteq N_{\log \Phi}$ of X . A *net tree* T of X is a tree that connects each point in N_i to its closest net point in N_{i+1} . If a point appears in multiple nets, we treat each appearance as a copy. For each point $u \in N_i$, we use the notation (u, i) for the node of the net tree corresponding to u at level i .

From the net-tree T of X , Chan et al. [CGMZ16] construct a $(1+\epsilon)$ -spanner for X by connecting all pairs of net points in N_i at distance at most $(\frac{4}{\epsilon} + 32)r_i$. We call such pairs *cross edges*. We refer to the spanner construction in [CGMZ16] as a $(1+\epsilon)$ net-tree spanner G of X . We use edges in G to guide our construction. The key to our construction is the *approximate edge* of each pair.

Definition 5.1. For each $uv \in \binom{X}{2}$, let i_{uv} be the lowest level such that there exists a cross edge $u'v'$ such that (u', i_{uv}) and (v', i_{uv}) are the ancestors of $(u, 0)$ and $(v, 0)$, respectively. The edge $u'v'$ is called the *approximate edge* of uv and i_{uv} is the *approximate level* of uv .

We show that the approximate edge of uv has both endpoints close to those of uv . For any graph F , let $V(F)$ and $E(F)$ denote the vertex set and the edge set of F , respectively. For any point $x \in \mathbb{R}^d$ and $r > 0$, let $B(x, r)$ be the Euclidean ball centered at x with radius r .

Claim 5.1. For an edge uv of length r , let $u'v' \in E(G)$ be the approximate edge of uv . Then, $u' \in B(u, \epsilon r)$ and $v' \in B(v, \epsilon r)$.

Proof. Let i be the approximate level of uv and let $(u'', i-1), (v'', i-1)$ be the ancestors of $(u, 0), (v, 0)$ in the net-tree at level $i-1$. The net-tree spanner construction yields $u'v' \in E(G)$. Using the triangle inequality, we get

$$\|uv\| \geq \|u''v''\| - \|uu''\| - \|vv''\| \geq \left(\frac{4}{\epsilon} + 32\right) r_{i-1} - 4 \cdot r_{i-1} \geq \frac{2}{\epsilon} \cdot r_i. \quad (10)$$

Using geometric series, we obtain $\|uu'\| \leq 2 \cdot r_i \leq \epsilon \|uv\|$, consequently $u' \in B(u, \epsilon r)$. Similarly, we can show that $v' \in B(v, \epsilon r)$. $\square$

5.2 Fast Implementation

We are given a set $X \subset \mathbb{R}^d$ be a set of n points, and a sufficiently small $\epsilon > 0$ satisfying Equation (1). We start with a spanner with sparsity $\epsilon^{-O(d)}$ and lightness $\epsilon^{-O(d)}$. Such a spanner can be constructed in $O_{\epsilon,d}(n \log n)$ time [GLN02]. Our construction consists of $k \leq \log^*(d/\epsilon) + O(1)$ iterations of two pruning phases. Assume that at the beginning of the current iteration, we have a $(1+\delta)$ -spanner $H = (X, E)$. We use the same notation for H_1, H_2, E_1 and E_2 as in the original algorithm.

Classification of edges in E . For each edge (s, t) in E , we define the approximate set $A'_{s,t}$ of $A_{s,t}$ as follow: Let h be the approximate level of (s, t) and $N(A_{s,t})$ be the set containing all net points $w \in N_h$ at level h such that $B(w, 2r_h) \cap A_{s,t} \neq \emptyset$. Let $A'_{s,t} = \bigcup_{w \in N(A_{s,t})} B(w, 2r_h)$. Since $r_h \leq \frac{\epsilon}{2} \|st\|$, then for every $x \in A'_{s,t}$, we obtain $\frac{\|s - \text{proj}_{st}(x)\|}{\|st\|} \in [\frac{3}{8} - \frac{1}{50} - \epsilon, \frac{3}{8} + \frac{1}{50} + \epsilon]$ and $\frac{\|t - \text{proj}_{st}(x)\|}{\|st\|} \in [\frac{5}{8} - \frac{1}{50} - \epsilon, \frac{5}{8} + \frac{1}{50} + \epsilon]$. Similarly, we define $B'_{s,t}$ based on the region $B_{s,t}$. An edge (s, t) is type-(i) if $A'_{s,t}$ or $B'_{s,t}$ is empty. Otherwise, (s, t) is a type-(ii) edge. We also denote the set of type-(i) and type-(ii) edges by $E^{(i)}$ and $E^{(ii)}$, respectively.

First pruning phase. Recall that for every pair $\{x, y\} \subset X$, if $|P_{x,y}| \geq \frac{\alpha}{2^i \kappa}$ (for each sub-iteration i), we delete every edge in $P_{x,y}$. However, since the total number of pairs we need to check is $\Theta(n^2)$, this step would lead to a quadratic running time for each iteration. Thus, instead of considering all pairs of vertices $\{x, y\} \subset X$, we only consider the edges in the net-tree spanner G . Assuming that $\|xy\| \geq \beta^j/25$, we define an approximate set

$$P'_{x,y} = \left\{ st \in L_j \cap E_1 \cap E^{(i)} : \|sx\| + \|xy\| + \|yt\| \leq (1 + 5\epsilon)\|st\| \right\}. \quad (11)$$

The difference between $P_{x,y}$ and $P'_{x,y}$ is the distortion $(1 + \epsilon$ versus $1 + 5\epsilon)$. If $|P'_{x,y}| \geq \frac{\alpha}{2^i \kappa}$, then add xy to E_1 as a new edge, and remove all type-(i) edges in $P'_{x,y}$ from E_1 .

Second pruning phase. For each type-(ii) edge in E_1 , we check whether the distance between two endpoints is already preserved by edges in E_2 . This can be done in nearly linear time using the technique in [DN97]. Our method is similar to [DN97], except we only consider each edge for $\Theta(\log n)$ levels before contracting it.

Let $E_{1,i} = \{st \in E_1 : 2^i \leq \|st\| \leq 2^{i+1}\}$ and $E_{2,i} = \{st \in E_2 : 2^i \leq \|st\| \leq 2^{i+1}\}$ for $i \in [0, \log \Phi + 1]$, $E_{2,<i}$ be the union of $E_{2,0}, E_{2,1}, \dots, E_{2,i-1}$ and $H_{2,<i} = (X, E_{2,<i})$. For each i , we build a cluster graph F_i with vertex set X . We greedily construct the set of balls with radius $\epsilon 2^i$ in $H_{2,<i}$ (each ball is a cluster) such that the distance between any two centers in $H_{2,<i}$ is at least $\epsilon 2^i$ and the set of balls covers X . Let $\{C_1, C_2, \dots\}$ be the set of clusters. Note that one point might belong to multiple clusters. There are two types of edges in F_i : inter-cluster and intra-cluster. Intra-cluster edges are edges between a point u and a center v of a cluster C such that $u \in C$. The weight of uv is $\text{dist}_{H_2}(u, v)$. We add all intra-cluster edges to F_i . An inter-cluster edge is an edge between two cluster centers. There is an edge between two cluster centers v_j and $v_{j'}$ if and only if $\text{dist}_{H_2}(v_j, v_{j'}) \leq 2^i$ or there exists an edge in $E_{2,<i}$ from a point in C_j to a point in $C_{j'}$, where C_j and $C_{j'}$ are the clusters corresponding to the centers v_j and $v_{j'}$. The inter-cluster edge $v_j v_{j'}$ has weight $\text{dist}_{H_2}(v_j, v_{j'})$ in the first case and $\min_{u \in C_j, w \in C_{j'}, uw \in E_2} (\text{dist}_{H_2}(u, v_j) + \text{dist}_{H_2}(w, v_{j'}) + \|uw\|)$ in the second case. We use Dijkstra's algorithm to compute all the intra-cluster and inter-cluster edges.

Our implementation of the second pruning phase runs in $\log \Phi + 1$ iterations. At iteration i , we first build the cluster graph F_i from $H_{2,<i}$. We consider edges in $E_{1,i}$ in increasing order of length. For each $st \in E_{1,i}$, if st is not type-(ii), we simply add st to E_2 . Otherwise, we check all paths from s to t within $O(1)$ hops in F_i that contains at most 2 intra-cluster edges. If the length of the shortest of those paths is less than or equal to $(1 + \kappa^2 \delta)(1 + \epsilon)\|st\|$, skip that edge. Otherwise, we add st and its helper edge to E_2 . After an edge st is added to E_2 , we add inter-cluster edges between the centers of all clusters containing u to the centers of all clusters containing v . For any center c_s of a cluster containing s and any center c_t of a cluster containing t , we add an edge $c_s c_t$ of weight $\text{dist}_{F_i}(c_s, s) + \|st\| + \text{dist}_{F_i}(t, c_t)$.

However, computing all intra-cluster edges for all F_i is expensive since one will have to run a single source shortest path (SSSP) algorithm for all cluster centers. Instead, we contract all edges of lengths less than or equal to $\frac{2^i \epsilon^2}{n}$. The total weight of all those edges in any path in F_i is $2^i \epsilon^2$. Hence, for every path of length at least 2^i , the total weight of contracted edges is significantly smaller compared to the total length. We then construct the graph F'_i similar to F_i . Let $i' = i - \log(n\epsilon^{-2})$ and $E'_{2,<i} = \bigcup_{k=i'}^{i-1} E_{2,k}$, we compute the graph F'_i as follow: first, contract all edges in $E_{2,<i'}$ and second, compute F'_i from $E'_{2,<i}$ similar to F_i . Instead of checking $\text{dist}_{E_2}(u, v) \leq (1 + \kappa^2 \delta) \|uv\|$ as in the original algorithm, we check whether $\text{dist}_{F'_i}(s, t) + \epsilon^2 2^i \leq (1 + \epsilon)(1 + \kappa^2 \delta) \|st\|$. If true, we skip the edge st . Otherwise, we keep st and add its approximate helper edge to E_2 . The approximate helper edge is any edge between a net point in $N(A_{s,t})$ to a net point in $N(B_{s,t})$.

After the iteration (including two pruning phases), we update $\delta \leftarrow \Delta'(\kappa, \delta) = (1 + \delta)(1 + \kappa \delta)(1 + 5\epsilon)(1 + \kappa^2 \delta)(1 + \epsilon) - 1$ and $\alpha \leftarrow \log \alpha$.

5.3 Analysis

5.3.1 Stretch

We prove that Claim 4.2 still holds in our implementation. After the first pruning phase, for any edge $st \in E$, we are guaranteed that $\text{dist}_{H_1} \leq (1 + \kappa \delta)(1 + 5\epsilon) \cdot \|st\|$. Throughout the execution of the second phase, for any edge $st \in E_1 \cap E$, we have $\text{dist}_{H_2}(s, t) \leq (1 + \kappa^2 \delta)(1 + \epsilon) \cdot \|st\|$.

Claim 5.2. *Throughout the execution of the first pruning phase, for any edge $st \in E$, we are guaranteed that $\text{dist}_{H_1}(s, t) \leq (1 + \kappa \delta)(1 + 5\epsilon) \cdot \|st\|$. Also, throughout the execution of the second phase, for any edge $st \in E_1 \cap E$, we have $\text{dist}_{H_2}(s, t) \leq (1 + \kappa^2 \delta)(1 + \epsilon) \cdot \|st\|$.*

The bound on the stretch after the first pruning phase can be proven using the same argument as in Claim 4.2. We then focus on the second pruning phase. Claim 5.3 shows that if there is a good approximation path from s to t in the cluster graph F_i , there is a good approximation path from s to t in the current greedy spanner of E_2 .

Claim 5.3 (Lemma 3.2 and Lemma 3.3 [DN97]). *Let st be a type-(ii) edge in E_1 such that $2^i \leq \|st\| \leq 2^{i+1}$ for some i and F_i be the cluster graph constructed above. Let P be the shortest path from s to t in H_2 and P_C be the shortest path from s to t in F_h . Then, $1 \leq \frac{\|P_C\|}{\|P\|} \leq 1 + \Theta(\epsilon)$.*

We observe that the distance in F'_i is approximately the distance in F_i .

Observation 5.1. *For all $x, y \in X$, we have $\text{dist}_{F'_i}(s, t) \leq \text{dist}_{F_i}(s, t) \leq \text{dist}_{F'_i}(s, t) + 2^i \epsilon^2$.*

Now Claim 5.3, combined with Observation 5.1, implies the following.

Claim 5.4. *Let s and t be two points such that $2^i \leq \|st\| \leq 2^{i+1}$, and Let F'_i be the cluster graph constructed above. Let P the shortest path from s to t in $H_2 = (X, E_2)$ and P'_C be the shortest path from s to t in F'_i . Then $1 \leq \frac{\|P'_C\| + 2^i \epsilon^2}{\|P\|} \leq 1 + \epsilon$.*

Proof. Let P_C be the shortest path from s to t in F_i . By Observation 5.1, we have $\|P'_C\| \leq \|P_C\| \leq \|P'_C\| + 2^i \epsilon^2$. Combined with Claim 5.3, we obtain $1 \leq \frac{\|P'_C\| + 2^i \epsilon^2}{\|P\|} \leq 1 + \Theta(\epsilon)$. The last term $\Theta(\epsilon)$ can be reduced to ϵ by scaling the cluster radius by a suitable constant. $\square$

We have the following corollary:

Corollary 5.1. *Let st be a type-(ii) edge in E_1 such that $2^i \leq \|st\| \leq 2^{i+1}$. If $\text{dist}_{H_2}(s, t) \leq (1 + \kappa^2 \delta) \|st\|$, then st is not added to E_2 .*

Proof. By Claim 5.4, the path from s to t in F'_i has length at most $(1 + \epsilon)\text{dist}_{H_2}(s, t) - \epsilon^2 2^i \leq (1 + \kappa^2 \delta)(1 + \epsilon)\|st\| - \epsilon^2 2^i$. Hence, st is not added to E_2 by our algorithm. $\square$

Then, for any edge st not added to E_2 , the distance from s to t in the spanner is at most $(1 + \kappa^2 \delta)(1 + \epsilon)\|st\| + \epsilon^2 2^i$. By proper scale of the cluster radius, we got $\text{dist}_{H_2}(s, t) \leq (1 + \kappa^2 \delta)(1 + \epsilon)\|st\|$. This completes the proof of Claim 5.2.

Recall that after each iteration, we update $\delta = \Delta'(\kappa, \delta)$ with $\Delta'(\kappa, \delta) = (1 + \delta)(1 + \kappa \delta)(1 + 5\epsilon)(1 + \kappa^2 \delta)(1 + \epsilon) - 1$. Observe that

$$\begin{aligned} \Delta'(\kappa, \delta) &\leq (1 + \epsilon)(1 + 5\epsilon)(1 + \delta)(1 + \kappa \delta)(1 + \kappa^2 \delta) - 1 \\ &\leq (1 + 7\epsilon)(1 + \delta)(1 + \kappa \delta)(1 + \kappa^2 \delta) - 1 \leq (\kappa + 1)^2 \delta \end{aligned} \quad (12)$$

for sufficiently small ϵ . Therefore, after the update at the end of the i -th iteration, we have $\delta \leq (\kappa + 1)^{2i} \epsilon$. After $O(\log^*(d/\epsilon))$ iterations, the stretch is bounded by $(\kappa + 1)^{O(\log^*(d/\epsilon))} \epsilon \leq \kappa^{-5}$.

5.3.2 Sparsity

The charging scheme for sparsity is similar to Ψ_0 . The only change we make for a fast implementation, compared to the original Ψ_0 , is that we use $A'_{s,t}$ and $B'_{s,t}$ instead of $A_{s,t}$ and $B_{s,t}$. For each edge $st \in E$, let $\pi_{s,t}$ be a spanner path in E_{spa} between s, t . If st is a type-(i) edge, then $A'_{s,t}$ or $B'_{s,t}$ is empty, implying that $A_{s,t}$ or $B_{s,t}$ is empty since $A'_{s,t}$ and $B'_{s,t}$ contain $A_{s,t}$ and $B_{s,t}$, respectively. Therefore, there must be a single edge e in $\pi_{s,t}$ that crosses the region $A_{s,t}$ or $B_{s,t}$. We charge the edge st to e . If st is a type-(ii) edge, there are two cases:

1. If either $\pi_{s,t} \cap A'_{s,t}$ or $\pi_{s,t} \cap B'_{s,t}$ is empty. In that case, there is an edge $e = s't'$ on $\pi_{s,t}$ that crosses either $A'_{s,t}$ or $B'_{s,t}$ (and hence cross either $A_{s,t}$ or $B_{s,t}$). If e only crosses one of the two regions (say $A_{s,t}$), then we have

$$\begin{aligned} \|\text{proj}_{st}(s) - \text{proj}_{st}(s')\| &< \left(\frac{3}{8} - \frac{1}{50}\right) \cdot \|st\|, \\ \left(\frac{3}{8} - \frac{1}{50}\right) \cdot \|st\| &< \|\text{proj}_{st}(t') - \text{proj}_{st}(t)\| < \left(\frac{5}{8} - \frac{1}{50}\right) \cdot \|st\|. \end{aligned}$$

Recall that for each edge $e \in E_{\text{spa}}$, we divide e evenly into κ sub-segments by adding at most $\kappa - 1$ Steiner points on e and Y is the union of X and the set of added Steiner points. Let $z \in Y \cap e$ be the Steiner points in $A_{s,t}$ on segment e that is closest to s' ; such a point z must exist since each sub-segment of e has length at most $\frac{\|e\|}{\kappa} < \frac{\|st\|}{25}$. Then, charge st to segment zt' which has length at least $(\frac{1}{25} - \frac{1}{\kappa})\|st\| > \frac{\|st\|}{26}$.

If e crosses both regions $A_{s,t}, B_{s,t}$, then let $z_1 \in Y \cap e$ be the Steiner point in $A_{s,t}$ which is the closest one from s' , and let $z_2 \in Y \cap e$ be the Steiner point in $B_{s,t}$ which is the closest one from t' . Then, charge st to segment $z_1 z_2$ which has length at least $(\frac{1}{4} + \frac{1}{25} - \frac{2}{\kappa})\|st\| > \frac{\|st\|}{4}$.

2. Otherwise, we charge st fractionally to set of edges in E_{spa} . Move along $\pi_{s,t}$ from s to t and let p be the last vertex in $A'_{s,t}$ and let q be the first vertex in $B'_{s,t}$. As $\|\pi_{s,t}\| \leq (1 + \epsilon) \cdot \|st\|$, we know that

$$\begin{aligned} \|\pi_{s,t}[p, q]\| &\leq \|\text{proj}_{st}(p) - \text{proj}_{st}(q)\| + \epsilon \cdot \|st\| \\ &\leq \|\text{proj}_{st}(p) - \text{proj}_{st}(q)\| + \epsilon \cdot \frac{1}{\left(\frac{5}{8} - \frac{1}{50} - \epsilon\right) - \left(\frac{3}{8} + \frac{1}{50} + \epsilon\right)} \cdot \|\text{proj}_{st}(p) - \text{proj}_{st}(q)\| \\ &\leq (1 + 10\epsilon) \cdot \|\text{proj}_{st}(p) - \text{proj}_{st}(q)\|. \end{aligned}$$

for sufficiently small ϵ . Therefore, Lemma 3.1 yields

$$\|E(\pi_{s,t}[p, q], st, 2\sqrt{10}\epsilon)\| > 0.5 \cdot \|\text{proj}_{st}(p) - \text{proj}_{st}(q)\|.$$

Then, for each edge $e \in E(\pi_{s,t}[p, q], st, 2\sqrt{10}\epsilon)$, charge a fraction of $\frac{2\|e\|}{\|\text{proj}_{st}(p) - \text{proj}_{st}(q)\|}$ of edge st to edge e .

We then have our modified Claim 4.3.

Claim 5.5. *If a type-(ii) edge st charges to an edge $xy \in E_{\text{spa}}^Y$, then angle $\angle(st, xy)$ is at most $15\sqrt{\epsilon}$. Plus, the projection $\text{proj}_{st}(z)$ of z onto line st lies in the segment st and satisfies $\frac{\|s - \text{proj}_{st}(z)\|}{\|st\|} \in [\frac{3}{8} - \frac{1}{50} - \epsilon, \frac{5}{8} + \frac{1}{50} + \epsilon]$ for all $z \in \{x, y\}$*

The difference between Claim 4.3 and Claim 5.5 is the region of the projection of each edge that st charges to. The region changes from $[\frac{3}{8} - \frac{1}{50}, \frac{5}{8} + \frac{1}{50}]$ to $[\frac{3}{8} - \frac{1}{50} - \epsilon, \frac{5}{8} + \frac{1}{50} + \epsilon]$. This is due to the design of our algorithm.

For sufficiently small ϵ , Claim 4.4 still holds for our algorithm. We now show that Claim 4.5 also holds.

Claim 5.6. *During the first pruning phase, the number of new edges added to E_1 is at most $O(|E_{\text{spa}}| \log \alpha)$. After the first pruning phase, the number of type-(i) edges in E_1 is at most $O(|E_{\text{spa}}|)$.*

Proof. We show by induction that during the first pruning phase, by the beginning of the i -th sub-iteration, $|E_1 \cap E^{(i)}|$ is at most $|E_{\text{spa}}| \alpha / 2^{i-1}$. Suppose otherwise for the sake of contradiction. Using the same argument as in the proof of Claim 4.5, we obtain that there exists a set F of more than $\alpha / 2^{i-1}$ edges charged to the same edge $e = xy$ in E_{spa} and there is an index j such that at least $\frac{\alpha}{2^i \kappa}$ edges in L_j have been charged to e . Let $x'y'$ be the approximate edge of xy . We show that $P'_{x',y'} \supseteq F \cap L_j$. For any edge $st \in F \cap L_j$, by the charging scheme, we get

$$\|sx\| + \|xy\| + \|yt\| \leq (1 + \epsilon)\|st\|.$$

By Claim 5.1, we have $\|xx'\|, \|yy'\| \leq \epsilon\|xy\|$. Then, using the triangle inequality, we have

$$\begin{aligned} \|sx'\| + \|x'y'\| + \|y't\| &\leq \|sx\| + \|xy\| + \|yt\| + 2\|xx'\| + 2\|yy'\| \\ &\leq (1 + \epsilon)\|st\| + 4\epsilon\|xy\| \leq (1 + 5\epsilon)\|st\|. \end{aligned}$$

Thus, the edge $x'y'$ must be added to our spanner and hence, all edges in $F \cap L_j$ are removed, which is a contradiction. $\square$

Observation 4.1 also holds in our construction. We then prove an analogue of Lemma 4.1.

Lemma 5.1. *Fix any edge $e \in E_{\text{spa}}^Y$ and level index $j \geq 0$. Then, after the second pruning phase, there is at most one type-(ii) edge in $E_2 \cap L_j$ that is charged to e .*

Proof. Assume that two type-(ii) edges $s_1t_1, s_2t_2 \in E_2 \cap L_j$ are charged to the same edge $e \in E_{\text{spa}}^Y$; and w.l.o.g. $\|s_1t_1\| \leq \|s_2t_2\|$. We prove that s_2t_2 could not have been added to E_2 since there is already a good approximation path from s_2 to t_2 at the time we consider s_2t_2 . Let r be an endpoint of e , and let c, w be the projections of a, r on s_1t_1 , and let f, g, h be the projections of a, r, b on s_2t_2 and p, q be the projections of c, w on s_2t_2 . Let D be the length of s_1t_1 , by the design of our algorithm, we still have $\|cd\| \leq (0.29 + 2\epsilon)D$ and $\angle(s_1t_1, s_2t_2) \leq 30\sqrt{\epsilon}$. Since s_2t_2

also charges to e , we have $\|s_2g\|/\|s_2t_2\| \in [\frac{3}{8} - \frac{1}{50} - \epsilon, \frac{5}{8} + \frac{1}{50} + \epsilon]$. Using the same argument as in the proof of Lemma 4.1, we have that $\|s_2a\| \leq \|s_2f\| + 2170\epsilon D$, $\|bt_2\| \leq \|ht_2\| + 2170\epsilon D$ and $\|ab\| \leq \|fh\| + 120\epsilon D$. Then, we also have:

$$\text{dist}_{H_2}(s_2, a) + \|ab\| + \text{dist}_{H_2}(b, t_2) < (1 + \kappa^2\delta)\|s_2t_2\|.$$

Therefore, (s_2t_2) cannot be added to E_2 due to the design of our algorithm. $\square$

Corollary 4.2 is then carries over. We conclude that the total number of edges of our spanner after a single iteration is $O(|E_{\text{spa}}| \log \alpha)$. Therefore, after k iterations, the output spanner has $O(\log^{(k)}(\epsilon^{-O(d)}|E_{\text{spa}}|)) = O((\log^{(k)}(1/\epsilon) + \log^{(k-1)}(d))|E_{\text{spa}}|)$ edges.

5.3.3 Lightness

For the lightness, we keep the same charging schemes, Ψ_1 and Ψ_2 , only changing regions $A_{s,t}$ and $B_{s,t}$ to $A'_{s,t}$ and $B'_{s,t}$, respectively. Using the same arguments as in the proof of sparsity, Claim 4.6, Claim 4.7 and Claim 4.8 still hold. It remains to prove that each edge $e \in E_{\text{light}}^Y$ receives $O(\|e\|)$ charges under the charging scheme Ψ_2 . For any single edge $st \in E$, we also let $F \subseteq E_{\text{light}}^Y$ be the set of edges that st charged to under the charging scheme Ψ_1 .

Claim 5.7. $\|F\| > \frac{1}{10}\|st\|$.

Proof. The projection of F onto line st is $\text{proj}_{st}(F)$, whose length is at least $\frac{1}{2}(\frac{1}{4} - \frac{1}{25} - 2\epsilon)\|st\| > \frac{1}{10}\|st\|$. $\square$

Let S be the subset of edges that are already charged by $\lceil \frac{\alpha}{2t} \rceil$ times by edges before st . If $\|S\| \leq 0.5\|F\|$, we re-distribute the charge similar to the original proof. We now consider the case when $\|S\| > 0.5\|F\|$. We now prove Lemma 4.2 for our implementation. Using the same proof as Claim 4.10, for each edge e in S , there exists an edge $\chi(e)$ charging to e such that $\frac{1}{\kappa}\|st\| \geq \|\chi(e)\| \geq \kappa\|e\|$. We define the set $P = \{\chi(e) : e \in S\}$ and construct the set $Q = \{a_1b_1, a_2b_2, \dots, a_\ell b_\ell\}$ similar to the proof of Lemma 4.2. We obtain $\|Q\| \geq \frac{1}{320}\|st\|$ by using the same argument as in the proof of Claim 4.11.

For each edge a_ib_i , let $e_i = z_iw_i$ be the corresponding edge in S that a_ib_i charged to and $\chi(e_i) = a_ib_i$. Let p_i (q_i) and c_i (d_i) be the intersection of the hyperplane perpendicular to st through z_i (w_i) with st and a_ib_i . We then prove Claim 4.12.

Claim 5.8. *Both points c_i and d_i lie on the segment a_ib_i ; furthermore, $\min\{\|a_ic_i\|, \|d_ib_i\|\} \geq 0.25\|a_ib_i\|$.*

Proof. Let c'_i be the projection of z_i on a_ib_i . Since $\angle(a_ib_i, st) \leq 2\sqrt{10}\epsilon$, we also obtain $\|c_ic'_i\| < 10\epsilon\|a_ib_i\|$ as in the proof of Claim 4.12. Since $\|a_ic'_i\|/\|a_ib_i\| \in [0.355 - \epsilon, 0.655 + \epsilon]$, we have

$$\|a_ic_i\| \geq \|a_ic'_i\| - \|c_ic'_i\| > (0.355 - \epsilon)\|a_ib_i\| - 10\epsilon\|a_ib_i\| > 0.25\|a_ib_i\|.$$

Similarly, $\|d_ib_i\| \geq 0.25\|a_ib_i\|$. $\square$

We then prove an analogue of Claim 4.13.

Claim 5.9. *Let $u_i = \text{proj}_{st}(a_i)$, $v_i = \text{proj}_{st}(b_i)$. Then, both u_i and v_i lie on segment st . Furthermore, $\|su_i\|, \|vt_i\| > \frac{1}{3}\|st\|$, and $\|a_iu_i\|, \|b_iv_i\| \leq 5\sqrt{\epsilon} \cdot \|st\|$.*

Proof. We have $\|sp_i\| \geq (\frac{3}{8} - \frac{1}{50} - \epsilon) \|st\|$. Therefore,

$$\|su_i\| \geq \|sp_i\| - \|a_i c_i\| > \left(\frac{3}{8} - \frac{1}{50} - \epsilon\right) \|st\| - \frac{1}{\kappa} \|st\| > \frac{1}{3} \|st\|.$$

Similarly, $\|v_i b\| > \frac{1}{3} \|st\|$, implying that u_i and v_i lie on segment st . Using the same argument as in the proof of Claim 4.13, we obtain $\|a_i u_i\| \leq 5\sqrt{\epsilon} \cdot \|st\|$. $\square$

The proof of Claim 4.14, Claim 4.15 and Claim 4.16 carry over. Using the proof of Lemma 4.2, we obtain $\text{dist}_{H_2}(s, t) \leq (1 + \kappa^2 \delta)$, implying that st is not added to E_2 , a contradiction.

Therefore, the total weight of the spanner after a single iteration is $O(\|E_{\text{light}}\| \log \alpha)$. After k iterations, the total weight is $O(\log^{(k)}(\epsilon^{-O(d)})) \cdot \|E_{\text{light}}\| = O(\log^{(k)}(1/\epsilon) + \log^{(k-1)}(d)) \cdot \|E_{\text{light}}\|$.

5.3.4 Running Time

We first show that for each edge st , we can determine whether st is type-(i) or type-(ii) in constant time.

Claim 5.10. *For every edge st , determining whether st is type-(i) or type-(ii) can be done in $\epsilon^{-O(d)}$ time.*

Proof. Let h be the approximate level of st . Consider the net N_h at level h . Recall that $A'_{s,t} = \bigcup_{w \in N(A_{s,t})} B(w, r_h)$, where $N(A_{s,t})$ is the set of net points $w \in N_h$ such that $B(w, r_h) \cap A_{s,t} \neq \emptyset$. Then, to determine whether an edge is type-(i) or type-(ii), we find all the net point $w \in N_h$ such that $B(w, r_h) \cap A_{s,t} \neq \emptyset$ or $B(w, r_h) \cap B_{s,t} \neq \emptyset$. To find them, we first find the ancestor (s', h) of $(s, 0)$. Then, observe that the ellipsoid $\Gamma_{s,t}$ is in $B(s', 2\|st\|)$. By the packing bound, there are $\epsilon^{-O(d)}$ net points in $B(s', 2\|st\|)$. For each net point w , checking whether $B(w, r_h) \cap A_{s,t} \neq \emptyset$ or $B(w, r_h) \cap B_{s,t} \neq \emptyset$ takes constant time. Thus, the total running time of determining the type of an edge is $\epsilon^{-O(d)}$. $\square$

The net-tree spanner G can be constructed in $O_{\epsilon,d}(n \log n)$ time [HPM06]. The first pruning phase can be implemented in $\epsilon^{-O(d)}n$ time as follows: Start with a spanner that has sparsity $\epsilon^{-O(d)}$ and lightness $\epsilon^{-O(d)}$, for each edge xy in G with $\|xy\| \geq \beta^i/25$, we find all type-(i) edges st in L_j such that $\|sx\| + \|xy\| + \|yt\| \leq (1 + 5\epsilon)\|st\|$. We compute $|P'_{x,y}|$ as follows: For each type-(i) edge st , we find all edge $x'y'$ in G such that $\|x'y'\| \geq \beta^j/25$ and $st \in P'_{x',y'}$. Let $\gamma = \log \beta^j/25$, and observe that any cross edge $x'y' \in E(G)$ satisfying $st \in P'_{x',y'}$ can only belong to $O(\log \epsilon^{-1})$ levels of the net-tree, between levels $\gamma - \log \epsilon^{-1}$ and $\gamma + \log \epsilon^{-1}$. Furthermore, any of those cross edges must have two endpoints in the ellipsoid with foci s and t and focal distance $(1 + 5\epsilon)\|st\|$. By the packing bound, for each level from $\gamma - \log \epsilon^{-1}$ to $\gamma + \log \epsilon^{-1}$, there are $\epsilon^{-O(d)}$ net points inside such an ellipsoid. Thus, the total number of edges we need to check is at most $\epsilon^{-O(d)}$. The overall running time of the first pruning phases over the entire algorithm is $\epsilon^{-O(d)}n \cdot \log^*(d/\epsilon)$.

Then, we prove that the second pruning phase can be implemented in $O_{\epsilon,d}(n \log^2 n)$ time. Assume that we are at the i -th checking iteration. All intra-cluster and inter-cluster edges in F'_i are computed by running Dijkstra's algorithm. For each cluster center v , we run Dijkstra's algorithm to find all shortest paths from v with length at most 2^i . Let $H'_{2,<i}$ be the graph $H_{2,<i}$ after contracting all edges of length at most $\frac{2^i \epsilon^2}{n}$. For any two vertices u and v for which $uv \in E(F'_i)$, let $\mathbf{w}(u, v)$ be the weight of uv in F'_i . Then, we bound the time required to compute those single source shortest path trees. We show that each point belongs to at most $\epsilon^{-O(d)}$ trees of maximum distance $O(2^i)$.

Lemma 5.2. *Each point in F'_i belongs to at most $\epsilon^{-O(d)}$ single source shortest path trees.*

Proof. Recall that $E_{2,<i}$ is the set of edges with weight within $[1, 2^i)$, $E'_{2,<i}$ is the set of edges with weight within $[\frac{\epsilon^2 2^i}{n}, 2^i)$ and $H_{2,<i} = (X, E_{2,<i})$. For any vertex $u \in V(H'_{2,<i})$, let $M(u)$ be the set of points in X that are contracted to u . For any point u , let $N(u)$ be the set of cluster centers s such that $\text{dist}_{H'_{2,<i}}(s, u) \leq 2^i$. We prove that $|N(u)| = \epsilon^{-O(d)}$ for every $u \in V(H'_{2,<i})$.

Let s be an arbitrary vertex in $N(u)$. We prove that for every $u' \in M(u)$ and $s' \in M(s)$, $\text{dist}_{H'_{2,<i}}(s', u') \leq 2^{i+1}$. Observe that there exist some $u_0 \in M(u)$ and $s_0 \in M(s)$ such that $\text{dist}_{H_{2,<i}}(u_0, v_0) \leq 2^i$. Furthermore, since the path from u' to u_0 has at most n contracted edges, $\text{dist}_{H_{2,<i}}(u', u_0) \leq n \cdot \frac{\epsilon^2 2^i}{n} = \epsilon^2 2^i$. Similarly, $\text{dist}_{H_{2,<i}}(s', s_0) \leq \epsilon^2 2^i$. Thus, $\text{dist}_{H_{2,<i}}(u', s') \leq 2^i + 2\epsilon^2 2^i \leq 2^{i+1}$. Since $\text{dist}_{H_{2,<i}}(u', s') \geq \|u's'\|$, we have $\|u's'\| \leq 2^{i+1}$.

We continue with bounding the distance between two points in $N(u)$.

Claim 5.11. *Let s_1 and s_2 be two arbitrary vertices in $N(u)$. For any $s'_1 \in M(s_1)$ and $s'_2 \in M(s_2)$, $\|s'_1 s'_2\| \geq \frac{\epsilon^2 2^i}{1 + (\kappa + 1)^2 \delta}$.*

Proof. From our construction of clusters, we have $\text{dist}_{H_{2,<i}}(s'_1, s'_2) \geq \text{dist}_{H'_{2,<i}}(s_1, s_2) \geq \epsilon 2^i$. Suppose, to the contrary, that $\|s'_1 s'_2\| < \frac{\epsilon^2 2^i}{1 + (\kappa + 1)^2 \delta}$. By the stretch argument, we have $\text{dist}_{H_2}(s'_1, s'_2) \leq \frac{\epsilon^2 2^i}{1 + (\kappa + 1)^2 \delta} \cdot (1 + (\kappa + 1)^2 \delta) = \epsilon^2 2^i$. Hence, the shortest path from s'_1 to s'_2 contains only edges of length at most $\epsilon^2 2^i$, implying that $\text{dist}_{H_2}(s'_1, s'_2) = \text{dist}_{H_{2,<i}}(s'_1, s'_2)$. Since the distance in the contract graph $H'_{2,i}$ is dominated by the distance in $H_{2,<i}$, we have $\text{dist}_{H'_{2,<i}}(s_1, s_2) \leq \text{dist}_{H_{2,<i}}(s'_1, s'_2) \leq \epsilon^2 2^i$. However, by our construction, $\text{dist}_{H'_{2,<i}}(s_1, s_2) > \epsilon^2 2^i$, a contradiction. $\square$

Let $N(u) = \{s_1, s_2, \dots, s_l\}$ and $s'_1, s'_2, \dots, s'_l$ be arbitrary vertices in $M(s_1), M(s_2), \dots, M(s_l)$, respectively. By Claim 5.11, $\|s'_{h_1} s'_{h_2}\| \geq \frac{\epsilon^2 2^i}{1 + (\kappa + 1)^2 \delta}$ for any $h_1 \neq h_2$. On the other hand, $\|us_h\| \leq 2^{i+1}$ for any $h \in [1, l]$. Therefore, by the packing bound, we have

$$|N(u)| = l \leq \left(\frac{2^{i+1}}{\epsilon^2 2^i / (1 + (\kappa + 1)^2 \delta)} \right)^{-O(d)} = \epsilon^{-O(d)}.$$

This completes the proof of Lemma 5.2. $\square$

Lemma 5.2 implies that each edge is considered at most $\epsilon^{-O(d)}$ times in the construction of all single source shortest path trees. Since the running time of Dijkstra's algorithm for a connected graph of m edges is bounded by $O(m \log m)$, the total construction time of all graphs F'_i is $\epsilon^{-O(d)} n \log(n/\epsilon^2) \log n = \epsilon^{-O(d)} n \log^2 n$.

For each query st , we prove that the shortest path from s to t in F'_i contains a constant number of edges. Here, we abuse the notation of s and t for vertices in F'_i .

Claim 5.12. *For any two points s and t in X such that $2^i \leq \|st\| < 2^{i+1}$, if $\text{dist}_{F'_i}(s, t) \leq (1 + \epsilon)(1 + \kappa^2 \delta)\|st\|$, then there exists a shortest path P'_C from s to t in F'_i comprising a constant number of edges. Furthermore, only the first and the last edge in P'_C are intra-cluster.*

Proof. Let $P'_C = \langle s = v_1, v_2, \dots, v_l = t \rangle$ be a shortest path from s to t in F'_i with the least number of intra-cluster edges. Observe that P'_C contains at most 2 intra-cluster edges. Otherwise, if there is an intra-cluster edge $v_h v_{h+1}$ such that $1 < h < l$, then v_h or v_{h+1} is not a cluster center. Assume that v_{h+1} is not a cluster center. Hence, $v_{h+1} v_{h+2}$ is also an intra-cluster edge and v_{h+2} is a cluster center. However, by our construction, there is an edge from v_h to v_{h+2} with weight at

most $\text{dist}_{H_{2,<i}}(v_h, v_{h+2}) \leq \text{dist}_{H_{2,<i}}(v_h, v_{h+1}) + \text{dist}_{H_{2,<i}}(v_{h+1}, v_{h+2}) = \mathbf{w}(v_h, v_{h+1}) + \mathbf{w}(v_{h+1}, v_{h+2})$. Replacing $v_h v_{h+1}$ and $v_{h+1} v_{h+2}$ by $v_h v_{h+2}$ in P'_C , we obtain another shortest path from s to t with fewer intra-cluster edges, a contradiction.

Furthermore, the intra-cluster edges can only be $v_1 v_2$ and/or $v_{l-1} v_l$. Consider the sub-path $Q = \langle v_2, v_3, \dots, v_{l-1} \rangle$. We have $\mathbf{w}(v_h, v_{h+1}) + \mathbf{w}(v_{h+1}, v_{h+2}) > 2^i$ for any $1 \leq h \leq l-3$, since otherwise there is an edge from v_h to v_{h+2} by our construction. Then, we have:

$$\|Q\| \geq \sum_{h=1}^{\lfloor l/2 \rfloor - 1} (\mathbf{w}(v_{2h}, v_{2h+1}) + \mathbf{w}(v_{2h+1}, v_{2h+2})) \geq (\lfloor l/2 \rfloor - 1) \cdot 2^i.$$

On the other hand, since $\|P'_C\| \leq (1+\epsilon)(1+\kappa^2\delta)\|st\| \leq 8 \cdot 2^i$, we get that $(\lfloor l/2 \rfloor - 1) \cdot 2^i \leq \|Q\| \leq 8 \cdot 2^i$. Thus, $l \leq 19$. $\square$

For each vertex u , let $\text{Clusters}(u)$ be the set of clusters containing u . From Claim 5.12, for any edge $st \in E_i^{(i)}$, among all shortest paths from s to t in F'_i , there exists a path P'_C such that all edges that are incident to neither s nor t in P'_C are inter-cluster. For each pair (s, t) , we only need to check the shortest paths from the centers of all clusters in $\text{Clusters}(s)$ to the center of all clusters in $\text{Clusters}(t)$ containing a constant number of inter-cluster edges. By the packing bound, the total number of inter-cluster edges incident to a vertex is bounded by $\epsilon^{-O(d)}$. Thus, the total time complexity of checking each edge st is also $\epsilon^{-O(d)}$.

To find a helper edge, we find two net points: one in $N(A_{s,t})$ and one in $N(B_{s,t})$. (Recall that $N(A_{s,t})$ contains all net points $w \in N_h$ such that $B(w, \epsilon r_h) \cap A_{s,t} \neq \emptyset$ with h being the approximate level of st .) Then, we add the edge between those net-points to E_2 . Since the number of net points in $N(A_{s,t})$ (resp., $N(B_{s,t})$) is $\epsilon^{-O(d)}$, one can find a helper edge in $\epsilon^{-O(d)}$ time per edge.

Therefore, the total time complexity of a single run of the second pruning phase is $\epsilon^{-O(d)} n \log^2 n$. This bound absorbs the time complexity of a single run of the first pruning phases. After summation over all $\log^*(d/\epsilon)$ iterations, the construction time of our spanner is $\log^*(d/\epsilon) \cdot \epsilon^{-O(d)} n \log^2 n = \epsilon^{-O(d)} n \log^2 n$.

6 Lower Bounds for the Greedy Spanner

In this section, we construct point sets in the plane for which the lightness and sparsity of the greedy spanner far exceed the instance-optimal lightness and sparsity, respectively. In Section 6.1, we first construct hard examples against the greedy $(1+\epsilon)$ -spanner and $(1+1.2\epsilon)$ -spanner (Theorem 6.1), and then generalize the ideas to work against the greedy $(1+x\epsilon)$ -spanner for all x , $1 \leq x \leq o(\epsilon^{-1/3})$, by refining both the design of the point set and the of the greedy algorithm (Theorem 1.1). Since both the greedy and the sparsest spanner use edges of comparable weight, this construction already establishes the same lower bound for lightness (Corollary 6.2). In Section 6.2, we lower bounds for lightness with a stronger dependence on ϵ , and a weaker dependence on x : The points sets in these constructions are uniformly distributed along a suitable circular arc (Theorems 1.2 and 6.2).

6.1 Sparsity Lower Bounds

We begin with lower bound construction in Euclidean plane against the greedy $(1+\epsilon)$ -spanner. The same construction also works well against the greedy $(1+x\epsilon)$ -spanner when $1 \leq x \leq 1.2$.

Theorem 6.1. *For every sufficiently small $\epsilon > 0$, there exists a finite set $S \subset \mathbb{R}^2$ such that*

$$|E_{\text{gr}}| = |E_{\text{gr}(1.2)}| \geq \Omega(\epsilon^{-1/2}) \cdot |E_{\text{spa}}|,$$

where E_{gr} is the edge set of the greedy $(1+\epsilon)$ -spanner, $E_{\text{gr}(1.2\epsilon)}$ is the edge set of the greedy $(1+1.2\epsilon)$ -spanner, and E_{spa} is the edge set of a sparsest $(1+\epsilon)$ -spanner for S .

Proof. Let $\epsilon > 0$ be given. We construct a point set S as follows; refer to Fig. 9. All points are in an axis-aligned rectangle R of width 1 and height $\tan \alpha$, where α is determined by the equation

$$\frac{1}{\cos \alpha} = 1 + \epsilon. \quad (13)$$

This means, in particular, that the diagonal of R is exactly $1 + \epsilon$. Using the Taylor estimate $\frac{1}{\cos x} = 1 + \frac{x^2}{2} + O(x^4)$, we obtain $\alpha = \sqrt{2\epsilon} + O(\epsilon)$. Using the Taylor estimate $\tan x = x + O(x^3)$, this implies that the height of R is $\tan \alpha = \Theta(\alpha) = \Theta(\sqrt{\epsilon})$.

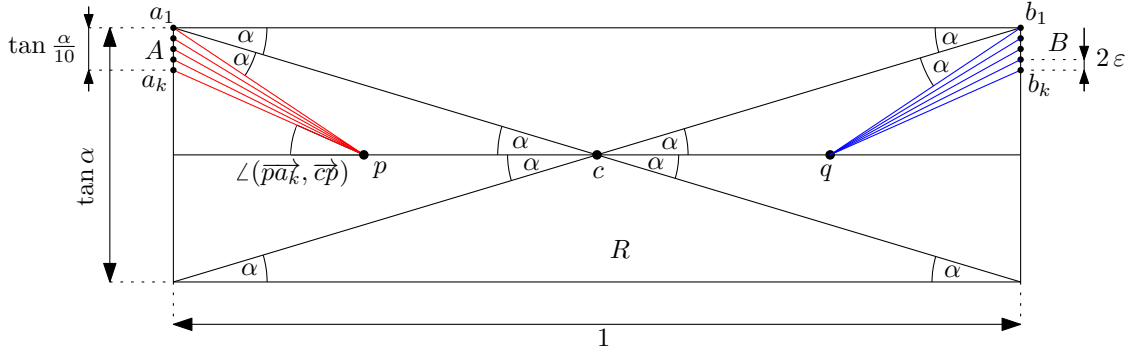


Figure 9: Lower bound construction for the sparsity of the greedy algorithm. Greedy adds all red and blue edges, but it does not add any edges between c and the points in the left and right sides of R . However, it adds all edges between A and B .

Now we can describe the point set S . Along the left and right edges of R , resp., place points $A = \{a_1, \dots, a_k\}$ and $B = \{b_1, \dots, b_k\}$ such that a_1 and b_1 are the upper-left and upper-right corners of R , the distance between any two points is at least 2ϵ , and $\text{diam}(A) = \text{diam}(B) = \tan(\alpha/10)$. The set S comprises $A \cup B$ and three additional points: the center c of R , and points p and q in the interior of R such that p, c, q lie on a horizontal line and $\angle ca_1p = \alpha$ and $\angle cb_1q = \alpha$. This completes the description of S . Note that $|S| = 2k + 3$. We have $k = \Theta(\frac{\tan(\alpha/10)}{\epsilon}) = \Theta(\alpha/\epsilon) = \Theta(\epsilon^{-1/2})$, consequently $|S| = \Theta(\epsilon^{-1/2})$.

Optimal sparsity. We construct a $(1+\epsilon)$ -spanner H for S with $\Theta(|S|)$ edges (i.e., sparsity $O(1)$). Let H contain the vertical paths $\langle a_1, a_2, \dots, a_k \rangle$ and $\langle b_1, b_2, \dots, b_k \rangle$, the horizontal path $pc \circ cq$, and all edges between $\{p, c, q\}$ and $A \cup B$. Clearly, H has $\Theta(|S|)$ edges. To show that H is a $(1+\epsilon)$ -spanner, consider a pair of points that are not adjacent in H . If both points are in A (or both are in B), then they are on a vertical path in H . The point pair $\{p, q\}$ is connected by the horizontal path $pc \circ cq$. It remains to consider the pairs $\{a_i, b_j\}$ for $a_i \in A$ and $b_j \in B$. We show that the path $a_i c \circ c b_j$ has weight at most $(1+\epsilon)\|a_i b_j\|$. On one hand, A and B lie on two parallel lines at distance 1 apart, hence $\|a_i b_j\| \geq 1$. On the other hand, $\|a_i c\| \leq \|a_1 c\| = \frac{1}{2 \cos \alpha} = \frac{1+\epsilon}{2}$ and similarly, $\|c b_j\| \leq \|c b_1\| = \frac{1}{2 \cos \alpha} = \frac{1+\epsilon}{2}$. Consequently, $\|a_i c\| + \|c b_j\| \leq \|a_1 c\| + \|c b_1\| \leq 1 + \epsilon \leq (1+\epsilon)\|a_i b_j\|$, as required.

Greedy sparsity. Now let us consider the greedy algorithm on the point set S . The same analysis works for the greedy $(1+\epsilon)$ -spanner and the greedy $(1+1.2\epsilon)$ -spanner (for short, the

greedy spanner). First observe that the greedy spanner contains all edges of the vertical paths $\langle a_1, a_2, \dots, a_k \rangle$ and $\langle b_1, b_2, \dots, b_k \rangle$, and the horizontal path $pc \circ cq$. It also contains all edges between p and A , and all edges between q and B (since the distance between any two points in A and any two points in B is at least 2ϵ). However, the greedy algorithm does not add any of the edges between c and $A \cup B$, because for every $a_i \in A$, the path $a_i p \circ pc$ has weight less than $(1 + \epsilon)\|a_i c\|$. Indeed, both segments $a_i p$ and cp make an angle at most α with $a_i c$. Combined with (13), this already implies $\|a_i p\| + \|pc\| \leq (1 + \epsilon)\|a_i c\|$. Similarly for every $b_j \in B$, the path $b_j q \circ qc$ has weight at most $(1 + \epsilon)\|b_j c\|$.

Finally, we show that the greedy algorithm will add all $|A| \cdot |B| = \Omega(\epsilon^{-1})$ edges between A and B . On the one hand, for any $a_i \in A$ and any $b_j \in B$, we have

$$\|a_i b_j\| \leq \frac{1}{\cos(\alpha/10)} = 1 + \frac{\alpha^2}{200} + O(\alpha^4) = 1 + \frac{\epsilon}{100} + O(\epsilon^2). \quad (14)$$

On the other hand, when the greedy algorithm considers adding edge $a_i b_j$, the shortest $a_i b_j$ -path in the current greedy spanner is $a_i p \circ pc \circ cq \circ q b_j$. For $a_i = a_1$ and $b_j = b_1$, the length of this path is $(1 + \epsilon)^2 = 1 + 2\epsilon + \epsilon^2 > (1 + \epsilon)\|a_1 b_1\|$. In general, for $i, j \in \{1, \dots, k\}$, the length of this $a_i b_j$ -path is minimized for $a_i = a_k$ and $b_j = b_k$, that is,

$$\|a_i p\| + \|pq\| + \|qb_j\| \geq \|a_k p\| + \|pq\| + \|qb_k\|. \quad (15)$$

By construction, $\Delta a_1 b_1 c$, $\Delta a_1 c p$, and $\Delta b_1 c q$ are isosceles triangles, and so $\|a_1 p\| = \|pc\| = \|cq\| = \|b_1 q\|$. Due to the choice of α , we have $\|a_1 c\| + \|b_1 c\| = 1 + \epsilon$ and $\|a_1 p\| + \|pc\| + \|cq\| + \|b_1 q\| = (1 + \epsilon)^2 = 1 + 2\epsilon + \epsilon^2$. This implies $\|a_1 p\| = \|pc\| = \|cq\| = \|b_1 q\| = \frac{1}{4} + \frac{\epsilon}{2} + O(\epsilon^2)$.

Note that $a_1 p$ and $a_k p$ have the same orthogonal projection to the x -axis, and so

$$\|a_1 p\| \cos(2\alpha) = \|a_k p\| \cos \angle(\overrightarrow{pa_k}, \overrightarrow{cp}). \quad (16)$$

Since $\|a_1 a_k\| \leq \tan \frac{\alpha}{10} < \frac{1}{8} \cdot \tan \alpha$ for all sufficiently small $\epsilon > 0$, then $\angle(\overrightarrow{pa_k}, \overrightarrow{cp}) < \frac{3\alpha}{2}$. This, combined with (16), the Taylor estimate $\cos x = 1 - \frac{x^2}{2} + O(x^4)$, and $\alpha = \sqrt{2\epsilon} + O(\epsilon)$, yields

$$\begin{aligned} \|a_k p\| &> \|a_1 p\| \frac{\cos(2\alpha)}{\cos(3\alpha/2)} \\ &= \|a_1 p\| (1 - 2\alpha^2 + O(\alpha^4)) \left(1 + \frac{9\alpha^2}{8} + O(\alpha^4)\right) \\ &= \left(\frac{1}{4} + \frac{\epsilon}{2} + O(\epsilon^2)\right) (1 - 4\epsilon + O(\epsilon^2)) \left(1 + \frac{9\epsilon}{4} + O(\epsilon^2)\right) \\ &= \frac{1}{4} + \frac{\epsilon}{16} + O(\epsilon^2). \end{aligned}$$

Overall, using (14) and (15), we obtain

$$\begin{aligned} \|a_i p\| + \|pq\| + \|qb_j\| &\geq \|a_k p\| + \|pq\| + \|qb_k\| \\ &= 2\|a_k p\| + 2\|pc\| \\ &> 2 \left(\frac{1}{4} + \frac{\epsilon}{16} + O(\epsilon^2)\right) + 2 \left(\frac{1}{4} + \frac{\epsilon}{2} + O(\epsilon^2)\right) \\ &= 1 + \frac{5\epsilon}{4} + O(\epsilon^2) \\ &> (1 + 1.2\epsilon) (1 + \epsilon/100 + O(\epsilon^2)) \\ &\geq (1 + 1.2\epsilon) \|a_i b_j\| \end{aligned}$$

for sufficiently small $\epsilon > 0$. This shows that the greedy algorithm must add edge $a_i b_j$ for all $a_i \in A$ and $b_j \in B$. Consequently, the greedy $(1 + \epsilon)$ -spanner and the greedy $(1 + 1.2\epsilon)$ -spanner both have at least $|A'| \cdot |B'| = \Omega(\epsilon^{-1})$ edges. $\square$

The construction above implies a lower bound for the lightness ratio, as well. We obtain a better bound in Section 6.2.

Corollary 6.1. *For every sufficiently small $\epsilon > 0$, there exists a finite set $S \subset \mathbb{R}^2$ such that*

$$\|E_{\text{gr}}\| = \|E_{\text{gr}(1.2)}\| \geq \Omega(\epsilon^{-1/2}) \cdot \|E_{\text{light}}\|,$$

where E_{gr} is the edge set of the greedy $(1 + \epsilon)$ -spanner, $E_{\text{gr}(1.2)}$ is the edge set of the greedy $(1 + 1.2\epsilon)$ -spanner, and E_{light} is the edge set of a minimum-weight spanner for S .

Proof. For $\epsilon > 0$, consider the point set S constructed in the proof of Theorem 6.1. We have shown that S admits a $(1 + \epsilon)$ -spanner H with $O(\epsilon^{-1/2})$ edges, each of weight $O(1)$. Consequently, $\|E_{\text{light}}\| \leq \|E(H)\| = O(\epsilon^{-1/2})$.

We have also shown that the greedy $(1 + \epsilon)$ - and $(1 + 1.2\epsilon)$ -spanner each contain $\Omega(|S|^2) = \Omega(\epsilon^{-1})$ edges, each of weight $\Omega(1)$. Consequently, their total weight is $\Omega(\epsilon^{-1})$, which is $\Omega(\epsilon^{-1/2})\|E_{\text{light}}\|$. $\square$

The lower bound construction generalizes to the case when the stretch is $(1 + x\epsilon)$ for some $1 \leq x \leq o(\epsilon^{-1/2})$, and we compare the sparsest $(1 + \epsilon)$ -spanner with the greedy $(1 + x\epsilon)$ -spanner.

Theorem 1.1 (Sparsity lower bound for greedy). *For every sufficiently small $\epsilon > 0$ and $1 \leq x \leq o(\epsilon^{-1/3})$, there exists a finite set $S \subset \mathbb{R}^2$ such that*

$$|E_{\text{gr}(x)}| \geq \Omega\left(\frac{\epsilon^{-1/2}}{x^{3/2}}\right) \cdot |E_{\text{spa}}|,$$

where $E_{\text{gr}(x)}$ is the edge set of the greedy $(1 + x\epsilon)$ -spanner, and E_{spa} is the edge set of a sparsest $(1 + \epsilon)$ -spanner for S .

Proof. Let $\epsilon > 0$ and $1 \leq x \leq o(\epsilon^{-1/3})$ be given. We construct a point set S as follows; refer to Fig. 10. Let R be an axis-aligned rectangle R of width 1 and height $\tan \alpha$, where α is determined by the equation

$$\frac{1}{\cos \alpha} = 1 + \epsilon. \tag{17}$$

This means, in particular, that the diagonals of R have length precisely $1 + \epsilon$. Using the Taylor estimate $\frac{1}{\cos x} = 1 + \frac{x^2}{2} + O(x^4)$, we obtain $\alpha = \sqrt{2\epsilon} + O(\epsilon)$. Using the Taylor estimate $\tan x = x + O(x^3)$, this implies that the height of R is $\tan \alpha = \sqrt{2\epsilon} + O(\epsilon) = \Theta(\sqrt{\epsilon})$.

Denote by a_1 and b_1 the upper-left and upper-right corners of R , respectively, and let c be the center of R . Next, place points p and q below the segments $a_1 c$ and $b_1 c$, resp., such that

$$\|a_1 p\| = \|p c\| = \|c q\| = \|q b_1\| = \frac{1}{2}(1 + x\epsilon)\|a_1 c\| = \frac{1}{4}(1 + x\epsilon)(1 + \epsilon) = \frac{1}{4}\left(1 + (x + 1)\epsilon + O(\epsilon^2)\right).$$

Note that $\triangle a_1 c p$ and $\triangle b_1 c q$ are isosceles triangles, where $\beta := \angle c a_1 p = \angle p c a_1 = \angle b_1 c q = \angle q b_1 c$ and $\beta = \sqrt{2x\epsilon} + O(x\epsilon)$. Let R_a be the rectangle with one side $a_1 c$ and the opposite side containing p ; and similarly let R_b be the rectangle with one side $b_1 c$ and the opposite side containing q .

Now we can describe the point set S . Along the left side of R_a and the right side of R_b , resp., place equally spaced points $A = \{a_1, \dots, a_k\}$ and $B = \{b_1, \dots, b_k\}$ such that the distance between any two points is at least $2x\epsilon$, and $\text{diam}(A) = \text{diam}(B) = \frac{1}{20\sqrt{x}} \tan \alpha$. Our point set is $S = A \cup B \cup \{c, p, q\}$. Note that $|S| = 2k + 3$. We have $k = \Theta\left(\frac{\tan \alpha}{\sqrt{x \cdot x\epsilon}}\right) = \Theta\left(\frac{\alpha}{x^{3/2}\epsilon}\right) = \Theta(\epsilon^{-1/2}/x^{3/2})$, consequently $|S| = \Theta(\epsilon^{-1/2}/x^{3/2})$.

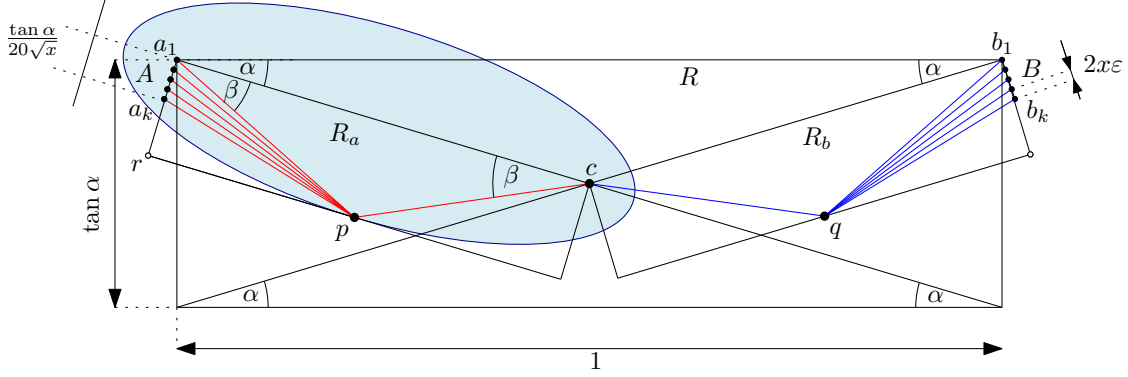


Figure 10: Lower bound construction for the sparsity of the $(1 + x\epsilon)$ -greedy spanner. The locus of all points p such that $\|a_1p\| + \|pc\| = (1 + x\epsilon)\|a_1c\|$ is an ellipse with foci a_1 and c . The greedy $(1 + x\epsilon)$ -spanner contains all red and blue edges, as well as all edges between A and B .

Optimal sparsity. We construct a $(1 + \epsilon)$ -spanner H for S with $\Theta(|S|)$ edges (i.e., sparsity $O(1)$). Let H contain the paths $\langle a_1, a_2, \dots, a_k \rangle$ and $\langle b_1, b_2, \dots, b_k \rangle$, the edges of the triangle Δcpq , and all edges between $\{p, c, q\}$ and $A \cup B$. Clearly, H has $\Theta(|S|)$ edges.

To show that H is a $(1 + \epsilon)$ -spanner, consider a pair of points that are not adjacent in H . Any pair of points in A (resp., B), are connected by a path of collinear edges. Consider a point pair $\{a_i, b_j\}$, where $a_i \in A$ and $b_j \in B$. We show that the path $a_i c \circ c b_j$ has weight at most $(1 + \epsilon)\|a_i b_j\|$. This clearly holds for $i = j = 1$, where $\|a_1 c\| + \|c b_1\| = (1 + \epsilon) = (1 + \epsilon)\|a_1 b_1\|$ by the choice of α . In particular, $c \in \mathcal{E}(a_1, b_1)$, where $\mathcal{E}(f_1, f_2)$ denotes the ellipse with foci f_1 and f_2 and great axis $(1 + \epsilon)\|f_1 f_2\|$. For all $i \in \{1, \dots, k\}$, the segment $a_i b_i$ is horizontal and $\|a_i b_i\| \geq \|a_1 b_1\|$, therefore $\mathcal{E}(a_i, b_i)$ is obtained from $\mathcal{E}(a_1, b_1)$ by a vertical translation and scaling; hence $c \in \mathcal{E}(a_i, b_i)$. Finally, for all $j \geq i$, we have $\|a_i b_j\| \geq \|a_i b_i\|$, and $\mathcal{E}(a_i, b_j)$ is obtained from $\mathcal{E}(a_i, b_i)$ by a rotation about a_i and scaling; and so $c \in \mathcal{E}(a_i, b_j)$. Overall, we have $c \in \mathcal{E}(a_i, b_j)$ for all $i, j \in \{1, \dots, k\}$, which implies $\|a_i c\| + \|b_j c\| \leq 1 + \epsilon \leq (1 + \epsilon)\|a_i b_j\|$, as required.

Greedy sparsity. Now let us consider the greedy $(1 + x\epsilon)$ -spanner $G_{\text{gr}(x)} = (S, E_{\text{gr}(x)})$ on the point set S . The greedy algorithm sorts the point pairs in S by weight: It adds all edges of the paths $(a_1, \dots, a_k)$ and $(b_1, \dots, b_k)$ to $G_{\text{gr}(x)}$. It then considers the edges between p and A (resp., q and B) by increasing weight: We claim that it adds all these edges to $G_{\text{gr}(x)}$. By symmetry, it is enough to show that if the greedy algorithm has already added $a_i p$ for some $1 < i \leq k/2$, then it also adds $a_{i-1} p$. That is, $\|a_{i-1} a_i\| + \|a_i p\| \geq (1 + x\epsilon)\|a_{i-1} p\|$. Since $\|a_{i-1} a_i\| \geq 2x\epsilon$ by construction, then a_{i-1} lies outside of the ellipse with foci a_i and p , and great axis $(1 + x\epsilon)\|a_i p\|$. Thus the greedy spanner contains all edges between A and p , and between B and q .

Next, we show that the greedy algorithm does not add any edge between A and $\{c, q\}$. We claim that for any point $a_i \in A$, the paths $a_i p \circ pc$ and $a_i p \circ pc \circ cq$ have stretch at most $1 + x\epsilon$. This clearly holds for the path $a_1 p \circ pc$ by the definition of p . An easy calculation shows that it holds for all other point pairs in $A \times \{c, q\}$. Symmetrically, the greedy $(1 + x\epsilon)$ -spanner has no edges between B and $\{p, c\}$.

Finally, the greedy algorithm considers pairs $\{a_i, b_j\}$ for $a_i \in A$ and $b_j \in B$ sorted by weight. We show that for all such pairs, we have $\|a_i p \circ pc \circ cq \circ q b_j\| \geq (1 + x\epsilon)\|a_i b_j\|$, and the greedy algorithm must add the edge $a_i b_j$ to $G_{\text{gr}(x)}$. This is clear for the pair $\{a_1, b_1\}$, where the definition

of p and q gives

$$\|a_1p \circ pc \circ cq \circ qb_1\| = (1 + x\epsilon)(1 + \epsilon) = 1 + (x + 1)\epsilon + O(x\epsilon^2) > (1 + x\epsilon)\|a_1b_1\|. \quad (18)$$

In general, consider an arbitrary point pair $(a_i, b_j) \in A \times B$. The distance between a_i and b_j is maximized for $a_i = a_k$ and $b_j = b_k$, where

$$\|a_i b_j\| \leq \|a_k b_k\| = 1 + 2 \cdot \frac{\tan \alpha}{20\sqrt{x}} \sin \alpha \leq 1 + \frac{\sqrt{2\epsilon}}{10} \cdot \sqrt{2\epsilon} + O(\epsilon^{3/2}) = 1 + \frac{\epsilon}{5} + O(\epsilon^{3/2}).$$

We give a lower bound for $\|a_i p \circ pc \circ cq \circ qb_j\|$ using Equation (18) and the differences $\|a_1 p\| - \|a_i p\|$ and $\|b_1 q\| - \|b_j q\|$. By symmetry and monotonicity, it is enough to give an upper bound for $\|a_1 p\| - \|a_k p\|$. Let r denote the bottom-right vertex of R_a ; see Figure 10. The Pythagorean theorem for the right triangles $\Delta a_1 p r$ and $\Delta a_k p r$, combined with the identity $\sqrt{y} - \sqrt{z} = \frac{y-z}{\sqrt{y}+\sqrt{z}}$, yields

$$\begin{aligned} \|a_1 p\| - \|a_k p\| &= \sqrt{\|a_1 r\|^2 + \|pr\|^2} - \sqrt{\|a_k r\|^2 + \|pr\|^2} \\ &= \frac{\|a_1 r\|^2 - \|a_k r\|^2}{\sqrt{\|a_1 r\|^2 + \|pr\|^2} + \sqrt{\|a_k r\|^2 + \|pr\|^2}} \\ &< \frac{(\|a_1 a_k\| + \|a_k r\|)^2 - \|a_k r\|^2}{\|a_1 p\|} \\ &= \frac{\|a_1 a_k\|^2 + 2 \cdot \|a_1 a_k\| \cdot \|a_k r\|}{\|a_1 p\|} \\ &< \frac{2 \cdot \|a_1 a_k\| \cdot \|a_1 r\|}{\|a_1 p\|} \\ &= \frac{2 \cdot \frac{\tan \alpha}{20\sqrt{x}} \cdot \frac{1}{4}(1 + \epsilon) \tan \beta}{\frac{1}{4}(1 + \epsilon)(1 + x\epsilon)} \\ &= \frac{\tan \alpha \cdot \tan \beta}{10\sqrt{x}(1 + x\epsilon)} \\ &= \frac{1}{10\sqrt{x}} \left(\sqrt{2\epsilon} + O(\epsilon) \right) \left(\sqrt{2x\epsilon} + O(x\epsilon) \right) \left(1 - x\epsilon + O(x^2\epsilon^2) \right) \\ &\leq \frac{\epsilon}{5} + O(\epsilon^{3/2}). \end{aligned}$$

Overall, for all $a_i \in A$ and $b_j \in B$, we have

$$\begin{aligned} \|a_i p \circ pc \circ cq \circ qb_j\| &\geq \|a_1 p \circ pc \circ cq \circ qb_1\| - \left(\|a_1 p\| - \|a_i p\| \right) - \left(\|b_1 q\| - \|b_j q\| \right) \\ &\geq \|a_1 p \circ pc \circ cq \circ qb_1\| - 2 \cdot \left(\|a_1 p\| - \|a_k p\| \right) \\ &\geq \left(1 + (x + 1)\epsilon + O(x\epsilon^2) \right) - 2 \cdot \left(\frac{\epsilon}{5} + O(\epsilon^{3/2}) \right) \\ &\geq 1 + \left(x + \frac{3}{5} \right) \epsilon + O(\epsilon^{3/2}) \\ &> (1 + x\epsilon) \left(1 + \frac{\epsilon}{5} + O(\epsilon^{3/2}) \right) \\ &\geq (1 + x\epsilon) \|a_i b_j\| \end{aligned}$$

for all sufficiently small $\epsilon > 0$ and $x = o(\epsilon^{-1/3})$. This shows that the greedy $(1+x\epsilon)$ -spanner contains edges $a_i b_j$ for all $a_i \in A$ and $b_j \in B$. In particular, we have $|E_{\text{spa}}| = O(|S|)$ and $|E_{\text{gr}(x)}| = \Omega(|S|^2)$. Consequently, $|E_{\text{gr}(x)}| \geq \Omega(|S|) \cdot |E_{\text{spa}}| \geq \Omega(\epsilon^{-1/2}/x^{3/2}) \cdot |E_{\text{spa}}|$.

The construction above has $|S| = O(\epsilon^{-1/2}/x^{3/2})$ points. However, we can obtain arbitrarily large point sets with the same property as a disjoint union of translated copies of the construction above: Both G_{spa} and $G_{\text{gr}(x)}$ would have only one extra edge per copy, so the bound $|E_{\text{gr}(x)}| \geq \Omega(\epsilon^{-1/2}/x^{3/2}) \cdot |E_{\text{spa}}|$ carries over. $\square$

Corollary 6.2. *For every sufficiently small $\epsilon > 0$ and $1 \leq x \leq o(\epsilon^{-1/3})$, there exists a finite set $S \subset \mathbb{R}^2$ such that*

$$\|E_{\text{gr}(x)}\| \geq \Omega\left(\frac{\epsilon^{-1/2}}{x^{3/2}}\right) \cdot \|E_{\text{light}}\|,$$

where $E_{\text{gr}(x)}$ is the edge set of the greedy $(1+x\epsilon)$ -spanner, and E_{light} is the edge set of a minimum-weight $(1+\epsilon)$ -spanner for S .

Proof. For $\epsilon > 0$, consider the point set S constructed in the proof of Theorem 1.1. We have shown that S admits a $(1+\epsilon)$ -spanner H with $O(|S|) = O(\epsilon^{-1/2}/x^{3/2})$ edges, each of weight $O(1)$.

We have also shown that the greedy $(1+x\epsilon)$ -spanner has $\Omega(|S|^2)$ edges, each of weight $\Omega(1)$. Consequently, $\|E_{\text{gr}(x)}\|/\|E_{\text{light}}\| \geq \Omega(|S|) \geq \Omega(\epsilon^{-1/2}/x^{3/2})$. $\square$

6.2 Lightness Lower Bounds

Theorem 6.2. *For every sufficiently small $\epsilon > 0$, there exists a finite set $S \subset \mathbb{R}^2$ such that*

$$\|E_{\text{gr}}\| \geq \|E'_{\text{gr}}\| \geq \Omega(\epsilon^{-1}) \cdot \|E_{\text{light}}\|,$$

where E_{gr} is the edge set of a greedy $(1+\epsilon)$ -spanner, E'_{gr} is the edge set of the greedy $(1+1.01\epsilon)$ -spanner, and E_{light} is the edge set of a minimum-weight $(1+\epsilon)$ -spanner for S .

Proof. Let $\epsilon > 0$ be given. We construct a point set S as follows; refer to Fig. 11. The points in S lie on a circular arc C , which is an arc of length $\alpha + \beta$ of a circle of radius 1 centered at the origin o , for angles $\alpha < \beta$ to be specified later. For any $s, t \in C$, let $C(s, t)$ denote the subarc of C between s and t , and let $\text{arc}(s, t) = \|C(s, t)\|$ denote the length of the circular arc $C(s, t)$. First, we place four points $p_1, \dots, p_4 \in C$ such that $\text{arc}(p_1, p_2) = \alpha$, $\text{arc}(p_2, p_3) = \beta - \alpha$, and $\text{arc}(p_3, p_4) = \alpha$. Now we construct the point set S as follows: Place a large number of equally spaced points along $C(p_1, p_2)$ and $C(p_2, p_3)$, and then populate $C(p_3, p_4)$ with the rotated copy of the points in $C(p_1, p_2)$. Note that, by construction, S contains a large number of point pairs $\{s, t\} \subset S$ with $\|st\| = \|p_1 p_3\|$.

Let P be the spanning path of S that connects pairs of consecutive points along the circle. For $s, t \in S$, let $P(s, t)$ denote the subpath of P between s and t . If the points in S are sufficiently dense along C , the length of $P(s, t)$ will be arbitrarily close to $\text{arc}(s, t)$. In the remainder of this proof, we assume that $\|P(s, t)\| = \text{arc}(s, t) + O(\epsilon^2)$ for all $s, t \in S$.

Using the Taylor estimates for sine and the bound $\text{arc}(s, t) \leq 2\beta = O(\epsilon^{1/2})$, the difference between $\text{arc}(s, t)$ and the length of the line segment st is

$$\|P(s, t)\| - \|st\| = \text{arc}(s, t) - 2 \sin\left(\frac{1}{2} \text{arc}(s, t)\right) + O(\epsilon^2) = \frac{(\text{arc}(s, t))^3}{48} + O(\epsilon^2). \quad (19)$$

To complete the construction, we specify α and β . We set $\alpha = \frac{\beta}{10}$ and choose β such that

$$\|P(p_1, p_3)\| > (1+\epsilon)\|p_1 p_3\|, \quad (20)$$

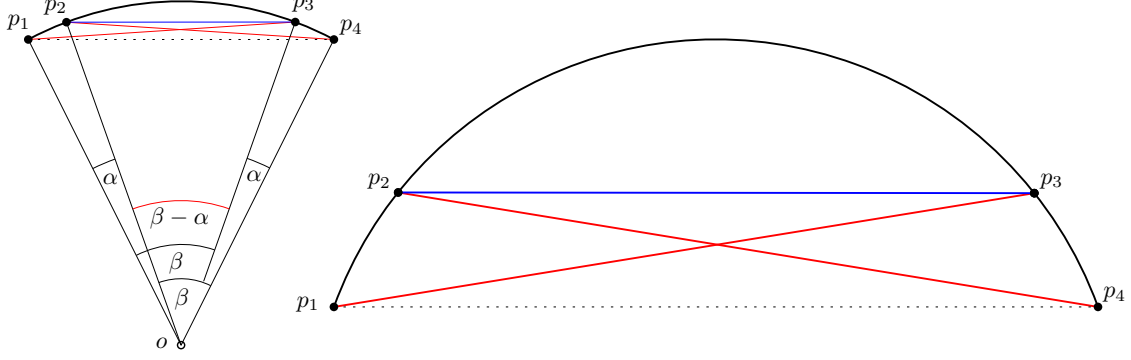


Figure 11: Lower bound construction for the weight of the greedy algorithm.

but $\|P(s, t)\| \leq (1 + \epsilon)\|st\|$, for all $s, t \in C$ with $\|P(s, t)\| < \|P(p_1, p_3)\|$. Recall that $\|P(p_1, p_3)\| = \text{arc}(p_1, p_3) + O(\epsilon^2) = \beta + O(\epsilon^2)$. The line segment p_1p_3 has length $\|p_1p_3\| = 2 \sin \frac{\beta}{2}$. Using the Taylor estimate $x - \frac{x^3}{6} \leq \sin x \leq x - \frac{x^3}{6} + O(x^5)$, we have $\beta(1 - \frac{\beta^2}{48}) \leq \|p_1p_3\| \leq \beta(1 - \frac{\beta^2}{48} + O(\beta^4))$. Now $\|P(p_1, p_3)\| = (1 + \epsilon)\|p_1p_3\| + O(\epsilon^2)$ is attained for $\epsilon = \beta^2/48 + \Theta(\beta^4)$ or $\beta = \sqrt{48\epsilon} + \Theta(\epsilon)$.

Optimal weight. We show that the minimum weight of an $(1 + \epsilon)$ -spanner for S is $\|E_{\text{light}}\| = O(\epsilon^{1/2})$. We claim that the graph H comprised of the path P and the edge p_2p_3 is a $(1 + \epsilon)$ -spanner for S . Consequently, $\|E_{\text{light}}\| \leq \|E(H)\| = \|P\| + \|p_2p_3\| \leq (\alpha + \beta) + (\beta - \alpha) = 2\beta = O(\epsilon^{1/2})$.

To prove the claim, note first that for any pair $s, t \in S$ with $\|st\| < \|p_1p_3\|$, the path $P(s, t) \subset H$ has weight at most $(1 + \epsilon)\|st\|$ due to the choice of β (cf., Equation (20)). Consider now a pair $s, t \in S$ with $\|st\| \geq \|p_1p_3\|$. Then the points s and t lie in two distinct arcs $C(p_1, p_2)$ and $C(p_3, p_4)$. We may assume w.l.o.g. that $s \in C(p_1, p_2)$ and $t \in C(p_3, p_4)$. Then H contains the st -path $P(s, p_2) \circ p_2p_3 \circ P(p_3, t)$. The weight of this st -path is bounded by:

$$\begin{aligned}
\|P(s, p_2) \circ p_2p_3 \circ P(p_3, t)\| &\leq \text{arc}(s, p_2) + \|p_2p_3\| + \text{arc}(p_3, t) + O(\epsilon^2) \\
&\leq \left(\text{arc}(s, p_2) + \text{arc}(p_2, p_3) + \text{arc}(p_3, t) \right) + (\|p_2p_3\| - \text{arc}(p_2, p_3)) + O(\epsilon^2) \\
&= \text{arc}(s, t) + \left(\|p_2p_3\| - \text{arc}(p_2, p_3) \right) + O(\epsilon^2) \\
&= \|st\| + \left(\text{arc}(s, t) - \|st\| \right) + \left(\|p_2p_3\| - \text{arc}(p_2, p_3) \right) + O(\epsilon^2) \\
&= \|st\| + \frac{(\text{arc}(s, t))^3 - (\text{arc}(p_2, p_3))^3}{48} + O(\epsilon^2) \\
&\leq \|st\| + \frac{(\text{arc}(p_1, p_4))^3 - (\text{arc}(p_2, p_3))^3}{48} + O(\epsilon^2) \\
&= \|st\| + \frac{(\alpha + \beta)^2 - (\beta - \alpha)^3}{48} + O(\epsilon^2) \\
&= \|st\| + \frac{(1.1^3 - 0.9^3)\beta^3}{48} + O(\epsilon^2) \\
&< \|st\| + \frac{0.7 \cdot (\sqrt{48\epsilon} + \Theta(\epsilon))^3}{48} + O(\epsilon^2) \\
&= \|st\| + 0.7 \cdot \sqrt{48\epsilon} \cdot \epsilon + O(\epsilon^2) \\
&= \|st\| + 0.7 \cdot \beta \cdot \epsilon + O(\epsilon^2)
\end{aligned}$$

$$\begin{aligned}
&< \|st\| + 0.7 \cdot \left(\|p_1 p_2\| + \frac{\beta^3}{48} + O(\epsilon^2) \right) \cdot \epsilon + O(\epsilon^2) \\
&= \left(1 + 0.7 \cdot \epsilon \right) \|st\| + O(\epsilon^2) \\
&< (1 + \epsilon) \|st\|,
\end{aligned} \tag{21}$$

for a sufficiently small $\epsilon > 0$. This confirms that H is a $(1 + \epsilon)$ -spanner for S , as claimed.

Greedy weight. We show that the greedy $(1 + \epsilon)$ -spanner and the greedy $(1 + 1.01\epsilon)$ -spanner for S both have weight $\Omega(\epsilon^{-1/2})$. We argue about the greedy $(1 + \epsilon)$ -spanner (greedy spanner, for short), but essentially the same argument holds for the greedy $(1 + 1.01\epsilon)$ -spanner, as well. Let $G_{\text{gr}} = (S, E_{\text{gr}})$. The greedy algorithm adds the entire path $P = P(p_1, p_4)$ to G_{gr} , and then considers point pairs sorted by increasing weight. For all point pairs $\{s, t\} \subset S$ with $\|st\| < \|p_1 p_3\|$, we have $\|P(s, t)\| < (1 + \epsilon) \|st\|$, and so none of these edges is added to G_{gr} . By construction, S contains a large number of point pairs $\{s, t\}$ with $\|st\| = \|p_1 p_3\|$. We show that the greedy algorithm adds $\Omega(\alpha/\epsilon)$ such pairs to G_{gr} .

Specifically, we claim that every circular arc of length $2\epsilon\beta$ of $C(p_1, p_2)$ contains an endpoint of some edge of weight $\|p_1 p_3\|$ in G_{gr} . Suppose, for the sake of contradiction, that this is not the case. Then there is a point pair $\{s, t\}$ with $\|st\| = \|p_1 p_3\|$ such that G_{gr} does not contain any edge of this weight whose endpoints are within arc distance $\epsilon\beta$ from s or t . This means that when the greedy algorithm considers the point pair $\{s, t\}$, all st -paths have length more than $(1 + \epsilon) \|st\|$: The path $P(s, t)$ has weight more than $(1 + \epsilon) \|st\|$ due to (20). Furthermore, any st -path that goes through a previously added edge of weight $\|p_1 p_3\|$ must include subpaths of length at least $\epsilon\beta$ in the neighborhood of s and t , resp., and so its total length is at least $2\epsilon\beta + \|p_1 p_2\| > (1 + \epsilon) \|st\|$. Consequently, the greedy algorithm must add st to the spanner, which is a contradiction. This completes the proof of the claim.

Since $\text{arc}(p_1, p_2) = \alpha$, then G_{gr} contains $\Omega(\alpha/(\epsilon\beta)) = \Omega(\epsilon^{-1})$ edges of weight $\|p_1 p_3\| = \Theta(\beta) = \Theta(\epsilon^{1/2})$, and so the weight of the greedy spanner is $\|E_{\text{gr}}\| \geq \Omega(\epsilon^{-1} \cdot \epsilon^{1/2}) = \Omega(\epsilon^{-1/2})$. $\square$

The lower bound generalizes to the case where we relax the stretch to $a + x\epsilon$ for $2 \leq x \leq O(\epsilon^{-1/2})$, and compare the lightest $(1 + \epsilon)$ -spanner with the greedy $(1 + x\epsilon)$ -spanner.

Theorem 1.2 (Lightness lower bound for greedy). *For every sufficiently small $\epsilon > 0$ and $x \in [2, \epsilon^{-1/2}/48]$, there exists a finite set $S \subset \mathbb{R}^2$ such that*

$$\|E_{\text{gr}(x)}\| \geq \Omega\left(\frac{\epsilon^{-1}}{x^2 \cdot \log x}\right) \cdot \|E_{\text{light}}\|,$$

where $E_{\text{gr}(x)}$ is the edge set of the greedy $(1 + x\epsilon)$ -spanner, and E_{light} is the edge set of a minimum-weight $(1 + \epsilon)$ -spanner for S .

Proof. We use the point set in the proof of Theorem 6.2, with $x\epsilon$ in place of ϵ . That is, S is a set of points on a circular arc of radius 1 and angle $\alpha_x + \beta_x$, where $\alpha_x = \beta_x/10$ and $\beta_x = \sqrt{48x\epsilon} + O(x\epsilon)$. In the proof of Theorem 6.2, we saw that $\|G_{\text{gr}(x)}\| = \Omega((x\epsilon)^{-1/2})$.

Optimal weight. We show that S admits a $(1 + \epsilon)$ -spanner of weight $O(\epsilon^{1/2} \cdot x^2 \log x)$. Specifically, we construct a graph H on S that comprises the path P and a set of chords. For $i = -2, -1, 0, 1, \dots, 2 \lceil 10x \log \sqrt{x} \rceil$, we augment H with a maximal collection of chords, each of length $\sqrt{48\epsilon} \cdot \left(1 + \frac{1}{10x}\right)^{i/2}$, such that the arc distance between the left endpoints of any two chords is $\frac{1}{20} \cdot \sqrt{48\epsilon} \cdot \left(1 + \frac{1}{10x}\right)^i$.

We claim that H is a $(1 + \epsilon)$ -spanner for S . Consider a point pair $s, t \in S$. If $\text{arc}(s, t) \leq \sqrt{48\epsilon}$, then $\|P(s, t)\| \leq (1 + \epsilon)\|st\|$. Otherwise, there exists a chord $ab \in H$ such that $a, b \in C(s, t)$ and

$$\left(1 + \frac{1}{10x}\right)^{-1} \text{arc}(s, t) \leq \text{arc}(a, b) \leq \left(1 + \frac{1}{10x}\right)^{-1/2} \text{arc}(s, t).$$

Equation (21) now shows (substituting $\beta_x = \sqrt{48x\epsilon} + O(x\epsilon)$ instead of $\beta = \sqrt{48\epsilon} + O(\epsilon)$) that $\|P(sa)\| + \|ab\| + \|P(bt)\| \leq (1 + \epsilon)\|st\|$.

It remains to bound the weight of H . The weight of the path P is $\|P\| = O(\beta) = O(\sqrt{x\epsilon})$. For every $i \in \{-2, -1, \dots, 2 \lceil 10x \log \sqrt{x} \rceil\}$, the total weight of the chords of length $\sqrt{48\epsilon} \cdot \left(1 + \frac{1}{10x}\right)^i$ is $O(20 \cdot \beta_x) = O(\sqrt{x\epsilon})$. Consequently, $\|E(H)\| = O(\sqrt{x\epsilon} \cdot x \log x) = O(\epsilon^{1/2} \cdot x^{3/2} \log x)$. $\square$

Acknowledgements. The second-named author thanks Mike Dinitz for interesting discussions. We thank anonymous reviewers of FOCS 2024 for thorough comments, especially on approximating spanners of general graphs. Hung Le and Cuong Than are supported by the NSF CAREER award CCF-2237288, the NSF grant CCF-2121952, and a Google Research Scholar Award. Research by Csaba D. Tóth was supported by the NSF award DMS-2154347. Shay Solomon and Tianyi Zhang are funded by the European Union (ERC, DynOpt, 101043159). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Council. Neither the European Union nor the granting authority can be held responsible for them. Shay Solomon is also supported by the Israel Science Foundation (ISF) grant No.1991/1 and by a grant from the United States-Israel Binational Science Foundation (BSF), Jerusalem, Israel, and the United States National Science Foundation (NSF).

References

- [ABS⁺20] Reyhan Ahmed, Greg Bodwin, Faryad Darabi Sahneh, Keaton Hamm, Mohammad Javad Latifi Jebelli, Stephen Kobourov, and Richard Spence. Graph spanners: A tutorial review. *Computer Science Review*, 37:100253, 2020. doi:10.1016/j.cosrev.2020.100253.
- [ADD⁺93] Ingo Althöfer, Gautam Das, David Dobkin, Deborah Joseph, and José Soares. On sparse spanners of weighted graphs. *Discrete & Computational Geometry*, 9(1):81–100, 1993. doi:10.1007/BF02189308.
- [Aro98] Sanjeev Arora. Polynomial time approximation schemes for Euclidean traveling salesman and other geometric problems. *Journal of the ACM*, 45(5):753–782, 1998. doi:10.1145/290179.290180.
- [BBGW24] Kevin Buchin, Maike Buchin, Joachim Gudmundsson, and Sampson Wong. Bicriteria approximation for minimum dilation graph augmentation, 2024. To appear in ESA 2024. arXiv:2407.04614.
- [BBM⁺11] Piotr Berman, Arnab Bhattacharyya, Konstantin Makarychev, Sofya Raskhodnikova, and Grigory Yaroslavtsev. Improved approximation for the directed spanner problem. In *Proc. 38th International Colloquium on Automata, Languages, and Programming (ICALP)*, volume 6755 of *LNCS*, pages 1–12. Springer, 2011. doi:10.1007/978-3-642-22006-7_1.

- [BGJ⁺12] Arnab Bhattacharyya, Elena Grigorescu, Kyomin Jung, Sofya Raskhodnikova, and David P Woodruff. Transitive-closure spanners. *SIAM Journal on Computing*, 41(6):1380–1425, 2012. doi:10.1137/110826655.
- [BRR10] Piotr Berman, Sofya Raskhodnikova, and Ge Ruan. Finding sparser directed spanners. In *Proc. IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS)*, volume 8 of *LIPICs*, pages 424–435. Schloss Dagstuhl, 2010. doi:10.4230/LIPICs.FSTTCS.2010.424.
- [BT22] Sujoy Bhore and Csaba D Tóth. Euclidean Steiner spanners: Light and sparse. *SIAM Journal on Discrete Mathematics*, 36(3):2411–2444, 2022. doi:10.1137/22M1502707.
- [CCY13] Paz Carmi and Lilach Chaitman-Yerushalmi. Minimum weight Euclidean t -spanner is NP-hard. *J. Discrete Algorithms*, 22:30–42, 2013. doi:10.1016/j.jda.2013.06.010.
- [CDNS95] Barun Chandra, Gautam Das, Giri Narasimhan, and José Soares. New sparseness results on graph spanners. *Int. J. Comput. Geom. Appl.*, 5:125–144, 1995. doi:10.1142/S0218195995000088.
- [CGMZ16] T.-H. Hubert Chan, Anupam Gupta, Bruce M. Maggs, and Shuheng Zhou. On hierarchical routing in doubling metrics. *ACM Trans. Algorithms*, 12(4):55:1–55:22, 2016. Preliminary version appeared in SODA 2005. doi:10.1145/2915183.
- [Che86] L. Paul Chew. There is a planar graph almost as good as the complete graph. In *Proc. 2nd ACM Symposium on Computational Geometry (SoCG)*, pages 169–177, 1986. doi:10.1145/10515.10534.
- [Cla87] Kenneth Clarkson. Approximation algorithms for shortest path motion planning. In *Proc. 19th ACM Symposium on Theory of Computing (STOC)*, pages 56–65, 1987. doi:10.1145/28395.28402.
- [CS22] Markus Chimani and Finn Stutzenstein. Spanner approximations in practice. In *Proc. 30th European Symposium on Algorithms (ESA)*, volume 244 of *LIPICs*, pages 37:1–37:15. Schloss Dagstuhl, 2022. doi:10.4230/LIPICs.ESA.2022.37.
- [DK11] Michael Dinitz and Robert Krauthgamer. Directed spanners via flow-based linear programs. In *Proceedings of the 43rd ACM Symposium on Theory of Computing (STOC)*, pages 323–332, 2011. doi:10.1145/1993636.1993680.
- [DKR15] Michael Dinitz, Guy Kortsarz, and Ran Raz. Label cover instances with large girth and the hardness of approximating basic k -spanner. *ACM Transactions on Algorithms (TALG)*, 12(2):1–16, 2015. doi:10.1145/2818375.
- [DN97] Gautam Das and Giri Narasimhan. A fast algorithm for constructing sparse Euclidean spanners. *International Journal of Computational Geometry & Applications*, 7(04):297–315, 1997. doi:10.1142/S0218195997000193.
- [DZ16] Michael Dinitz and Zeyu Zhang. Approximating low-stretch spanners. In *Proceedings of the 27th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 821–840, 2016. doi:10.1137/1.9781611974331.CH59.

- [EP01] Michael Elkin and David Peleg. The client-server 2-spanner problem with applications to network design. In *Proceedings of the 8th International Colloquium on Structural Information and Communication Complexity (SIROCCO)*, volume 8 of *Proceedings in Informatics*, pages 117–132. Carleton Scientific, 2001.
- [FG09] Mohammad Farshi and Joachim Gudmundsson. Experimental study of geometric t -spanners. *ACM Journal of Experimental Algorithmics*, 14, 2009. doi:10.1145/1498698.1564499.
- [FLN03] Ronald Fagin, Amnon Lotem, and Moni Naor. Optimal aggregation algorithms for middleware. *Journal of Computer and System Sciences*, 66:614–656, 2003. doi:10.1145/375551.375567.
- [FS20] Arnold Filtser and Shay Solomon. The greedy spanner is existentially optimal. *SIAM J. Comput.*, 49(2):429–447, 2020. doi:10.1137/18M1210678.
- [GKK⁺10] Panos Giannopoulos, Rolf Klein, Christian Knauer, Martin Kutz, and Dániel Marx. Computing geometric minimum-dilation graphs is NP-hard. *International Journal of Computational Geometry & Applications*, 20(2):147–173, 2010. doi:10.1142/s0218195910003244.
- [GLN02] Joachim Gudmundsson, Christos Levcopoulos, and Giri Narasimhan. Fast greedy algorithms for constructing sparse geometric spanners. *SIAM Journal on Computing*, 31(5):1479–1500, 2002. doi:S0097539700382947.
- [GW22] Joachim Gudmundsson and Sampson Wong. Improving the dilation of a metric graph by adding edges. *ACM Transactions on Algorithms (TALG)*, 18(3):1–20, 2022. doi:10.1145/3517807.
- [HPM06] Sarel Har-Peled and Manor Mendel. Fast construction of nets in low-dimensional metrics and their applications. *SIAM Journal on Computing*, 35(5):1148–1184, 2006. doi:10.1137/S0097539704446281.
- [Kei88] J. Mark Keil. Approximating the complete Euclidean graph. In *Proc. 1st Scandinavian Workshop on Algorithm Theory (SWAT)*, volume 318 of *LNCS*, pages 208–213. Springer, 1988. doi:10.1007/3-540-19487-8_23.
- [KK06] Rolf Klein and Martin Kutz. Computing geometric minimum-dilation graphs is NP-hard. In *Proc. 14th International Symposium on Graph Drawing*, volume 4372 of *LNCS*, pages 196–207. Springer, 2006. doi:10.1007/978-3-540-70904-6_20.
- [Kor01] Guy Kortsarz. On the hardness of approximating spanners. *Algorithmica*, 30:432–450, 2001. doi:10.1007/S00453-001-0021-Y.
- [KP94] Guy Kortsarz and David Peleg. Generating sparse 2-spanners. *Journal of Algorithms*, 17(2):222–236, 1994. doi:10.1006/JAGM.1994.1032.
- [LS22] Hung Le and Shay Solomon. Truly optimal Euclidean spanners. *SIAM Journal on Computing*, 0(0):FOCS19–135–FOCS19–199, 2022. Preliminary version at FOCS 2019. doi:10.1137/20M1317906.

- [LS23] Hung Le and Shay Solomon. A unified framework for light spanners. In *Proc. 55th ACM Symposium on Theory of Computing (STOC)*, pages 295–308, 2023. doi:10.1145/3564246.3585185.
- [NS07] Giri Narasimhan and Michiel Smid. Geometric analysis: The leapfrog property. In *Geometric Spanner Networks*, chapter 14, pages 257–315. Cambridge University Press, 2007. doi:10.1017/CB09780511546884.
- [RS91] Jim Ruppert and Raimund Seidel. Approximating the d -dimensional complete Euclidean graph. In *Proc. 3rd Canadian Conference on Computational Geometry (CCCG)*, pages 207–210, 1991. URL: <https://cccg.ca/proceedings/1991/paper50.pdf>.
- [RS98] Satish B. Rao and Warren D. Smith. Approximating geometrical graphs via “spanners” and “banyans”. In *Proc. 30th ACM Symposium on Theory of Computing (STOC)*, STOC ’98, pages 540–550, 1998. Full version available here. doi:10.1145/276698.276868.
- [SZ04] Mikkel Sigurd and Martin Zachariasen. Construction of minimum-weight spanners. In *Proc. 12th European Symposium on Algorithms (ESA)*, volume 3221 of *LNCS*, pages 797–808. Springer, 2004. doi:10.1007/978-3-540-30140-0_70.
- [Won23] Sampson Wong. *Analysing trajectory similarity and improving graph dilation*. PhD thesis, University of Sydney, 2023. URL: <https://hdl.handle.net/2123/31331>.
- [Yao82] Andrew Chi-Chih Yao. Space-time tradeoff for answering range queries (extended abstract). In *Proc. 14th ACM Symposium on Theory of Computing (STOC)*, pages 128–136, 1982. doi:10.1145/800070.802185.