Balancing Degree, Diameter and Weight in Euclidean Spanners [*]

Shay Solomon [†§]
Michael Elkin[†‡]

Abstract

In a seminal STOC'95 paper, Arya et al. [4] devised a construction that for any set S of n points in $\mathbb{R}^d$ and any $\epsilon > 0$, provides a $(1+\epsilon)$-spanner with diameter $O(\log n)$, weight $O(\log^2 n) \cdot w(MST(S))$, and constant maximum degree. Another construction of [4] provides a $(1+\epsilon)$-spanner with $O(n)$ edges and diameter $O(\alpha(n))$, where α stands for the inverse Ackermann function. There are also a few other known constructions of $(1+\epsilon)$-spanners. Das and Narasimhan [20] devised a construction with constant maximum degree and weight $O(w(MST(S)))$, but the diameter may be arbitrarily large. In another construction by Arya et al. [4] there is diameter $O(\log n)$ and weight $O(\log n) \cdot w(MST(S))$, but it may have arbitrarily large maximum degree. While these constructions address some important practical scenarios, they fail to address situations in which we are prepared to compromise on one of the parameters, but cannot afford this parameter to be arbitrarily large.

In this paper we devise a novel *unified* construction that trades between the maximum degree, diameter and weight gracefully. For a positive integer k, our construction provides a $(1+\epsilon)$-spanner with maximum degree $O(k)$, diameter $O(\log_k n + \alpha(k))$, weight $O(k \cdot \log_k n \cdot \log n) \cdot w(MST(S))$, and $O(n)$ edges. Note that for $k = O(1)$ this gives rise to maximum degree $O(1)$, diameter $O(\log n)$ and weight $O(\log^2 n) \cdot w(MST(S))$, which is one of the aforementioned results of [4]. For $k = n^{1/\alpha(n)}$ this gives rise to diameter $O(\alpha(n))$, weight $O(n^{1/\alpha(n)} \cdot \log n \cdot \alpha(n)) \cdot w(MST(S))$ and maximum degree $O(n^{1/\alpha(n)})$. In the corresponding result from [4] the spanner has the same number of edges and diameter, but its weight and degree may be arbitrarily large. Our bound of $O(\log_k n + \alpha(k))$ on the diameter is optimal under the constraints that the maximum degree is $O(k)$ and the number of edges is $O(n)$. Similarly to the bound of Arya et al. [4], our bound on the weight is optimal up to a factor of $\log n$. Our construction also provides a similar tradeoff in the complementary range of parameters, i.e., when the weight should be smaller than $\log^2 n$, but the diameter is allowed to grow beyond $\log n$.

For random point sets in the d-dimensional unit cube, we "shave" a factor of $\log n$ from the weight bound. Specifically, in this case our construction achieves maximum degree $O(k)$, diameter $O(\log_k n + \alpha(k))$ and weight that is with high probability $O(k \cdot \log_k n) \cdot w(MST(S))$.

Finally, en route to these results we devise optimal constructions of 1-spanners for general tree metrics, which are of independent interest.

---

[*] A preliminary version of this paper appeared in ESA'10.

[†] Department of Computer Science, Ben-Gurion University of the Negev, POB 653, Beer-Sheva 84105, Israel.
E-mail: {`shayso,elkinm`}`@cs.bgu.ac.il`

Both authors are partially supported by the Lynn and William Frankel Center for Computer Sciences.

§This research has been supported by the Clore Fellowship grant No. 81265410.

[‡] This research has been supported by the BSF grant No. 2008430.

1 Introduction

1.1 Euclidean Spanners

Consider the weighted complete graph $\mathcal{S} = (S, \binom{S}{2})$ induced by a set S of n points in $\mathbb{R}^d, d \geq 2$. The weight of an edge $(x, y) \in \binom{S}{2}$, for a pair of distinct points $x, y \in S$, is defined to be the Euclidean distance $\|x - y\|$ between x and y. Let $G = (S, E)$ be a spanning subgraph of $\mathcal{S}$, with $E \subseteq \binom{S}{2}$, and assume that exactly as in $\mathcal{S}$, for any edge $e = (x, y) \in E$, its weight $w(e)$ in G is defined to be $\|x - y\|$. For a parameter $\epsilon > 0$, the spanning subgraph G is called a $(1 + \epsilon)$-*spanner* for the point set S if for every pair $x, y \in S$ of distinct points, the distance $dist_G(x, y)$ between x and y in G is at most $(1 + \epsilon) \cdot \|x - y\|$. Euclidean spanners were introduced[1] in 1986 by Chew [17]. Since then they evolved into an important subarea of Computational Geometry [33, 19, 41, 34, 3, 18, 20, 4, 21, 6, 40, 1, 11, 23]. (See also the book by Narasimhan and Smid on Euclidean spanners [37], and the references therein.) Also, Euclidean spanners have numerous applications in geometric approximation algorithms [40, 28, 29], geometric distance oracles [28, 30, 29], Network Design [32, 36] and in other areas.

In many of these applications one is required to construct a $(1 + \epsilon)$-spanner $G = (S, E)$ that satisfies a number of useful properties. First, the spanner should contain $O(n)$ (or nearly $O(n)$) edges. Second, its *weight* $w(G) = \sum_{e \in E} w(e)$ should not be much greater than the weight $w(MST(S))$ of the minimum spanning tree $MST(S)$ of S. Third, its *diameter* $\Lambda = \Lambda(G)$ should be small, i.e., for every pair of points $x, y \in S$ there should exist a path P in G that contains at most Λ edges and has weight $w(P) = \sum_{e \in E(P)} w(e) \leq (1 + \epsilon) \cdot \|x - y\|$. Fourth, its *maximum degree* (henceforth, *degree*) $\Delta(G)$ should be small.

In a seminal STOC'95 paper that culminated a long line of research, Arya et al. [4] devised a construction of $(1 + \epsilon)$-spanners with lightness[2] $O(\log^2 n)$, diameter $O(\log n)$ and constant degree. They also devised a construction of $(1 + \epsilon)$-spanners with diameter $O(\alpha(n))$ (respectively, $O(1)$) and $O(n)$ (resp., $O(n \cdot \log^* n)$) edges, where α stands for the inverse Ackermann function. However, in the latter construction the resulting spanners may have arbitrarily large (i.e., $\Omega(n)$) lightness and degree. There are also a few other known constructions of $(1 + \epsilon)$-spanners. Das and Narasimhan [20] devised a construction with constant degree and lightness, but the diameter may be arbitrarily large. (See also [27] for a faster implementation of a spanner construction with constant degree and lightness.) There is also another construction by Arya et al. [4] that guarantees that both the diameter and the lightness are $O(\log n)$, but the degree may be arbitrarily large. While these constructions address some important practical scenarios, they certainly do not address all of them. In particular, they fail to address situations in which we are prepared to compromise on one of the parameters, but cannot afford this parameter to be arbitrarily large.

In this paper we devise a novel *unified* construction that trades between the degree, diameter and weight gracefully. For a positive integer k, our construction provides a $(1 + \epsilon)$-spanner with degree $O(k)$, diameter $O(\log_k n + \alpha(k))$, lightness $O(k \cdot \log_k n \cdot \log n)$, and $O(n)$ edges. Also, we can improve the bound on the diameter from $O(\log_k n + \alpha(k))$ to $O(\log_k n)$, at the expense of increasing the number of edges from $O(n)$ to $O(n \cdot \log^* n)$. Note that for $k = O(1)$ our tradeoff gives rise to degree $O(1)$, diameter $O(\log n)$ and lightness $O(\log^2 n)$, which is one of the aforementioned results of [4]. Also, for $k = n^{1/\alpha(n)}$ it gives rise to a spanner with degree $O(n^{1/\alpha(n)})$, diameter $O(\alpha(n))$ and lightness $O(n^{1/\alpha(n)} \cdot \log n \cdot \alpha(n))$. In the corresponding result from [4] the spanner has the same number of edges and diameter, but its lightness and degree may be arbitrarily large.

In addition, we can achieve lightness $o(\log^2 n)$ at the expense of increasing the diameter. Specifically, for a parameter k the second variant of our construction provides a $(1 + \epsilon)$-spanner with degree $O(1)$, diameter $O(k \cdot \log_k n)$, and lightness $O(\log_k n \cdot \log n)$. For example, for $k = \log^\delta n$, for an arbitrarily small constant $\delta > 0$, we get a $(1 + \epsilon)$-spanner with degree $O(1)$, diameter $O(\log^{1+\delta} n)$ and lightness $O(\frac{\log^2 n}{\log \log n})$.

Our unified construction can be implemented in $O(n \cdot \log n)$ time. This matches the state-of-the-art

---

[1] The notion "spanner" was coined by Peleg and Ullman [38], who also introduced spanners for general graphs.

[2] For convenience, we will henceforth refer to the normalized notion of weight $\Psi(G) = \frac{w(G)}{w(MST(S))}$, which we call *lightness*.

	[4]	[4]	**New**	**New**	**New**	**New**	**New**	**New**
	I,II		**I**	**I**	**I**	**II**	**II**	**II**
k	1		$\log^{\delta} n$	$2^{\sqrt{\log n}}$	$n^{1/\alpha(n)}$	$\log^{\delta} n$	$2^{\sqrt{\log n}}$	n^{ζ}
Δ	1	n	$\log^{\delta} n$	$2^{\sqrt{\log n}}$	$n^{1/\alpha(n)}$	1	1	1
Λ	$\log n$	$\alpha(n)$	$\frac{\log n}{\log\log n}$	$\sqrt{\log n}$	$\alpha(n)$	$\log^{1+\delta} n$	$2^{O(\sqrt{\log n})}$	n^{ζ}
Ψ	$\log^2 n$	n	$\log^{2+\delta} n$	$2^{O(\sqrt{\log n})}$	$n^{O(1/\alpha(n))}$	$\frac{\log^2 n}{\log\log n}$	$\log^{3/2} n$	$\log n$

Table 1: A concise comparison of previous and new results. Each column corresponds to a set of parameters that can be achieved simultaneously. For each column the first row indicates whether the result is new or due to [4]. (The first column is due to [4], but can also be achieved from both our tradeoffs.) For new results, the second row indicates whether it is obtained by the first (I) or the second (II) tradeoff. (The first tradeoff is degree $O(k)$, diameter $O(\log_k n + \alpha(k))$, and lightness $O(k \cdot \log_k n \cdot \log n)$. The second tradeoff is degree $O(1)$, diameter $O(k \cdot \log_k n)$ and lightness $O(\log_k n \cdot \log n)$.) The third row indicates the value of k that is substituted in the corresponding tradeoff. The next three rows indicate the resulting degree (Δ), diameter (Λ) and lightness (Ψ). The number of edges used in all constructions is $O(n)$. To save space, the O notation is omitted everywhere except for the exponents. The letters δ and ζ stand for arbitrarily small positive constants.

running time of the aforementioned constructions [4, 27]. See Table 1 for a concise comparison of previous and new results.

Note that in any construction of spanners with degree $O(k)$, the diameter is $\Omega(\log_k n)$. Also, Chan and Gupta [11] showed that any $(1+\epsilon)$-spanner with $O(n)$ edges must have diameter $\Omega(\alpha(n))$. Consequently, our upper bound of $O(\log_k n + \alpha(k))$ on the diameter is tight under the constraints that the degree is $O(k)$ and the number of edges is $O(n)$. If we allow $O(n \cdot \log^* n)$ edges in the spanner, then our bound on the diameter is reduced to $O(\log_k n)$, which is again tight under the constraint that the degree is $O(k)$.

In addition, Dinitz et al. [23] showed that for any construction of spanners, if the diameter is at most $O(\log_k n)$, then the lightness is at least $\Omega(k \cdot \log_k n)$ and vice versa, if the lightness is at most $O(\log_k n)$, then the diameter is at least $\Omega(k \cdot \log_k n)$. This lower bound implies that the bound on lightness in both our tradeoffs cannot possibly be improved by more than a factor of $\log n$. The same slack of $\log n$ is present in the result of [4] that guarantees lightness $O(\log^2 n)$, diameter $O(\log n)$ and constant degree.

1.1.1 Euclidean Spanners for Random Point Sets

For random point sets in the d-dimensional unit cube (henceforth, unit cube), we "shave" a factor of $\log n$ from the lightness bound in both our tradeoffs, and show that the first (respectively, second) variant of our construction achieves maximum degree $O(k)$ (resp., $O(1)$), diameter $O(\log_k n + \alpha(k))$ (resp., $O(k \cdot \log_k n)$) and lightness that is with high probability (henceforth, w.h.p.) $O(k \cdot \log_k n)$ (resp., $O(\log_k n)$). Note that for $k = O(1)$ both these tradeoffs give rise to degree $O(1)$, diameter $O(\log n)$ and lightness (w.h.p.) $O(\log n)$. In addition to these tradeoffs, we can get a construction of $(1 + \epsilon)$-spanners with diameter $O(\log n)$ and lightness (w.h.p.) $O(1)$.

1.1.2 Spanners for Doubling Metrics

The *doubling dimension* of a metric (X, δ) is the smallest value ζ such that every ball B in the metric can be covered by at most 2^{ζ} balls of half the radius of B. The metric (X, δ) is called *doubling* if its doubling dimension ζ is constant. Spanners for doubling metrics have received much attention in recent years (see, e.g., [12, 31, 11, 26]). In particular, Chan et al. [12] showed that for any doubling metric (X, δ) there exists a $(1 + \epsilon)$-spanner with constant maximum degree, but this spanner may have arbitrarily large diameter. In addition, Chan and Gupta [11] devised a construction of $(1 + \epsilon)$-spanners for doubling metrics that achieves the optimal tradeoff between the number of edges and diameter, but these spanners may have arbitrarily large degree. We present a single construction of $O(1)$-spanners for doubling metrics that achieves the optimal tradeoff between the degree, diameter and number of edges in the entire range of parameters. Specifically, for a parameter k, our construction provides an $O(1)$-spanner with maximum degree $O(k)$, diameter $O(\log_k n + \alpha(k))$, and $O(n)$ edges. Also, we can improve the bound on the diameter from $O(\log_k n + \alpha(k))$ to $O(\log_k n)$, at the expense of increasing the number of edges from $O(n)$ to $O(n \cdot \log^* n)$. More generally, we can achieve the same optimal tradeoff between the number of edges and diameter as the spanners of [11] do, while also having the optimal maximum degree. The drawback is, however, that the stretch of our spanners is $O(1)$ rather than $1 + \epsilon$.

1.2 Spanners for Tree Metrics

Let ϑ_n be the metric induced by n points $v_1, v_2, \ldots, v_n$ lying on the x-axis with coordinates $1, 2, \ldots, n$, respectively. In a classical STOC'82 paper [48], Yao showed that there exists a 1-spanner[3] $G = (V, E)$ for ϑ_n with diameter $O(\alpha(n))$ and $O(n)$ edges, and that this is tight. Chazelle [15] extended the result of [48] to arbitrary tree metrics. Other proofs of Chazelle's result appeared in [2, 8, 47, 43]. Thorup [47] also devised an efficient parallel algorithm for computing this 1-spanner. The problem was also studied for planar metrics [46], general metrics [45] and even for general graphs [7]. (See also Chapter 12 in [37] for an excellent survey on this problem.) The problem is also closely related to the extremely well-studied problem of computing partial-sums. (See the papers of Tarjan [44], Yao [48], Chazelle and Rosenberg [16], Pătraşcu and Demaine [39], and the references therein.) For a discussion about the relationship between these two problems see the introduction of [1].

In all constructions [48, 15, 2, 8, 47, 43] of 1-spanners for tree metrics, the degree and lightness of the resulting spanner may be arbitrarily large. Moreover, the constraint that the diameter is $O(\alpha(n))$ implies that the degree must be $n^{\Omega(1/\alpha(n))}$. A similar lower bound on the lightness follows from the result of [23].

En route to our tradeoffs for Euclidean spanners, we have extended the results of [48, 15, 2, 8, 47, 43] and devised a construction that achieves the *optimal* (up to constant factors) *tradeoff between all involved parameters*. Specifically, consider an n-vertex tree T of degree $\Delta(T)$, and let k be a positive integer. Our construction provides a 1-spanner for the tree metric M_T induced by T with degree $O(\Delta(T)+k)$, diameter $O(\log_k n + \alpha(k))$, lightness $O(k \cdot \log_k n)$, and $O(n)$ edges. We can also get a spanner with diameter $O(\log_k n)$, $O(n \cdot \log^* n)$ edges, and the same degree and lightness as above. For the complementary range of diameter, the second variant of our construction provides a 1-spanner with degree $O(\Delta(T))$, diameter $O(k \cdot \log_k n)$, lightness $O(\log_k n)$, and $O(n)$ edges. As was mentioned above, both these tradeoffs are *optimal up to constant factors*.

We show that this general tradeoff between various parameters of 1-spanners for tree metrics is useful for deriving new results (and improving existing results) in the context of Euclidean spanners and spanners for doubling metrics. We anticipate that this tradeoff would be found useful in the context of partial sums problems, and for other applications.

1.3 Our and Previous Techniques

The starting point for our construction is the construction of Arya et al. [4] that achieves diameter $O(\log n)$, lightness $O(\log^2 n)$ and constant degree. The construction of [4] is built in two stages. First, a construction for the 1-dimensional case is devised. Then the 1-dimensional construction is extended to arbitrary constant dimension. For 1-dimensional spaces Arya et al. [4] start with devising a construction of 1-spanners with diameter, lightness and degree all bounded by $O(\log n)$. This construction is quite simple; it is essentially a flattened version of a deterministic skip-list. Next, by a more involved argument they show that the degree can be reduced to $O(1)$, at the expense of increasing the stretch parameter from 1 to $1 + \epsilon$. Finally, the generalization of their construction to point sets in the plane (or, more generally, to $\mathbb{R}^d$) is far more involved. Specifically, to this end Arya et al. [4] employed two main tools. The first one is the *dumbbell trees*, the theory of which was developed by Arya et al. in the same paper [4]. (See also Chapter 11 of [37].) The second one is the bottom-up clustering technique that was developed by Frederickson [25] for topology trees. Roughly speaking, the *Dumbbell Theorem* of [4] states that for

---

[3] The graph G is said to be a *1-spanner* for ϑ_n if for every pair of distinct vertices $v_i, v_j \in V$, the distance between them in G is equal to their distance $\|i - j\|$ in ϑ_n. Yao stated this problem in terms of partial sums. However, the two statements of the problem are equivalent.

every point set S, one can construct a forest $\mathcal{D}$ of $O(1)$ dumbbell trees, in which there exists a tree $T \in \mathcal{D}$ for every pair x, y of points from S, such that the distance $dist_T(x, y)$ between x and y in T is at most $(1+\epsilon)$ times their Euclidean distance $\|x-y\|$. Arya et al. employ Frederickson's clustering technique on each of these $O(1)$ dumbbell trees to obtain their ultimate spanner.

Similarly to [4], we start with devising a construction of 1-spanners for the 1-dimensional case. However, our construction achieves both diameter and lightness at most $O(\log n)$, in conjunction with the *optimal degree* of at most 3.[4] (Note that [4] paid for decreasing the degree from $O(\log n)$ to $O(1)$ by increasing the stretch of the spanner from 1 to $1+\epsilon$. Our construction achieves stretch 1 in conjunction with logarithmic diameter and lightness, and with the optimal degree.) Moreover, our construction is far more general, as it provides the *entire suite* of all possible values of diameter, lightness and degree, and it is *optimal up to constant factors* in the entire range of parameters. We then proceed to extending it to arbitrary tree metrics. Finally, we employ the dumbbell trees of Arya et al. [4]. Specifically, we construct our 1-spanners for the metrics induced by each of these dumbbell trees, and return their union as our ultimate spanner. As a result we obtain a *unified* construction of Euclidean spanners that achieves near-optimal tradeoffs in the entire range of parameters. We remark that it is unclear whether the construction of Arya et al. [4] can be extended to provide additional combinations between the diameter and lightness other than $O(\log n)$ and $O(\log^2 n)$, respectively; roughly speaking, the logarithms there come from the number of levels in Frederickson's topology trees. In particular, the construction of Arya et al. [4] that achieves diameter $O(\alpha(n))$ and arbitrarily large lightness and degree is based on completely different ideas. On the other hand, our construction yields a stronger result (diameter $O(\alpha(n))$, lightness and degree $n^{O(1/\alpha(n))}$), and this result is obtained by substituting a different parameter into one of our tradeoffs. Moreover, our construction is much simpler and more modular than that of [4]. In particular, it does not employ Frederickson's bottom-up clustering technique, but rather constructs 1-spanners for dumbbell trees directly.

Also, our construction of 1-spanners for tree metrics (that we use for dumbbell trees) is fundamentally different from the previous constructions due to [48, 15, 2, 8, 47, 43]. In particular, the techniques of [15, 2, 8, 47, 43] for generalizing constructions of 1-spanners from 1-dimensional metrics to general tree metrics ensure that the diameter of the resulting spanners is not (much) greater than the diameter in the 1-dimensional case. However, the degree and/or lightness of spanners for tree metrics that are obtained by these techniques may be arbitrarily large. To overcome this obstacle we adapt the techniques of [15, 2, 8, 43] to our purposes. Next, we overview this adaptation. A central ingredient in the generalization techniques of [15, 2, 8, 43] is a tree decomposition procedure. Given an n-vertex rooted tree (T, rt) and a parameter k, this procedure computes a set C of $O(k)$ *cut vertices*. This set has the property that removing all vertices of C from the tree T decomposes T into a collection $\mathcal{F}$ of trees, so that each tree $\tau \in \mathcal{F}$ contains $O(n/k)$ vertices. This decomposition induces a tree $\mathcal{Q} = \mathcal{Q}(\tau, C)$ over the vertex set $C \cup \{rt\}$ in a natural way: a cut vertex $w \in C$ is defined to be a child of its closest ancestor in T that belongs to $C \cup \{rt\}$. For our purposes, it is crucial that the degree of the tree $\mathcal{Q}$ will not be (much) greater than the degree of T. In addition, it is essential that each tree $\tau \in \mathcal{F}$ will be incident to at most $O(1)$ cut vertices. We devise a novel decomposition procedure that guarantees these two basic properties. Intuitively, our decomposition procedure "slices" the tree in a "path-like" fashion. This path-like nature of our decomposition enables us to keep the degree and lightness of our construction for general tree metrics (essentially) as small as in the 1-dimensional case.

1.4 Structure of the Paper

In Section 2 we describe our construction of 1-spanners for tree metrics. Therein we start (Section 2.1) with outlining our basic scheme. We proceed (Section 2.2) with describing our 1-dimensional construction. In Section 2.3 we extend this construction to general tree metrics. Our tree decomposition procedure (which is in the heart of this extension) is described in Section 2.3.1. In Section 3 we derive our results

---

[4] Observe that any graph (not necessarily 1-spanner) with maximum degree 2 must have diameter at least $\frac{n-1}{2}$.

for Euclidean spanners and spanners for doubling metrics.

1.5 Preliminaries

An *n-point metric space* $M = (V, dist)$ can be viewed as the complete graph $G(M) = (V, \binom{V}{2}, dist)$ in which for every pair of points $x, y \in V$, the weight of the edge $e = (x, y)$ in $G(M)$ is defined by $w(x, y) = dist(x, y)$. Let G be a spanning subgraph of M. We say that G is a *t-spanner* for M if for every pair $x, y \in V$ of distinct points, there exists a path in G between x and y whose weight (i.e., the sum of all edge weights in it) is at most $t \cdot dist(x, y)$. Such a path is called a *t-spanner path*. The *stretch* of G is the minimum number t, such that G is a t-spanner for M. Let T be an arbitrary tree, and denote by $V(T)$ the vertex set of T. For any two vertices u, v in T, their (weighted) distance in T is denoted by $dist_T(u, v)$. The tree metric M_T induced by T is defined as $M_T = (V(T), dist_T)$. The *size* of T, denoted $|T|$, is the number of vertices in T. Finally, for a positive integer n, we denote the set $\{1, 2, \ldots, n\}$ by $[n]$.

2 1-Spanners for Tree Metrics

2.1 The Basic Scheme

Consider an arbitrary n-vertex (weighted) rooted tree (T, rt), and let M_T be the tree metric induced by T. Clearly, T is both a 1-spanner and an MST of M_T, but its diameter may be arbitrarily large. We would like to reduce the diameter of this 1-spanner by adding to it some edges. On the other hand, the number of edges of the resulting spanner should still be linear in n. Moreover, the lightness and the maximum degree of the resulting spanner should also be reasonably small.

Let H be a spanning subgraph of M_T. The *monotone distance* between any two points u and v in H is defined as the minimum number of edges in a 1-spanner path in H connecting them. Two points in M_T are called *comparable* if one is an ancestor of the other in the underlying tree T. The *monotone diameter* (respectively, *comparable monotone diameter*) of H, denoted $\Lambda(H)$ (resp., $\bar{\Lambda}(H)$), is defined as the maximum monotone distance in H between any two points (resp., any two comparable points) in M_T. Observe that if any two *comparable* points are connected via a 1-spanner path that consists of at most h edges, then any two *arbitrary* points are connected via a 1-spanner path that consists of at most $2h$ edges. Consequently, $\bar{\Lambda}(H) \le \Lambda(H) \le 2 \cdot \bar{\Lambda}(H)$. We henceforth restrict the attention to comparable monotone diameter in the sequel.

Let $k \ge 2$ be a fixed parameter. The first ingredient of the algorithm is to select a set of $O(k)$ *cut vertices* whose removal from T partitions it into a collection of subtrees of size $O(n/k)$ each. (As mentioned in the last paragraph of Section 1.3, we also require this set to satisfy several additional properties.) Having selected the cut vertices, the next step of the algorithm is to connect the cut vertices via $O(k)$ edges, so that the monotone distance between any pair of comparable cut vertices will be small. (This phase does not involve a recursive call of the algorithm.) Finally, the algorithm calls itself recursively for each of the subtrees.

We insert all edges of the original tree T into our final spanner H. These edges connect between cut vertices and subtrees in the spanner. We remark that the spanner contains no other edges that connect between cut vertices and subtrees. Moreover, the spanner contains no edges that connect between different subtrees.

2.2 1-Dimensional Spaces

In this section we devise an optimal construction of 1-spanners for ϑ_n. (Recall that ϑ_n is the metric induced by n points $v_1, v_2, \ldots, v_n$ lying on the x-axis with coordinates $1, 2, \ldots, n$, respectively.) Our argument extends easily to any 1-dimensional space.

Denote by P_n the path $(v_1, v_2), (v_2, v_3), \ldots, (v_{n-1}, v_n)$ that induces the metric ϑ_n. We remark that the edges of P_n (henceforth, *path-edges*) belong to all spanners that we construct.

2.2.1 Selecting the Cut-Vertices

Let $k \geq 2$ be a fixed parameter. The task of selecting the cut vertices in the 1-dimensional case is trivial. (We assume for simplicity that n is an integer power of k.) In addition to the two endpoints v_1 and v_n of the path, we select the $k-1$ points $r_1, r_2, \ldots, r_{k-1}$ to be cut vertices, where for each $i \in [k-1]$, $r_i = v_{i(n/k)}$. Indeed, by removing the $k+1$ cut vertices $r_0 = v_1, r_1, \ldots, r_{k-1}, r_k = v_n$ from the path (along with their incident edges), we are left with k intervals $I_1, I_2, \ldots, I_k$ of length at most n/k each. The two endpoints v_1 and v_n of the path are called the *sentinels*, and they play a special role in the construction. (See Figure 1 for an illustration for the case $k = 2$.)

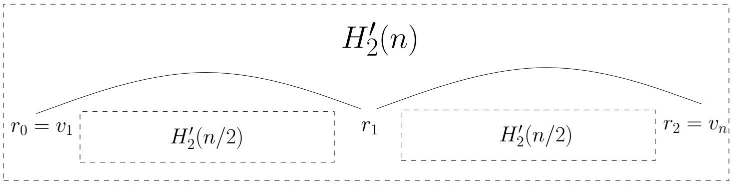

Figure 1: The construction for $k = 2$. Only the first level of the recursion is illustrated. (Path-edges are not depicted in the figure.) The cut vertex $r_1 = v_{n/2}$ is connected via edges to the two sentinels v_1 and v_n. The construction proceeds recursively for each of the two intervals I_1 and I_2.

2.2.2 1-Spanners with Low Diameter

In this section we devise a construction $H_k(n)$ of 1-spanners for ϑ_n with comparable monotone diameter $\bar{\Lambda}(n) = \bar{\Lambda}(H_k(n))$ in the range $\Omega(\alpha(n)) = \bar{\Lambda}(n) = O(\log n)$. In Section 2.2.3 we turn our attention to spanners with larger monotone diameter.

First, the algorithm connects the $k+1$ cut vertices $r_0 = v_1, r_1, \ldots, r_{k-1}, r_k = v_n$ via one of the aforementioned constructions of 1-spanners from [48, 15, 2, 8, 47, 43] (henceforth, *list-spanner*). In other words, $O(k)$ edges between cut vertices are added to the spanner $H_k(n)$ to guarantee that the monotone distance in the spanner between any two cut vertices[5] will be $O(\alpha(k))$. Then the algorithm adds to the spanner $H_k(n)$ edges that connect each of the two sentinels to all other k cut vertices. Finally, the algorithm calls itself recursively for each of the intervals $I_1, I_2, \ldots, I_k$. At the bottom level of the recursion, i.e., when $n \leq k$, the algorithm uses the list-spanner to connect all points, and, in addition, it adds to the spanner edges that connect each of the two sentinels v_1 and v_n to all the other $n-1$ points. (See Figure 2 for an illustration.)

Denote by $E(n)$ the number of edges in $H_k(n)$, excluding edges of P_n. Clearly, $E(n)$ satisfies the recurrence $E(n) \leq O(k) + k \cdot E(n/k)$, with the base condition $E(q) = O(q)$, for all $q \leq k$, yielding $E(n) = O(n)$. Denote by $\Delta(n)$ the maximum degree of a vertex in $H_k(n)$, excluding edges of P_n. Clearly, $\Delta(n)$ satisfies the recurrence $\Delta(n) \leq \max\{k, \Delta(n/k)\}$, with the base condition $\Delta(q) \leq q-1$, for all $q \leq k$, yielding $\Delta(n) \leq k$. Including edges of P_n, the number of edges increases by $n-1$ units, and the maximum degree increases by at most two units.

Denote by $w(n)$ the weight of $H_k(n)$, excluding edges of P_n. Note that at most $O(k)$ edges are added between cut vertices. Each of these edges has weight at most $n-1$. The total weight of all edges within an interval I_i is at most $w(n/k)$. Hence $w(n)$ satisfies the recurrence $w(n) \leq O(n \cdot k) + k \cdot w(n/k)$, with the base condition $w(q) = O(q^2)$, for all $q \leq k$. It follows that $w(n) = O(n \cdot k \cdot \log_k n) = O(k \cdot \log_k n) \cdot w(MST(\vartheta_n))$. Including edges of P_n, the weight increases by $w(P_n) = n-1$ units.

Next, we show that the comparable monotone diameter $\bar{\Lambda}(n)$ of $H_k(n)$ is at most $O(\log_k n + \alpha(k))$. The *monotone radius* $R(n)$ of $H_k(n)$ is defined as the maximum monotone distance in $H_k(n)$ between one of the sentinels (either v_1 or v_n) and some other point in ϑ_n. Let v_j be a point in ϑ_n, and let i be the index such that $v_j \in \{r_i\} \cup I_i$. (In other words, i is the index such that $i(n/k) \leq j < (i+1)(n/k)$.) If

---

[5] In the 1-dimensional case any two points are comparable.

$j = i(n/k)$ then v_j is the cut vertex r_i; in this case the 1-spanner path $\Pi = \Pi(v_1, v_j)$ in $H_k(n)$ connecting the sentinel v_1 and the point v_j will consist of the single edge (v_1, v_j). Otherwise, $j > i(n/k)$ and $v_j \in I_i$. In this case the path Π will start with the two edges $(v_1, v_{i(n/k)})$, $(v_{i(n/k)}, v_{i(n/k)+1})$. The point $v_{i(n/k)+1}$ is a sentinel of the ith interval I_i. Hence, the path Π will continue recursively, from $v_{i(n/k)+1}$ to v_j. It follows that the monotone radius $R(n)$ satisfies the recurrence $R(n) \leq 2 + R(n/k)$, with the base condition $R(q) = 1$, for all $q \leq k$, yielding $R(n) = O(\log_k n)$. It is easy to verify that $\bar{\Lambda}(n)$ satisfies the recurrence $\bar{\Lambda}(n) \leq \max\{\bar{\Lambda}(n/k), O(\alpha(k)) + 2R(n/k)\}$, with the base condition $\bar{\Lambda}(q) = O(\alpha(q))$, for all $q \leq k$. Hence $\bar{\Lambda}(n) = O(\log_k n + \alpha(k))$.

Denote the worst-case running time of the algorithm by $t(n)$, excluding the time needed to add the edges of P_n to the spanner. We remark that the list-spanner of [48, 15, 2, 8, 47, 43] can be implemented in linear time. By construction, $t(n)$ satisfies the recurrence $t(n) \leq O(k) + k \cdot t(n/k)$, with the base condition $t(q) = O(q)$, for all $q \leq k$, yielding $t(n) = O(n)$. Hence, the overall running time of the algorithm is $O(n)$.

Finally, we remark that the maximum degree of this construction can be easily reduced from $k+2$ to $k+1$, without increasing any of the other parameters by more than a constant factor; the details of this technical argument are omitted. In particular, for $k = 2$ we will get this way the optimal degree 3, together with diameter and lightness $O(\log n)$; the same result also follows from Theorem 2.2 below.

Theorem 2.1 *For any n-point 1-dimensional space and a parameter $k \geq 2$, there exists a 1-spanner with maximum degree at most $k+1$, diameter $O(\log_k n + \alpha(k))$, lightness $O(k \cdot \log_k n)$, and $O(n)$ edges. The running time of this construction is $O(n)$.*

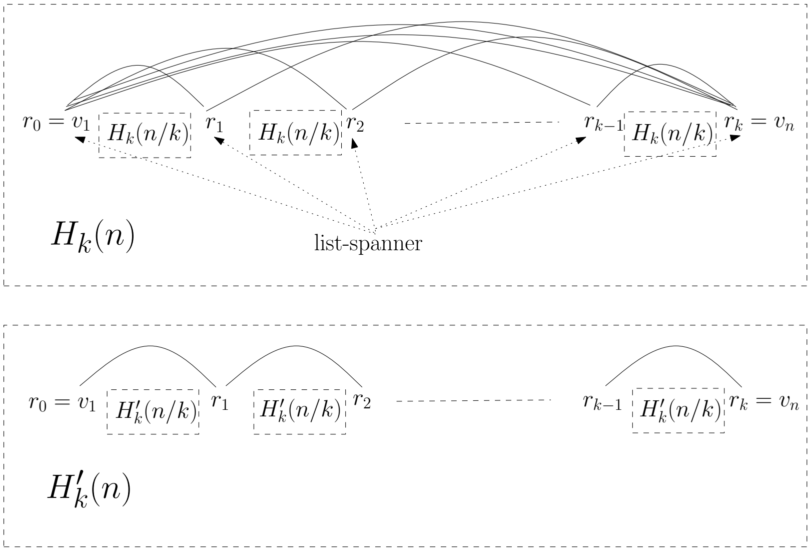

Figure 2: The constructions $H_k(n)$ and $H'_k(n)$ for a general parameter $k, k \geq 2$. Only the first level of the recursion is illustrated. (Path-edges are not depicted in the figure.) For $H_k(n)$, all the cut vertices are connected via the list-spanner, and, in addition, each of the two sentinels is connected to all other k cut vertices. For $H'_k(n)$, each cut vertex r_{i-1} is connected to the next cut vertex r_i in line, $i \in [k]$.

2.2.3 1-Spanners with High Diameter

In this section we devise a construction $H'_k(n)$ of 1-spanners for ϑ_n with comparable monotone diameter $\bar{\Lambda}'(n) = \bar{\Lambda}(H'_k(n))$ in the range $\bar{\Lambda}'(n) = \Omega(\log n)$.

The algorithm connects the $k+1$ cut vertices $r_0 = v_1, r_1, \ldots, r_{k-1}, r_k = v_n$ via a path of length k, i.e., it adds the edges $(r_0, r_1), (r_1, r_2), \ldots, (r_{k-1}, r_k)$ into the spanner. In addition, it calls itself recursively for each of the intervals $I_1, I_2, \ldots, I_k$. At the bottom level of the recursion, i.e., when $n \leq k$, the algorithm adds no additional edges to the spanner. (See Figures 1 and 2 for an illustration.)

Denote by $\Delta'(n)$ the maximum degree of a vertex in $H'_k(n)$, excluding edges of P_n. Clearly, $\Delta'(n)$ satisfies the recurrence $\Delta'(n) \leq \max\{2, \Delta'(n/k)\}$, with the base condition $\Delta'(q) = 0$, for all $q \leq k$, yielding $\Delta'(n) \leq 2$. Including edges of P_n, the maximum degree increases by at most two units, and so $\Delta(H'_k(n)) \leq 4$. Consequently, the number of edges in $H'_k(n)$ is no greater than $2n$.

Denote by $w'(n)$ the weight of $H'_k(n)$, excluding edges of P_n. Note that the weight of the path connecting all $k+1$ cut vertices is equal to $n-1$. The total weight of all edges within an interval I_i is at most $w'(n/k)$. Hence $w'(n)$ satisfies the recurrence $w'(n) \leq n - 1 + k \cdot w'(n/k)$, with the base condition $w'(q) \leq q - 1$, for all $q \leq k$. It follows that $w'(n) = O(n \cdot \log_k n) = O(\log_k n) \cdot w(MST(\vartheta_n))$. Including edges of P_n, the weight increases by $w(P_n) = n - 1$ units.

Note that the monotone radius $R'(n)$ of $H'_k(n)$ satisfies the recurrence $R'(n) \leq k + R'(n/k)$, with the base condition $R'(q) \leq q - 1$, for all $q \leq k$. Hence, $R'(n) = O(k \cdot \log_k n)$. Using reasoning similar to that of Section 2.2.2, we get that the comparable monotone diameter $\bar{\Lambda}'(n) = \bar{\Lambda}(H'_k(n))$ of $H'_k(n)$ satisfies the recurrence $\bar{\Lambda}'(n) \leq \max\{\bar{\Lambda}'(n/k), k + 2R'(n/k)\}$, with the base condition $\bar{\Lambda}'(q) \leq q - 1$, for all $q \leq k$. It follows that $\bar{\Lambda}'(n) = O(k \cdot \log_k n)$.

We remark that the spanner $H'_k(n)$ is a planar graph.

Denote the worst-case running time of the algorithm by $t'(n)$, excluding the time needed to add the edges of P_n to the spanner. It is easy to see that $t'(n)$ satisfies the recurrence $t'(n) \leq O(k) + k \cdot t'(n/k)$, with the base condition $t'(q) = O(1)$, for all $q \leq k$, yielding $t'(n) = O(n)$. Hence, the overall running time of the algorithm is $O(n)$.

Finally, similarly to the construction of Section 2.2.2, the maximum degree of this construction can be reduced from 4 to 3, without increasing any of the other parameters by more than a constant factor.

Theorem 2.2 *For any n-point 1-dimensional space and a parameter k, there exists a 1-spanner with maximum degree 3, diameter $O(k \cdot \log_k n)$, and lightness $O(\log_k n)$. Moreover, this 1-spanner is a planar graph. The running time of this construction is $O(n)$.*

2.3 General Tree Metrics

In this section we extend the constructions of Section 2.2 from line metrics to general tree metrics.

2.3.1 Selecting the Cut-Vertices

In this section we present a procedure for selecting, given a tree T, a subset of $O(k)$ vertices whose removal from the tree partitions it into subtrees of size $O(|T|/k)$ each. This subset will also satisfy several additional useful properties.

Let (T, rt) be a rooted tree. For an inner vertex v in T with $ch(v)$ children, we denote its children, from left to right, by $c_1(v), c_2(v), \ldots, c_{ch(v)}(v)$. Suppose without loss of generality that the size of the subtree $T_{c_1(v)}$ of v is no smaller than the size of any other subtree of v, i.e., $|T_{c_1(v)}| \geq |T_{c_2(v)}|, |T_{c_3(v)}|, \ldots, |T_{c_{ch(v)}(v)}|$. (This assumption can be guaranteed by a straightforward procedure that runs in linear time.) We say that the vertex $c_1(v)$ is the *left-most* child of v. Also, an edge in T is called *left-most* if it connects a vertex v in T and its left-most child $c_1(v)$. We denote by $P(v) = (v, c_1(v), \ldots, l(v))$ the path of left-most edges leading down from v to some leaf $l(v)$ in the subtree T_v of T rooted at v; the leaf $l(v)$ is referred to as the *left-most vertex* in T_v. Also, let $l(T) = l(rt)$ denote the left-most vertex in the entire tree T. An inner vertex v in T is called *d-balanced*, for $d \geq 1$, or simply *balanced* if d is clear from the context, if $|T_{c_1(v)}| \leq |T| - d$. The first (i.e., closest to v) balanced vertex along $P(v)$ is denoted by $b(v)$; if no vertex along $P(v)$ is balanced, we write $b(v) = NULL$. Observe that for $|T| \geq 2d$, we have $|T| - d \geq d \geq 1$; in this case the one-before-last vertex along $P(v)$ (namely, the parent $\pi(l(v))$ of $l(v)$ in T) is balanced. Hence, in this case $b(v) \neq NULL$.

Next, we present the Procedure CV (standing for cut vertices) that accepts as input a rooted tree (T, rt) and a parameter $d \geq 1$, and returns as output a subset of $V(T)$. If $|T| < 2d$, the procedure returns the empty set $\emptyset$. Otherwise $|T| \geq 2d$, and so the first balanced vertex $b = b(rt)$ along $P(rt)$ satisfies

$b \neq NULL$. In this case for each child $c_i(b)$ of b, $i \in [ch(b)]$, the procedure recursively constructs the subset $C_i = CV((T_{c_i(b)}, c_i(b)), d)$, and then returns as output the vertex set $\bigcup_{i=1}^{ch(b)} C_i \cup \{b\}$. (See Figure 3 for an illustration.)

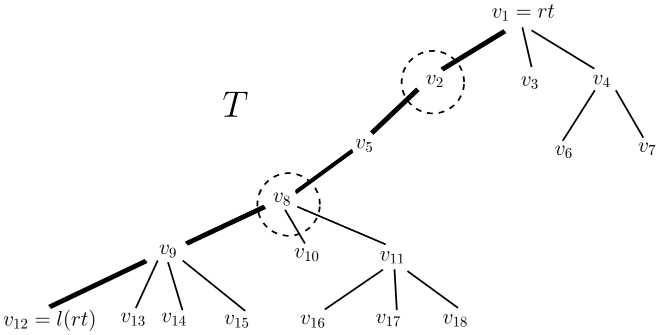

Figure 3: A rooted tree (T, rt) with $n = |T| = 18$ vertices $v_1 = rt, v_2, \ldots, v_{18}$. The edges of $P(rt)$ are depicted by bold lines. The first 6-balanced vertex along $P(rt)$ is v_2. The procedure CV on input (T, rt) and $d = 6$ returns the subset $\{v_2, v_8\}$.

It is easy to see that the running time of this procedure is linear in $|T|$.

Let (T, rt) be an n-vertex rooted tree, and let $d \geq 1$ be a fixed parameter. For convenience, we define $n_i = |T_{c_i(b)}|$, for each $i \in [ch(b)]$. Next, we analyze the properties of the set $C = CV((T, rt), d)$ of *cut vertices*.

Observe that for $n < 2d$, $C = \emptyset$, and for $n \geq 2d$, C is non-empty.
Next, we provide an upper bound on $|C|$ in the case $n \geq 2d$.

Lemma 2.3 *For $n \geq 2d$, $|C| \leq (n/d) - 1$.*

Proof: The proof is by induction on $n = |T|$.
Basis: $2d \leq n < 3d$. Fix an index $i \in [ch(b)]$. Since b is balanced, we have

$$n_i \ \leq \ n_1 \ \leq \ n - d \ < \ 2d,$$

implying that $C_i = \emptyset$. It follows that $C = \bigcup_{i=1}^{ch(b)} C_i \cup \{b\} = \{b\}$, and so $|C| = 1 \leq (n/d) - 1$.
Induction Step: We assume the correctness of the statement for all smaller values of n, $n \geq 3d$, and prove it for n. Let I be the set of all indices i in $[ch(b)]$ for which $n_i \geq 2d$. Observe that for each $i \in [ch(b)] \setminus I$, $C_i = \emptyset$, and by the induction hypothesis, for each $i \in I$, $|C_i| \leq (n_i/d) - 1$. By construction, the vertex sets $C_1, C_2, \ldots, C_{ch(b)}$ and $\{b\}$ are pairwise disjoint, and $C = \bigcup_{i=1}^{ch(b)} C_i \cup \{b\}$. Hence

$$|C| \ = \ \sum_{i=1}^{ch(b)} |C_i| + 1 \ = \ \sum_{i \in I} |C_i| + 1 \ \leq \ \sum_{i \in I} ((n_i/d) - 1) + 1. \tag{1}$$

The analysis splits into three cases depending on the value of $|I|$.
Case 1: $|I| = 0$. Equation (1) yields $|C| \leq 1 \leq (n/d) - 1$.
Case 2: $|I| = 1$. By construction, $n_1 \geq n_i$, for each $i \in [ch(b)]$, implying that $I = \{1\}$. Since b is balanced, $n_1 \leq n - d$, and so (1) yields

$$|C| \ \leq \ (n_1/d) - 1 + 1 \ \leq \ (n - d)/d \ = \ (n/d) - 1.$$

Case 3: $|I| \geq 2$. Clearly, $\sum_{i \in I} n_i \leq n - 1$, and so (1) yields

$$|C| \ \leq \ \sum_{i \in I} ((n_i/d) - 1) + 1 \ = \ \sum_{i \in I} (n_i/d) - |I| + 1 \ \leq \ (n - 1)/d - 2 + 1 \ \leq \ (n/d) - 1.$$

□

Let $b = b(rt)$, and let $\overline{T_b}$ be the subtree of T obtained by removing the subtree T_b from T. We use the following claim to prove Lemma 2.5.

Claim 2.4 $|\overline{T_b}| < d$.

Proof: If $b = rt$, then $\overline{T_b}$ is empty and the assertion of the claim is immediate. Otherwise, consider the parent $\pi(b)$ of b in T. Since b is the first (i.e., closest to rt) balanced vertex along $P(rt)$, $\pi(b)$ is non-balanced, and so $|T_b| = |T_{c_1(\pi(b))}| > n - d$. Hence $|\overline{T_b}| = n - |T_b| < d$, and we are done. □

For a subset U of $V(T)$, we denote by $T \setminus U$ the forest obtained from T by removing all vertices in U along with the edges that are incident to them.

Lemma 2.5 *The size of any subtree in the forest $T \setminus C$ is smaller than $2d$.*

Proof: The proof is by induction on $n = |T|$. The basis $n < 2d$ is trivial.
Induction Step: We assume the correctness of the statement for all smaller values of n, $n \geq 2d$, and prove it for n. First, note that $b = b(rt) \in C$. Also, observe that for $n \geq 2d$,

$$T \setminus C \ = \ \bigcup_{i=1}^{ch(b)} (T_{c_i(b)} \setminus C_i) \cup \{\overline{T_b}\}. \tag{2}$$

Consider a subtree T' in the forest $T \setminus C$. By (2), either $T' = \overline{T_b}$, or it belongs to the forest $T_{c_i(b)} \setminus C_i$, for some index $i \in [ch(b)]$. In the former case, the size bound follows from Claim 2.4, whereas in the latter case it follows from the induction hypothesis. □

Any subset U of $V(T)$ induces a forest $\mathcal{Q}(T, U)$ over U in the natural way: a vertex $v \in U$ is defined to be a child of its closest ancestor in T that belongs to U. Define $\mathcal{Q} = \mathcal{Q}(T, C)$. Observe that for $n < 2d$, $C = \emptyset$, and so $\mathcal{Q} = \emptyset$. Also, for $n \geq 2d$, C is non-empty and $b = b(rt) \neq NULL$.

Lemma 2.6 *For $n \geq 2d$, $\mathcal{Q}$ is a spanning tree of C rooted at $b = b(rt)$, such that for each vertex v in C, the number of children of v in $\mathcal{Q}$, denoted $ch_{\mathcal{Q}}(v)$, is no greater than the corresponding number $ch(v)$ in T.*

Remark: This lemma implies that $\Delta(\mathcal{Q}) \leq \Delta(T)$.
Proof: The proof is by induction on $n = |T|$.
Basis: $2d \leq n < 3d$. In this case $C = \{b\}$, and so $\mathcal{Q}$ consists of a single root vertex b.
Induction Step: We assume the correctness of the statement for all smaller values of n, $n \geq 3d$, and prove it for n. Let I be the set of all indices i in $[ch(b)]$ for which $n_i \geq 2d$, and write $I = \{i_1, i_2, \ldots, i_{|I|}\}$. Observe that for each index $i \in [ch(b)] \setminus I$, $C_i = \emptyset$, and so $\mathcal{Q}(T_{c_i(b)}, C_i)$ is an empty tree. By the induction hypothesis, for each $i \in I$, $\mathcal{Q}_i = \mathcal{Q}(T_{c_i(b)}, C_i)$ is a spanning tree of C_i rooted at $b_i = b(c_i(b)) \neq NULL$ in which the number of children of each vertex is no greater than the corresponding number in $T_{c_i(b)}$. By definition, the only children of b in $\mathcal{Q}$ are the roots $b_{i_1}, b_{i_2}, \ldots, b_{i_{|I|}}$ of the non-empty trees $\mathcal{Q}_{i_1}, \mathcal{Q}_{i_2}, \ldots, \mathcal{Q}_{i_{|I|}}$, respectively, and so $ch_{\mathcal{Q}}(b) = |I| \leq ch(b)$. In addition, b has no parent in $\mathcal{Q}$, and so it is the root of $\mathcal{Q}$. □

For a tree τ, the root $rt(\tau)$ of τ and its left-most vertex $l(\tau)$ are called the *sentinels* of τ. The next lemma shows that each subtree in the forest $T \setminus C$ is incident to at most two cut vertices. The proof of this lemma follows similar lines as those in the proof of Lemma 2.5, and is thus omitted.

Lemma 2.7 *For any subtree T' in the forest $T \setminus C$, no other vertex in T' other than its two sentinels $rt(T')$ and $l(T')$ is incident to a vertex from C. Moreover, both $rt(T')$ and $l(T')$ are incident to at most one vertex from C; specifically, $rt(T')$ is incident to its parent in T, unless $rt(T')$ is the root of T, and $l(T')$ is incident to its left-most child in T, unless $l(T')$ is a leaf in T.*

Similarly to the 1-dimensional case, we add the two sentinels $rt(T)$ and $l(T)$ of the original tree T to the set C of cut vertices. From now on we refer to the appended set $\tilde{C} = C \cup \{rt(T), l(T)\}$ as the set of *cut vertices*. Intuitively, Lemma 2.7 shows that the Procedure CV "slices" the tree in a "path-like" fashion, i.e., in a way that is analogous to the decomposition of ϑ_n into intervals described in Section 2.2.1. (See Figure 4 for an illustration.)

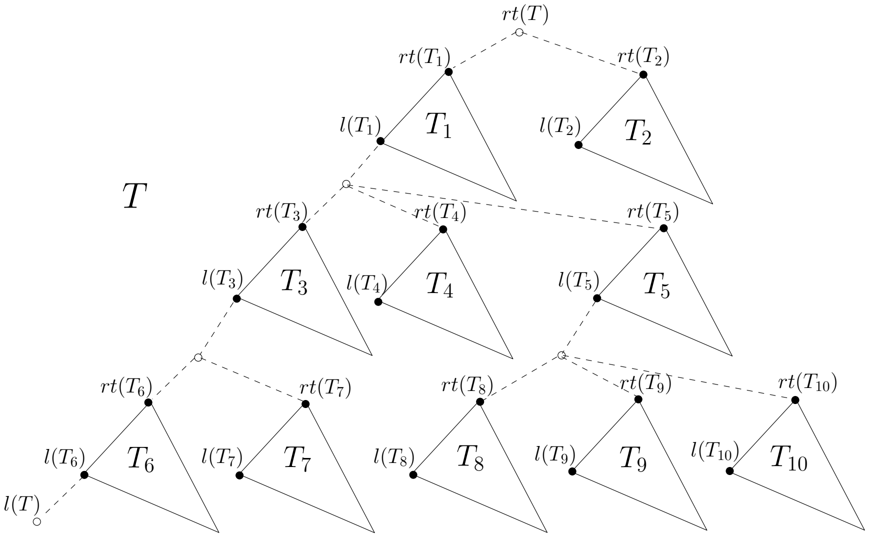

Figure 4: A "path-like" decomposition of the tree T into subtrees $T_1, T_2, \ldots, T_{10}$. The 5 cut vertices of $\tilde{C}$ (i.e., the 3 vertices of C and the 2 sentinels $rt(T)$ and $l(T)$ of T) are depicted in the figure by empty dots, whereas the 20 sentinels of the subtrees $T_1, T_2, \ldots, T_{10}$ are depicted by filled dots. Similarly to the 1-dimensional case, each subtree T_i is incident to at most two cut vertices. Edges in T that connect sentinels of subtrees with cut vertices are depicted by dashed lines.

Lemmas 2.3, 2.5, 2.6 and 2.7 imply the following corollary, which summarizes the properties of the set $\tilde{C}$ of cut vertices.

Corollary 2.8 1. *For $n \geq 2d$, $|\tilde{C}| \leq (n/d) + 1$.*

2. *The size of any subtree in the forest $T \setminus \tilde{C}$ is smaller than $2d$.*

3. *$\tilde{\mathcal{Q}} = \mathcal{Q}(T, \tilde{C})$ is a spanning tree of $\tilde{C}$ rooted at $rt(T)$, with $\Delta(\tilde{\mathcal{Q}}) \leq \Delta(T)$.*

4. *For any subtree T' in the forest $T \setminus \tilde{C}$, only the two sentinels $rt(T')$ and $l(T')$ of T' are incident to a vertex from $\tilde{C}$. Moreover, both $rt(T')$ and $l(T')$ are incident to at most one vertex from $\tilde{C}$; specifically, $rt(T')$ is incident to its parent in T, and $l(T')$ is incident to its left-most child in T, unless $l(T')$ is a leaf in T.*

Remark: The running time of the Procedure CV is $O(n)$. Hence the set $\tilde{C}$ of cut vertices can be computed in linear time.

2.3.2 1-Spanners with Low Diameter

Consider an n-vertex (weighted) tree T, and let M_T be the tree metric induced by T. In this section we devise a construction $\mathcal{H}_k(n)$ of 1-spanners for M_T with comparable monotone diameter $\bar{\Lambda}(n) = \bar{\Lambda}(\mathcal{H}_k(n))$ in the range $\Omega(\alpha(n)) = \bar{\Lambda}(n) = O(\log n)$. Both in this construction and in the one of Section 2.3.3, all edges of the original tree T are added to the spanner.

Let k be a fixed parameter such that $4 \leq k \leq n/2 - 1$, and set $d = n/k$. (Observe that $n \geq 2k + 2$ and $d > 2$.) To select the set $\tilde{C}$ of cut vertices, we invoke the procedure CV on the input (T, rt) and d. Set $C = CV((T, rt), d)$ and $\tilde{C} = C \cup \{rt(T), l(T)\}$. Since $k \geq 4$, it holds that $2d = 2n/k < n$. Denote the subtrees in the forest $T \setminus \tilde{C}$ by $T_1, T_2, \ldots, T_p$. By Corollary 2.8, $|\tilde{C}| \leq (n/d)+1 = k+1$, and each subtree T_i in $T \setminus \tilde{C}$ has size less than $2d = 2n/k$. Observe that $\sum_{i=1}^{p} |T_i| \;=\; n - |\tilde{C}| \;\geq\; n-k-1$, implying that the number p of subtrees in $T \setminus \tilde{C}$ satisfies

$$p \;\geq\; \frac{n-k-1}{2n/k} \;\geq\; k/4. \tag{3}$$

(The last inequality holds for $k \leq n/2 - 1$.)

To connect the set $\tilde{C}$ of cut vertices, the algorithm first constructs the tree $\tilde{\mathcal{Q}} = \mathcal{Q}(T, \tilde{C})$. Observe that $\tilde{\mathcal{Q}}$ inherits the tree structure of T, that is, for any two points u and v in $\tilde{C}$, u is an ancestor of v in $\tilde{\mathcal{Q}}$ if and only if it is its ancestor in T. Consequently, any 1-spanner path for the tree metric $M_{\tilde{\mathcal{Q}}}$ induced by $\tilde{\mathcal{Q}}$ between two arbitrary comparable[6] points is also a 1-spanner path for the original tree metric M_T. The algorithm proceeds by building a 1-spanner for $\tilde{\mathcal{Q}}$ via one of the aforementioned generalized constructions from [15, 2, 47, 43] (henceforth, *tree-spanner*). In other words, $O(k)$ edges between cut vertices are added to the spanner $\mathcal{H}_k(n)$ to guarantee that the monotone distance in the spanner between any two comparable cut vertices will be $O(\alpha(k))$. Then the algorithm adds to the spanner $\mathcal{H}_k(n)$ edges that connect each of the two sentinels to all other cut vertices. (In fact, the leaf $l(T)$ needs not be connected to all cut vertices, but rather only to those which are its ancestors in T.) Finally, the algorithm calls itself recursively for each of the subtrees $T_1, T_2, \ldots, T_p$ of T. At the bottom level of the recursion, i.e., when $n < 2k+2$, the algorithm uses the tree-spanner to connect all points, and, in addition, it adds to the spanner edges that connect each of the two sentinels $rt(T)$ and $l(T)$ to all the other $n-1$ points.

We denote by $E(n)$ the number of edges in $\mathcal{H}_k(n)$, excluding edges of T. Clearly, $E(n)$ satisfies the recurrence $E(n) \leq O(k) + \sum_{i=1}^{p} E(|T_i|)$, with the base condition $E(q) = O(q)$, for all $q < 2k+2$. Recall that for each $i \in [p]$, $|T_i| \leq 2d = 2n/k < n$, and by Equation (3), we have $p \geq k/4$. Also, since $\widetilde{C}$ is non-empty, it holds that $\sum_{i=1}^{p} |T_i| = n - |\tilde{C}| \leq n-1$. Next, we prove by induction on n that $E(n) \leq 4c(n-1)$, for a sufficiently large constant c. The basis $n < 2k+2$ is immediate. For $n \geq 2k+2$, the induction hypothesis implies that

$$E(n) \;\leq\; c \cdot k + 4c \cdot \sum_{i=1}^{p} (|T_i| - 1) \;=\; c \cdot k - 4c \cdot p + 4c \cdot \sum_{i=1}^{p} |T_i| \;\leq\; c(k-4p) + 4c(n-1) \;\leq\; 4c(n-1).$$

(The last inequality holds as $p \geq k/4$.)

Denote by $\Delta(n)$ the maximum degree of a vertex in $\mathcal{H}_k(n)$, excluding edges of T. Since $|\tilde{C}| \leq k+1$, $\Delta(n)$ satisfies the recurrence $\Delta(n) \leq \max\{k, \Delta(2n/k)\}$, with the base condition $\Delta(q) \leq 2k$, for all $q < 2k+2$. It follows that $\Delta(n) \leq 2k$. Including edges of the tree T, the number of edges increases by at most $n-1$ units and the maximum degree increases by at most $\Delta(T)$ units.

Next, we show that the lightness $\Psi(\mathcal{H}_k(n))$ of the spanner $\mathcal{H}_k(n)$ satisfies $\Psi(\mathcal{H}_k(n)) = O(k \cdot \log_k n)$. To this end, we extend the notion of *load* defined in [1][7] for 1-dimensional spaces to general tree metrics. Consider an edge $e' = (v, w)$ connecting two arbitrary points in M_T, and an edge $e \in E(T)$. The edge e' is said to *load* e if the unique path in T between the endpoints v and w of e' traverses e. For a spanning subgraph H of M_T, the number of edges $e' \in E(H)$ that load an edge $e \in E(T)$ is called the *load* of e by H, and is denoted by $\chi(e) = \chi_H(e)$. The *load* of H (with respect to T), $\chi(H) = \chi_T(H)$, is the maximum load of an edge of T by H. By double-counting,

$$\begin{aligned} w(H) &= \sum_{e' \in E(H)} w(e') \;=\; \sum_{e' \in E(H)} \sum_{\{e \in E(T)\,:\, e \text{ loaded by } e'\}} w(e) \;=\; \sum_{e \in E(T)} \sum_{\{e' \in E(H)\,:\, e' \text{ loads } e\}} w(e) \\ &= \sum_{e \in E(T)} \chi_H(e) \cdot w(e) \;\leq\; \chi(H) \cdot \sum_{e \in E(T)} w(e) \;=\; \chi(H) \cdot w(T), \end{aligned} \tag{4}$$

---

[6] This may not hold true for two points that are not comparable, as their least common ancestor may not belong to $\tilde{\mathcal{Q}}$.

[7] Agarwal et al. [1] used a slightly different notion which they called *covering*. The notion of load as defined above was introduced in [23], but the two notions are very close.

implying that $\Psi(H) = w(H)/w(T) \leq \chi(H)$. Thus it suffices to provide an upper bound of $O(k \cdot \log_k n)$ on the load $\chi(\mathcal{H}_k(n))$ of $\mathcal{H}_k(n)$. Denote by $\chi(n)$ the load of $\mathcal{H}_k(n)$, excluding edges of T. After the first level of recursion, $\mathcal{H}_k(n)$ contains only $O(k)$ edges that connect cut vertices. These edges contribute $O(k)$ units of load to each edge of T. In particular, after the first level of recursion, each subtree in the forest $T \setminus \tilde{C}$ is loaded by at most $O(k)$ edges. Hence $\chi(n)$ satisfies the recurrence $\chi(n) \leq O(k) + \chi(2n/k)$, with the base condition $\chi(q) = O(q)$, for all $q < 2k+2$, yielding $\chi(n) = O(k \cdot \log_k n)$. Including edges of the tree T, the load increases by one unit, and we are done.

Next, we show that $\bar{\Lambda}(n) = \bar{\Lambda}(\mathcal{H}_k(n)) = O(\log_k n + \alpha(k))$. The *leaf radius* $\hat{R}(n)$ of $\mathcal{H}_k(n)$ is defined as the maximum monotone distance between the left-most vertex $l(T)$ in T and one of its ancestors in T. By Corollary 2.8, similarly to the 1-dimensional case, $\hat{R}(n)$ satisfies the recurrence $\hat{R}(n) \leq 2 + \hat{R}(2n/k)$, with the base condition $\hat{R}(q) = 1$, for all $q < 2k+2$. Hence, $\hat{R}(n) = O(\log_k n)$. Similarly, we define the *root radius* $\check{R}(n)$ as the maximum monotone distance between the root $rt(T)$ of T and some other point in T. By the same argument we get $\check{R}(n) = O(\log_k n)$. Applying again Corollary 2.8 and reasoning similar to the 1-dimensional case, we get that $\bar{\Lambda}(n) \leq \max\{\bar{\Lambda}(2n/k), O(\alpha(k)) + \check{R}(2n/k) + \hat{R}(2n/k)\}$, with the base condition $\bar{\Lambda}(q) = O(\alpha(q))$, for all $q < 2k+2$. It follows that $\bar{\Lambda}(n) = O(\log_k n + \alpha(k))$.

Finally, we argue that the worst-case running time of the algorithm, denoted $t(n)$, is $O(n \cdot \log_k n)$. The algorithm starts by invoking the decomposition procedure for selecting the set $\tilde{C}$ of cut vertices. As was mentioned above, this step requires $O(n)$ time. Next, the algorithm builds the tree $\tilde{Q}$, which can be carried out in time $O(|\tilde{Q}|) = O(k)$. The algorithm proceeds by building the tree-spanner for $\tilde{Q}$. The tree-spanner of [15, 2, 47, 43] can be built within linear time. Hence, building the tree-spanner for $\tilde{Q}$ requires $O(k)$ time. Next, the algorithm adds to the spanner edges that connect each of the two sentinels to all other cut vertices, which can be carried out within time $O(k)$ as well. Finally, the algorithm calls itself recursively for each of the subtrees $T_1, T_2, \ldots, T_p$ of T, which requires at most $\sum_{i=1}^{p} t(|T_i|)$ time. At the bottom level of the recursion, i.e., when $n < 2k+2$, the algorithm uses the tree-spanner to connect all points, and in addition, it adds to the spanner edges that connect each of the two sentinels of the tree to all the other $n-1$ points. Hence, the running time of the algorithm at the bottom level of the recursion is $O(n)$. It follows that $t(n)$ satisfies the recurrence $t(n) \leq O(n) + \sum_{i=1}^{p} t(|T_i|)$, with the base condition $t(q) = O(q)$, for all $q < 2k+2$. Recall that $k \geq 4$, and for each $i \in [p]$, $|T_i| \leq 2n/k < n$. We conclude that $t(n) = O(n \cdot \log_k n)$.

Theorem 2.9 *For any tree metric M_T and a parameter k, there exists a 1-spanner with maximum degree at most $\Delta(T) + 2k$, diameter $O(\log_k n + \alpha(k))$, lightness $O(k \cdot \log_k n)$, and $O(n)$ edges. The running time of this construction is $O(n \cdot \log_k n)$.*

We remark that the maximum degree $\Delta(\mathcal{H})$ of the spanner $\mathcal{H} = \mathcal{H}_k(n)$ cannot be in general smaller than the maximum degree $\Delta(T)$ of the original tree. Indeed, consider a unit weight star T with edge set $\{(rt, v_1), (rt, v_2), \ldots, (rt, v_{n-1})\}$. Obviously, any spanner $\mathcal{H}$ for M_T with $\Delta(\mathcal{H}) < n-1$ distorts the distance between the root rt and some other vertex.

2.3.3 1-Spanners with High Diameter

In this section we devise a construction $\mathcal{H}'_k(n)$ of 1-spanners for M_T with comparable monotone diameter $\bar{\Lambda}'(n) = \bar{\Lambda}(\mathcal{H}'_k(n))$ in the range $\bar{\Lambda}'(n) = \Omega(\log n)$.

The algorithm starts with constructing the tree $\tilde{\mathcal{Q}} = \mathcal{Q}(T, \tilde{C})$ that spans the set $\tilde{C}$ of cut vertices. All edges of $\tilde{\mathcal{Q}}$ are inserted into $\mathcal{H}'_k(n)$. (This step is analogous to taking the edges $(r_0, r_1), (r_1, r_2), \ldots, (r_{k-1}, r_k)$ in the 1-dimensional construction of Section 2.2.3.) Observe that the depth of $\tilde{\mathcal{Q}}$ is at most k, implying that any two comparable cut vertices are connected via a 1-spanner path in $\tilde{\mathcal{Q}}$ that consists of at most k edges; since $\tilde{\mathcal{Q}}$ inherits the tree structure of T, this path is also a 1-spanner path for the original tree metric M_T. Then the algorithm calls itself recursively for each of the subtrees $T_1, T_2, \ldots, T_p$ of T. At the bottom level of the recursion, i.e., when $n < 2k+2$, the algorithm adds no additional edges to the spanner.

Similarly to Section 2.3.2 it follows that the number of edges in $\mathcal{H}'_k(n)$ is $O(n)$. Next, we analyze the maximum degree of this construction. Denote by $\Delta'(n)$ the maximum degree of a vertex in $\mathcal{H}'_k(n)$, excluding edges of T, and let $\Delta_0 = \Delta(T)$ denote the maximum degree of the original tree T. By the third assertion of Corollary 2.8, $\Delta(\tilde{\mathcal{Q}}) \leq \Delta_0$, and so $\Delta'(n)$ satisfies the recurrence $\Delta'(n) \leq \max\{\Delta_0, \Delta'(2n/k)\}$, with the base condition $\Delta'(q) = 0$, for all $q < 2k+2$, yielding $\Delta'(n) \leq \Delta_0$. It follows that the maximum degree $\Delta(\mathcal{H}'_k(n))$ of $\mathcal{H}'_k(n)$ is at most $2 \cdot \Delta_0 = 2 \cdot \Delta(T)$.

Next, we show that the load $\chi(\mathcal{H}'_k(n))$ of $\mathcal{H}'_k(n)$ is $O(\log_k n)$, which, by (4), implies that $\Psi(\mathcal{H}'_k(n)) = O(\log_k n)$. Denote by $\chi'(n)$ the load of $\mathcal{H}'_k(n)$, excluding edges of T. After the first level of recursion, $\mathcal{H}'_k(n)$ contains just the edges of the tree $\tilde{\mathcal{Q}}$. We argue that each edge $e = (u, v)$ of T is loaded by at most one edge of $\tilde{\mathcal{Q}}$. Indeed, if both u and v are cut vertices, then e is also an edge of $\tilde{\mathcal{Q}}$, and so it is loaded by itself. Otherwise, either u or v (or both of them) belongs to some subtree T_i in the forest $T \setminus \tilde{C}$. In this case, the fourth assertion of Corollary 2.8 implies that e is loaded by at most one edge in $\tilde{\mathcal{Q}}$, namely, the edge connecting the parent of $rt(T_i)$ in T and the left-most child of $l(T_i)$ in T, if exists. In particular, after the first level of recursion, each subtree in the forest $T \setminus \tilde{C}$ is loaded by at most one edge. Hence $\chi'(n)$ satisfies the recurrence $\chi'(n) \leq 1 + \chi'(2n/k)$, with the base condition $\chi'(q) = 0$, for all $q < 2k+2$. It follows that $\chi'(n) = O(\log_k n)$. Including edges of T, the load increases by one unit, and we are done.

By Corollary 2.8, similarly to the 1-dimensional case, the leaf radius $\hat{R}'(n)$ of $\mathcal{H}'_k(n)$ satisfies the recurrence $\hat{R}'(n) \leq k + \hat{R}'(2n/k)$, with the base condition $\hat{R}'(q) \leq q - 1$, for all $q < 2k+2$, yielding $\hat{R}'(n) = O(k \cdot \log_k n)$. Similarly, we get that $\check{R}'(n) = O(k \cdot \log_k n)$. Applying Corollary 2.8 and reasoning similar to the 1-dimensional case, we get that the comparable monotone diameter $\bar{\Lambda}'(n) = \bar{\Lambda}(\mathcal{H}'_k(n))$ of $\mathcal{H}'_k(n)$ satisfies the following recurrence $\bar{\Lambda}'(n) \leq \max\{\bar{\Lambda}'(2n/k), k + \check{R}'(2n/k) + \hat{R}'(2n/k)\}$, with the base condition $\bar{\Lambda}'(q) \leq q - 1$, for all $q < 2k+2$. It follows that $\bar{\Lambda}'(n) = O(k \cdot \log_k n)$.

We remark that $\mathcal{H}'_k(n)$ is a planar graph.

Finally, by employing an argument very similar to the one used in Section 2.3.2, we get that the worst-case running time of the algorithm is $O(n \cdot \log_k n)$.

Theorem 2.10 *For any tree metric M_T and a parameter k, there exists a 1-spanner with maximum degree at most $2 \cdot \Delta(T)$, diameter $O(k \cdot \log_k n)$, lightness $O(\log_k n)$, and $O(n)$ edges. Moreover, this 1-spanner is a planar graph. The running time of this construction is $O(n \cdot \log_k n)$.*

3 Euclidean Spanners

In this section we demonstrate that our 1-spanners for tree metrics can be used for constructing Euclidean spanners and spanners for doubling metrics.

We start with employing the Dumbbell Theorem of [4] in conjunction with our 1-spanners for tree metrics to construct Euclidean spanners.

Theorem 3.1 *("Dumbbell Theorem", Theorem 2 in [4]) Given a set S of n points in $\mathbb{R}^d$ and a parameter $\epsilon > 0$, a forest $\mathcal{D}$ consisting of $O(1)$ rooted binary trees of size $O(n)$ each can be built in time $O(n \cdot \log n)$, having the following properties:*

1. *For each tree in $\mathcal{D}$, there is a 1-1 correspondence between the leaves of this tree and the points of S.*
2. *Each internal vertex in the tree has a unique representative point, which can be selected arbitrarily from the points in any of its descendant leaves.*
3. *Given any two points $u, v \in S$, there is a tree in $\mathcal{D}$, so that the path formed by walking from representative to representative along the unique path in that tree between these vertices, is a $(1+\epsilon)$-spanner path for u and v.*

For each dumbbell tree in $\mathcal{D}$, we use the following representative assignment from [4]. Leaf labels are propagated up the tree. An internal vertex chooses to itself one of the propagated labels and propagates the other one up the tree. Each label is used at most twice, once at a leaf and once at an internal vertex. Any label assignment induces a weight function over the edges of the dumbbell tree in the obvious way. (The weight of an edge is set to be the Euclidean distance between the representatives corresponding to the two endpoints of that edge.) Arya et al. [4] proved that the lightness of dumbbell trees is always $O(\log n)$, regardless of which representative assignment is chosen for the internal vertices.

Next, we describe our construction of Euclidean spanners with diameter in the range $\Omega(\alpha(n)) = \Lambda = O(\log n)$.

We remark that each dumbbell tree has size $O(n)$. For each (weighted) dumbbell tree $DT_i \in \mathcal{D}$, denote by M_i the $O(n)$-point tree metric induced by DT_i. To obtain our construction of $(1+\epsilon)$-spanners with low diameter, we set $k = n^{1/\Lambda}$, and build the 1-spanner construction $\mathcal{H}^i = \mathcal{H}^i_k(O(n))$ that is guaranteed by Theorem 2.9 for each of the tree metrics M_i. Then we translate each $\mathcal{H}^i$ to be a spanning subgraph $\breve{\mathcal{H}}^i$ of S in the obvious way. (Each edge in $\mathcal{H}^i$ is replaced with an edge that connects the representatives corresponding to the endpoints of that edge.) Finally, let $\mathcal{E}_k(n)$ be the spanner obtained from the union of all the graphs $\breve{\mathcal{H}}^i$.

Theorem 2.9 implies that each graph $\breve{\mathcal{H}}_i$ contains only $O(n)$ edges. By the Dumbbell Theorem, $\mathcal{E}_k(n)$ is the union of a constant number of such graphs. Thus the total number of edges in $\mathcal{E}_k(n)$ is $O(n)$.

We proceed by showing that $\mathcal{E}_k(n)$ is a $(1+\epsilon)$-spanner for S with diameter $\Lambda = \Lambda(\mathcal{E}_k(n))$ at most $O(\log_k n + \alpha(k))$. By the Dumbbell Theorem, for any pair u, v of points in S, there is a dumbbell tree DT_i, so that the unique path $P_{u,v}$ connecting u and v in DT_i is a $(1+\epsilon)$-spanner path for them. Theorem 2.9 implies that there is a 1-spanner path P in $\mathcal{H}^i$ between u and v that consists of at most $O(\log_k n + \alpha(k))$ edges. By the triangle inequality, the weight of the corresponding translated path $\breve{P}$ in $\breve{\mathcal{H}}^i$ is no greater than the weight of $P_{u,v}$. Hence, $\breve{P}$ is a $(1+\epsilon)$-spanner path for u and v that consists of at most $O(\log_k n + \alpha(k))$ edges.

Next, we show that the maximum degree $\Delta(\mathcal{E}_k(n))$ of $\mathcal{E}_k(n))$ is $O(k)$. Since each dumbbell tree DT_i is binary, Theorem 2.9 implies that $\Delta(\mathcal{H}^i) = O(k)$. Recall that each label is used at most twice in DT_i, and so $\Delta(\breve{\mathcal{H}}^i) \le 2 \cdot \Delta(\mathcal{H}^i) = O(k)$. The union of $O(1)$ such graphs will also have maximum degree $O(k)$.

We argue that the lightness $\Psi(\mathcal{E}_k(n))$ of $\mathcal{E}_k(n)$ is $O(k \cdot \log_k n \cdot \log n)$. Consider an arbitrary dumbbell tree DT_i. Recall that the lightness of all dumbbell trees is $O(\log n)$, and so $w(DT_i) = O(\log n) \cdot w(MST(S))$. By Theorem 2.9, the weight $w(\mathcal{H}^i)$ of $\mathcal{H}^i$ is at most $O(k \cdot \log_k n) \cdot w(DT_i) = O(k \cdot \log_k n \cdot \log n) \cdot w(MST(S))$. By the triangle inequality, the weight of each edge in $\breve{\mathcal{H}}^i$ is no greater than the corresponding weight in $\mathcal{H}^i$, implying that the weight $w(\breve{\mathcal{H}}_i)$ of the graph $\breve{\mathcal{H}}_i$ satisfies $w(\breve{\mathcal{H}}^i) \le w(\mathcal{H}^i) = O(k \cdot \log_k n \cdot \log n) \cdot w(MST(S))$. The union of $O(1)$ such graphs will also have weight $O(k \cdot \log_k n \cdot \log n) \cdot w(MST(S))$.

Finally, we bound the running time of this construction. By the Dumbbell Theorem, the forest $\mathcal{D}$ of dumbbell trees can be built in $O(n \cdot \log n)$ time. Theorem 2.9 implies that we can compute each of the graphs $\mathcal{H}^i$ in time $O(n \cdot \log_k n) = O(n \cdot \log n)$. Moreover, as each graph $\mathcal{H}^i$ contains only $O(n)$ edges, translating it into a graph $\breve{\mathcal{H}}^i$ as described above can be carried out in $O(n)$ time. Since there is a constant number of such graphs, it follows that the overall time needed to compute our construction $\mathcal{E}_k(n)$ of Euclidean spanners is $O(n \cdot \log n)$.

To obtain our construction of Euclidean spanners for the complementary range $\Lambda = \Omega(\log n)$, we use our 1-spanners for tree metrics from Theorem 2.10 instead of Theorem 2.9.

Corollary 3.2 *For any set S of n points in $\mathbb{R}^d$, any $\epsilon > 0$ and a parameter k, there exists a $(1+\epsilon)$-spanner with maximum degree $O(k)$, diameter $O(\log_k n + \alpha(k))$, lightness $O(k \cdot \log_k n \cdot \log n)$, and $O(n)$ edges. There also exists a $(1+\epsilon)$-spanner with maximum degree $O(1)$, diameter $O(k \cdot \log_k n)$, and lightness $O(\log_k n \cdot \log n)$. Both these constructions can be implemented in time $O(n \cdot \log n)$.*

In Appendix A we show that the lightness of well-separated pair constructions for random point sets in the unit cube is (w.h.p.) $O(1)$. Also, the lightness of well-separated pair constructions provides an asymptotic upper bound on the lightness of dumbbell trees. We derive the following result as a corollary.

Corollary 3.3 *For any set S of n points that are chosen independently and uniformly at random from the unit cube, any $\epsilon > 0$ and a parameter k, there exists a $(1+\epsilon)$-spanner with maximum degree $O(k)$, diameter $O(\log_k n + \alpha(k))$, lightness (w.h.p.) $O(k \cdot \log_k n)$, and $O(n)$ edges. There also exists a $(1+\epsilon)$-spanner with maximum degree $O(1)$, diameter $O(k \cdot \log_k n)$, and lightness (w.h.p.) $O(\log_k n)$. Both these constructions can be implemented in time $O(n \cdot \log n)$.*

Arya et al. [4] devised a well-separated pair construction of $(1+\epsilon)$-spanners with both diameter and lightness at most $O(\log n)$. In addition, Lenhof et al. [35] showed that there exist point sets for which any well-separated pair construction must admit lightness at least $\Omega(\log n)$. While this existential bound holds true in the worst-case scenario, our probabilistic upper bound of $O(1)$ on the lightness of well separated pair constructions for random point sets implies that on average one can do much better.

Corollary 3.4 *For any set S of n points that are chosen independently and uniformly at random from the unit cube, there exists a $(1+\epsilon)$-spanner with diameter $O(\log n)$, lightness (w.h.p.) $O(1)$, and $O(n)$ edges. This construction can be implemented in $O(n \cdot \log n)$ time.*

Chan et al. [12] showed that for any doubling metric (X, δ) there exists a $(1+\epsilon)$-spanner with constant maximum degree. On the way to this result they proved the following lemma, which we employ in conjunction with our 1-spanners for tree metrics to construct spanners for doubling metrics.

Lemma 3.5 (Lemma 3.1 in [12]) *For any doubling metric (X, δ), there exists a collection $\mathcal{T}$ of $m = O(1)$ spanning trees for (X, δ), $\mathcal{T} = \{\tau_1, \tau_2, \ldots, \tau_m\}$, that satisfies the following two properties:*

1. *For each index $i \in [m]$, the maximum degree $\Delta(\tau_i)$ of the tree τ_i is constant, i.e., $\Delta(\tau_i) = O(1)$.*
2. *For each pair of points $x, y \in X$ there exists an index $i \in [m]$, such that $dist_{\tau_i}(x,y) = O(1) \cdot \delta(x,y)$.*

To obtain our spanners for doubling metrics we start with constructing the collection $\mathcal{T} = \{\tau_1, \tau_2, \ldots, \tau_m\}$ of spanning trees with properties listed in Lemma 3.5. Next, we apply Theorem 2.9 with some parameter k to construct a 1-spanner $\mathcal{Z}^i = \mathcal{Z}^i_k(n)$ for the tree metric induced by the ith tree τ_i in $\mathcal{T}$, for each $i \in [m]$. Notice that, in general, edge weights in the graphs $\mathcal{Z}^i$, $i \in [m]$, may be greater than the corresponding metric distances; for each $i \in [m]$, let $\breve{\mathcal{Z}}^i$ be the graph obtained from $\mathcal{Z}^i$, by assigning weight $\delta(x,y)$ to each edge $(x,y) \in \mathcal{Z}^i$. Our spanner $\mathcal{Z}$ is set to be the union of all the graphs $\breve{\mathcal{Z}}_i$, i.e., $\mathcal{Z} = \bigcup_{i=1}^m \breve{\mathcal{Z}}_i$.

By Theorem 2.9, each of the graphs $\breve{\mathcal{Z}}^i$ contains only $O(n)$ edges. Hence, the number of edges in $\mathcal{Z}$ is at most $m \cdot O(n) = O(n)$.

To argue that $\mathcal{Z}$ is an $O(1)$-spanner for (X, δ) consider a pair of points $x, y \in X$. By Lemma 3.5, there exists an index $i \in [m]$, such that $dist_{\tau_i}(x,y) = O(1) \cdot \delta(x,y)$. Since $\mathcal{Z}^i$ is a 1-spanner for the metric induced by τ_i, it follows that $dist_{\mathcal{Z}^i}(x,y) = dist_{\tau_i}(x,y)$. Also, we have $dist_{\breve{\mathcal{Z}}^i}(x,y) \le dist_{\mathcal{Z}^i}(x,y)$. Finally, since $\breve{\mathcal{Z}}^i \subseteq \mathcal{Z}$, we conclude that $dist_{\mathcal{Z}}(x,y) \le dist_{\breve{\mathcal{Z}}^i}(x,y) \le dist_{\mathcal{Z}^i}(x,y) = dist_{\tau_i}(x,y) = O(1) \cdot \delta(x,y)$. Observe also that $\Lambda(\mathcal{Z}^i) = O(\log_k n + \alpha(k))$, and so there is a path between x and y in $\mathcal{Z}^i$ that consists of at most $\Lambda(\mathcal{Z}^i)$ edges and has length at most $dist_{\mathcal{Z}^i}(x,y)$. Consequently, $\Lambda(\mathcal{Z}) = O(\log_k n + \alpha(k))$.

By Theorem 2.9, the maximum degree of each graph $\breve{\mathcal{Z}}^i$ satisfies $\Delta(\breve{\mathcal{Z}}^i) \le \Delta(\tau_i) + 2k$. By Lemma 3.5, for each index $i \in [m]$, $\Delta(\tau_i) = O(1)$. Hence $\Delta(\breve{\mathcal{Z}}^i) = O(k)$. Since $m = O(1)$, it follows that $\Delta(\mathcal{Z}) \le \sum_{i=1}^m \Delta(\breve{\mathcal{Z}}^i) \le m \cdot O(k) = O(k)$, and we are done.

Corollary 3.6 *For any n-point doubling metric (X, δ) and a parameter k, there exists an $O(1)$-spanner with maximum degree $O(k)$, diameter $O(\log_k n + \alpha(k))$, and $O(n)$ edges.*

4 Acknowledgments

We are grateful to Sunil Arya, David Mount and Michiel Smid for helpful discussions.

References

[1] P. K. Agarwal, Y. Wang, and P. Yin. Lower bound for sparse Euclidean spanners. In *Proc. of 16th SODA*, pages 670–671, 2005.

[2] N. Alon and B. Schieber. Optimal preprocessing for answering on-line product queries. *Manuscript*, 1987.

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

[4] S. Arya, G. Das, D. M. Mount, J. S. Salowe, and M. H. M. Smid. Euclidean spanners: short, thin, and lanky. In *Proc. of 27th STOC*, pages 489–498, 1995.

[5] S. Arya, D. M. Mount, N. S. Netanyahu, R. Silverman, and A. Y. Wu. An optimal algorithm for approximate nearest neighbor searching in fixed dimensions. *J. ACM*, 45(6):819–923, 1998.

[6] S. Arya and M. H. M. Smid. Efficient construction of a bounded degree spanner with low weight. *Algorithmica*, 17(1):33–54, 1997.

[7] A. Bhattacharyya, E. Grigorescu, K. Jung, S. Raskhodnikova, and D. P. Woodruff. Transitive-closure spanners. In *Proc. of 20th SODA*, pages 932–941, 2009.

[8] H. L. Bodlaender, G. Tel, and N. Santoro. Trade-offs in non-reversing diameter. *Nord. J. Comput.*, 1(1):111–134, 1994.

[9] P. B. Callahan and S. R. Kosaraju. A decomposition of multi-dimensional point-sets with applications to k-nearest-neighbors and n-body potential fields. In *Proc. of 24th STOC*, pages 546–556, 1992.

[10] P. B. Callahan and S. R. Kosaraju. Faster algorithms for some geometric graph problems in higher dimensions. In *Proc. of 4th SODA*, pages 291–300, 1993.

[11] H. T.-H. Chan and A. Gupta. Small hop-diameter sparse spanners for doubling metrics. In *Proc. of 17th SODA*, pages 70–78, 2006.

[12] H. T.-H. Chan, A. Gupta, B. M. Maggs, and S. Zhou. On hierarchical routing in doubling metrics. In *Proc. of 16th SODA*, pages 762–771, 2005.

[13] T. M. Chan. Well-separated pair decomposition in linear time? *Inf. Process. Lett.*, 107(5):138–141, 2008.

[14] B. Chandra. Constructing sparse spanners for most graphs in higher dimensions. *Inf. Process. Lett.*, 51(6):289–294, 1994.

[15] B. Chazelle. Computing on a free tree via complexity-preserving mappings. *Algorithmica*, 2:337–361, 1987.

[16] B. Chazelle and B. Rosenberg. The complexity of computing partial sums off-line. *Int. J. Comput. Geom. Appl.*, 1:33–45, 1991.

[17] L. P. Chew. There is a planar graph almost as good as the complete graph. In *Proc. of 2nd SOCG*, pages 169–177, 1986.

[18] G. Das, P. J. Heffernan, and G. Narasimhan. Optimally sparse spanners in 3-dimensional euclidean space. In *Proc. of 9th SOCG*, pages 53–62, 1993.

[19] G. Das and D. Joseph. Which triangulations approximate the complete graph? In *Proc. of the International Symp. on Optimal Algorithms, volume 401 of Lecture Notes in Computer Science*, pages 168–192, 1989.

[20] G. Das and G. Narasimhan. A fast algorithm for constructing sparse Euclidean spanners. In *Proc. of 10th SOCG*, pages 132–139, 1994.

[21] G. Das, G. Narasimhan, and J. S. Salowe. A new way to weigh malnourished euclidean graphs. In *Proc. of 6th SODA*, pages 215–222, 1995.

[22] M. de Berg, O. Cheong, M. van Kreveld, and M. Overmars. *Computational Geometry: Algorithms and Applications, third edition*. Springer-Verlag, Heidelberg, 2008.

[23] Y. Dinitz, M. Elkin, and S. Solomon. Low-light trees, and tight lower bounds for Euclidean spanners. *Discrete & Computational Geometry*, 43(4):736–783, 2010.

[24] J. Fischer and S. Har-Peled. Dynamic well-separated pair decomposition made easy. In *Proc. of 17th CCCG*, pages 235–238, 2005.

[25] G. N. Frederickson. A data structure for dynamically maintaining rooted trees. In *Proc. of 4th SODA*, pages 175–184, 1993.

[26] L. Gottlieb and L. Roditty. An optimal dynamic spanner for doubling metric spaces. In *Proc. of 16th ESA*, pages 478–489, 2008.

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

[28] J. Gudmundsson, C. Levcopoulos, G. Narasimhan, and M. H. M. Smid. Approximate distance oracles for geometric graphs. In *Proc. of 13th SODA*, pages 828–837, 2002.

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

[30] J. Gudmundsson, G. Narasimhan, and M. H. M. Smid. Fast pruning of geometric spanners. In *Proc. of 22nd STACS*, pages 508–520, 2005.
[31] S. Har-Peled and M. Mendel. Fast construction of nets in low dimensional metrics, and their applications. In *Proc. of 21st SOCG*, pages 150–158, 2005.
[32] Y. Hassin and D. Peleg. Sparse communication networks and efficient routing in the plane. In *Proc. of 19th PODC*, pages 41–50, 2000.
[33] J. M. Keil. Approximating the complete euclidean graph. In *Proc. of 1st SWAT*, pages 208–213, 1988.
[34] J. M. Keil and C. A. Gutwin. Classes of graphs which approximate the complete euclidean graph. *Discrete & Computational Geometry*, 7:13–28, 1992.
[35] H. P. Lenhof, J. S. Salowe, and D. E. Wrege. New methods to mix shortest-path and minimum spanning trees. manuscript, 1994.
[36] Y. Mansour and D. Peleg. An approximation algorithm for min-cost network design. *DIMACS Series in Discr. Math and TCS*, 53:97–106, 2000.
[37] G. Narasimhan and M. Smid. *Geometric Spanner Networks.* Cambridge University Press, 2007.
[38] D. Peleg and J. D. Ullman. An optimal synchronizer for the hypercube. *SIAM J. Comput.*, 18(4):740–747, 1989.
[39] M. Pătraşcu and E. D. Demaine. Tight bounds for the partial-sums problem. In *Proc. of 15th SODA*, pages 20–29, 2004.
[40] S. Rao and W. D. Smith. Approximating geometrical graphs via "spanners" and "banyans". In *Proc. of 30th STOC*, pages 540–550, 1998.
[41] J. Ruppert and R. Seidel. Approximating the d-dimensional complete Euclidean graph. In *Proc. of 3rd CCCG*, pages 207–210, 1991.
[42] M. H. M. Smid. Private communication.
[43] S. Solomon. An optimal-time construction of sparse euclidean spanners with tiny diameter. In *Proc. of 22st SODA*, pages 820–839, 2011.
[44] R. E. Tarjan. Applications of path compression on balanced trees. *J. ACM*, 26(4):690–715, 1979.
[45] M. Thorup. On shortcutting digraphs. In *Proc. of 18th WG*, pages 205–211, 1992.
[46] M. Thorup. Shortcutting planar digraphs. *Combinatorics, Probability & Computing*, 4:287–315, 1995.
[47] M. Thorup. Parallel shortcutting of rooted trees. *J. Algorithms*, 23(1):139–159, 1997.
[48] A. C. Yao. Space-time tradeoff for answering range queries. In *Proc. of 14th STOC*, pages 128–136, 1982.

Appendix

A Well-Separated Pair Constructions for Random Point Sets

In this appendix we show that for any set $\mathcal{S}$ of points that are chosen independently and uniformly at random from the unit square, the lightness of well-separated pair constructions is (w.h.p.) $O(1)$. Our argument also extends to higher constant dimensions.

The following lemma from [37] provides a lower bound on the weight of $MST(\mathcal{S})$.

Lemma A.1 (Lemma 15.1.6 in [37]) *For a set $\mathcal{S}$ of n points that are chosen independently and uniformly at random from the unit square, there are constants $c > 0$ and $0 < \rho < 1$, such that $Pr(w(MST(\mathcal{S})) < c \cdot \sqrt{n}) \leq \rho^n$.*

The following statement shows that the lightness of well-separated pair constructions for $\mathcal{S}$ is (w.h.p.) $O(1)$.

Proposition A.2 *For* any *set S of n points in the unit square, the weight of well-separated pair constructions is $O(\sqrt{n})$.*

Before we prove Proposition A.2, we provide (Appendix A.1) the relevant background and introduce some notation. The proof of Proposition A.2 appears in Appendix A.2.

Remark: After communicating this result to Michiel Smid, he [42] pointed out the following alternative argument for obtaining this probabilistic bound of $O(1)$ on the lightness of well-separated pair constructions. First, Chandra [14] showed that for random point sets in the unit cube, any edge set that satisfies the gap property has lightness (w.h.p.) $O(1)$. Second, consider the edge set E of the dumbbell trees of [4]. As shown in [4] this set can be partitioned into $E = E' \cup E''$, such that E' satisfies the *gap property* and $w(E'') = O(w(E'))$. Finally, use the observation that the lightness of well-separated pair constructions is asymptotically equal to that of dumbbell trees. On the other hand, our proof employs a simple, self-contained, combinatorial argument for analyzing the lightness of well-separated pair constructions *directly*. Hence we believe that our approach is advantageous, since, in particular, it does not take a detour through the heavy dumbbell trees machinery of [4].

A.1 Background and Notation

In what follows, let $s > 0$ be a real fixed number.

We say that two point sets in the plane A and B are *well-separated* with respect to s if A and B can be enclosed in two circles of radius r, such that the distance between the two circles is at least $s \cdot r$. The number s is called the *separation ratio* of A and B. A *well-separated pair decomposition* (WSPD) for a point set P in the plane with respect to s is a set $\{\{A_1, B_1\}, \{A_2, B_2\}, \ldots, \{A_m, B_m\}\}$ of pairs of nonempty subsets of P, for some integer m, such that: 1) For each $i \in [m]$, A_i and B_i are well-separated with respect to s, 2) For any two distinct points p and q of P, there is exactly one index i in $[m]$, such that either $p \in A_i$ and $q \in B_i$, or $p \in B_i$ and $q \in A_i$.

Next, we describe a well-known algorithm due to Callahan and Kosaraju [9] for computing a WSPD for P with respect to s. The algorithm consists of two phases. In the first phase, we construct a *split tree*, that is, a tree that corresponds to a hierarchical decomposition of P into rectangles of bounded aspect ratio, where rectangles serve as vertices of the tree, each being split into smaller rectangles as long as it contains more than one point of P. Observe that the split tree does not depend on s. In the second phase, we employ the split tree to construct the WSPD itself.

There are many variants of a split tree, and we outline below the fair split tree due to Callahan and Kosaraju [9]. Place a smallest-possible rectangle $R(P)$ about the point set P. The root of the fair split tree is $R(P)$. Choose the longer side of $R(P)$ and divide it into two equal parts, thus splitting $R(P)$ into two smaller rectangles of equal size, R_l and R_r. The left and right subtrees of the root $R(P)$ are the fair split trees that are constructed recursively for the point sets $R_l \cap P$ and $R_r \cap P$, respectively. This recursive process is repeated until a single point remains, in which case the split tree consists of just a single vertex that stores this point. Following Arya et al. [4], we consider a fair split tree in an ideal form, henceforth the *idealized box split tree*. In this tree rectangles are squares, each split recursively into four identical squares of half the side length. In other words, the idealized box split tree is a *quadtree*. (Refer to Chapter 14 of [22] for the definition of quadtree.) While actual constructions will be performed using the fair split tree or other closely related variants (see, e.g., the *compressed quadtrees* of [24] and [13], and the *balanced box-decomposition tree* of [5]), the idealized box split tree provides a clean and elegant way of conceptualizing the fair split tree in all its variants for purposes of analysis.

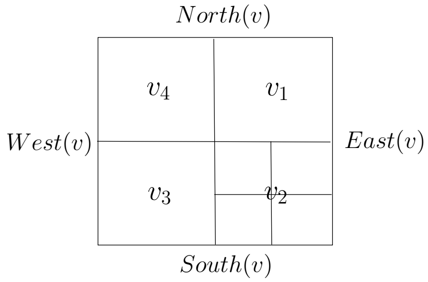

Figure 5: An illustration of a typical internal vertex v in $\mathcal{T}$. The vertex v has four children v_1, v_2, v_3 and v_4, each being a square of half the side length $side(v)/2$. Each child v_i of v, $i \in [4]$, has four children of its own (unless it is a leaf), of side length $side(v)/2^2$ each, and so on. In the illustration only the four children of v_2 are depicted.

Consider the idealized box split tree $\mathcal{T} = \mathcal{T}(P)$ that is constructed for P. We identify each vertex v in the tree T with the square in the plane corresponding to it. For example, the root $rt = rt(\mathcal{T})$ of $\mathcal{T}$ is identified with the smallest-possible square $R(P)$ about the point set P. Thus referring to, e.g., the side length of a vertex v in $\mathcal{T}$, is well-defined. Suppose without loss of generality that the sides of the square rt are parallel to the x and y axes. Consequently, each vertex v of $\mathcal{T}$ is a square whose sides are parallel to the x and y axes. Denote the four sides of v by $North(v)$, $South(v)$, $East(v)$ and $West(v)$, with $North(v)$ and $South(v)$ (respectively, $East(v)$ and $West(v)$) being parallel to the x-axis (resp., y-axis). Denote the four children of an internal vertex v in $\mathcal{T}$ by v_1, v_2, v_3 and v_4, each being a square of half the side length of v, where v_1, v_2, v_3 and v_4 are the North-Eastern, South-Eastern, South-Western and North-Western parts of v, respectively. The side length of v is denoted by $side(v)$. Notice that each child of rt has side length $\frac{side(rt)}{2}$, each grandchild of rt has side length $\frac{side(rt)}{2^2}$, etc. More generally, a vertex v in $\mathcal{T}$ of level[8] $L(v)$, $0 \leq L(v) \leq depth(\mathcal{T})$, has side length $side(v) = \frac{side(rt)}{2^{L(v)}}$. (See Figure 5 for an illustration.) Define $P(v) = v \cap P$. The vertex v is called *empty* if $P(v) = \emptyset$. Otherwise, it is *non-empty*. The depth of a vertex v in $\mathcal{T}$ is defined as the depth of the subtree $\mathcal{T}_v$ of $\mathcal{T}$ rooted at v. For any two vertices u and v in $\mathcal{T}$, we denote by $dist(u, v)$ the distance of closest approach between u and v, i.e., the minimum distance between a point lying on the boundary of u and a point lying on the boundary of v. Also, we denote by $distMax(P(u), P(v))$ the maximum distance between a point in $P(u)$ and a point in $P(v)$. Clearly, $distMax(P(u), P(v))$ is no smaller than $dist(u, v)$. On the other hand, it is bounded from above by the distance of furthest approach between u and v, i.e., the maximum distance between a point lying on the boundary of u and a point lying on the boundary of v, which is, in turn, bounded from above by $dist(u, v) + 2\sqrt{2} \cdot \max\{side(u), side(v)\}$. Thus, $dist(u, v) \leq distMax(P(u), P(v)) \leq dist(u, v) + 2\sqrt{2} \cdot \max\{side(u), side(v)\}$.

To compute the WSPD of P, we use a simple recursive algorithm which consists of the two procedures below. (This algorithm is essentially taken from Callahan and Kosaraju [9].) We initially invoke Procedure 1 below by making the call $WSPD(rt(\mathcal{T}))$, where $rt(\mathcal{T}) = R(P)$. The output returned by this call is the WSPD for P. We omit the proof of correctness, which resembles that of [9]. Notice that for

---

[8] The *level* of a vertex in a rooted tree is defined as its unweighted distance from the root.

any pair of vertices u and v in $\mathcal{T}$, both calls $WSPD(u, v)$ and $WSPD(u)$ return sets of well-separated pairs of P. In what follows we write $WSPD(u, v)$ and $WSPD(u)$ to refer to the sets that are returned by these calls (rather than to the calls themselves).

Procedure 1 $WPSD(u)$:

```
if $|P(u)| \leq 1$ then
    return $\emptyset$
end if
return $\bigcup_{1 \leq i \leq 4} WSPD(u_i) \cup \bigcup_{1 \leq i < j \leq 4} WSPD(u_i, u_j)$
```

Procedure 2 $WPSD(u, v)$:

```
if $P(u) = \emptyset$ or $P(v) = \emptyset$ then
    return $\emptyset$
end if
if $P(u)$ and $P(v)$ are well-separated then
    return $\{\{P(u), P(v)\}\}$
end if
if $side(u) \geq side(v)$ then
    return $\bigcup_{1 \leq i \leq 4} WSPD(u_i, v)$
else
    return $\bigcup_{1 \leq i \leq 4} WSPD(u, v_i)$
end if
```

A *representative assignment* for the split tree $\mathcal{T} = \mathcal{T}(P)$ is a mapping φ between vertices of $\mathcal{T}$ and points of P, sending each vertex v in $\mathcal{T}$ to a point $\varphi(v)$ in $P(v)$. The point $\varphi(v)$ is called the *representative* of v under the mapping φ. We say that a pair (A, B) of nonempty sets of P *belongs* to $\mathcal{T}$, if there are two vertices u and v in $\mathcal{T}$, such that $A = P(u)$ and $B = P(v)$. Given a representative assignment φ, there is a natural correspondence between a well-separated pair $\{P(u), P(v)\}$ that belongs to $\mathcal{T}$ and the edge $(\varphi(u), \varphi(v))$ connecting the representatives of u and v under φ. In the same way, there is a natural correspondence between a set S of well-separated pairs of P that belong to $\mathcal{T}$ and the edge set $E_\varphi(S)$, where $E_\varphi(S) = \{(\varphi(u), \varphi(v)) \mid \{P(u), P(v)\} \in S\}$. The weight $w(H)$ of an edge set H is defined as the sum $\sum_{e=(u,v) \in H} w(u, v)$ of all edge weights in it, where $w(u, v) = \|u - v\|$. Callahan and Kosaraju [10] showed that for any representative assignment φ, the edge set $E^* = E_\varphi(WSPD(P))$ that corresponds to $WSPD(P) = WSPD(rt(\mathcal{T}))$ constitutes a $(1 + \epsilon)$-spanner (with $O(n)$ edges), henceforth the *WSPD-spanner* of P, where ϵ is an arbitrarily small constant depending on s. (It can be easily shown that $\epsilon \leq \frac{8}{s-4}$.)

A.2 Proof of Proposition A.2

In this section we prove Proposition A.2.

Let P be an arbitrary set of n points in the plane, and let $\mathcal{T} = \mathcal{T}(P)$ and $WSPD(P) = WSPD(rt(\mathcal{T}))$ be the idealized box split tree and the WSPD that are constructed for it, respectively. Also, fix an arbitrary representative assignment φ for $\mathcal{T}$. Next, we show that the weight $w(E^*)$ of the WSPD-spanner $E^* = E_\varphi(WSPD(P))$ is at most $c^* \cdot side(rt(\mathcal{T})) \cdot \sqrt{n}$, where c^* is a sufficiently large constant that depends only on s. (We do not try to optimize the constant c^*.) In particular, for a point set P in the unit square we have $side(rt(\mathcal{T})) = 1$, thus proving Proposition A.2.

Observe that for any two vertices u and v in $\mathcal{T}$, both $WSPD(u, v)$ and $WSPD(u)$ are sets of well-separated pairs of P that belong to $\mathcal{T}$. Henceforth, we write $W(u, v)$ and $W(u)$ as shortcuts for $w(E_\varphi(WSPD(u, v)))$ and $w(E_\varphi(WSPD(u)))$, respectively.

Lemma A.3 *Let u and v be two vertices in $\mathcal{T}$, such that $dist(u, v) = c \cdot \max\{side(u), side(v)\}$, for some constant $1/2 \le c \le \sqrt{2}$. Then $W(u, v) \le \alpha \cdot \max\{side(u), side(v)\}$, where $\alpha = \alpha_s$ is a sufficiently large constant that depends only on s.*

Proof: From standard packing arguments, it follows that $|WSPD(u, v)| \le \tilde{\alpha}$, where $\tilde{\alpha} = \tilde{\alpha}_s$ is a sufficiently large constant that depends only on s. For each pair $\{P(x), P(y)\}$ in $WSPD(u, v)$, the weight $\|\varphi(x) - \varphi(y)\|$ of the corresponding edge $(\varphi(x), \varphi(y))$ is at most $dist(u, v) + 2\sqrt{2} \cdot \max\{side(u), side(v)\} = (c + 2\sqrt{2}) \cdot \max\{side(u), side(v)\}$. Define $\alpha = \tilde{\alpha}(c + 2\sqrt{2})$. It follows that $W(u, v) \le \tilde{\alpha}(c + 2\sqrt{2}) \cdot \max\{side(u), side(v)\} = \alpha \cdot \max\{side(u), side(v)\}$. $\square$

Figure 6: a) Two diagonal vertices w and z. b) Two adjacent vertices x and y.

We say that two vertices u and v in $\mathcal{T}$ of the same level are *diagonal* if their boundaries intersect at a single point. (See Figure 6.a for an illustration.) For example, for any vertex v in $\mathcal{T}$, its two children v_1 and v_3 are diagonal. Consider two diagonal vertices u and v in $\mathcal{T}$. Since by definition they are at the same level in $\mathcal{T}$, it holds that $side(u) = side(v)$. Also, notice that $dist(u, v) = 0$ and $\|\varphi(u) - \varphi(v)\| \le distMax(P(u), P(v)) \le 2\sqrt{2} \cdot side(u)$.

Lemma A.4 *For any two diagonal vertices u and v in $\mathcal{T}$, $W(u, v) \le \beta \cdot side(u)$, where $\beta = \beta_s$ is a sufficiently large constant that depends only on s.*

Proof: The proof is by induction on the sum $h = depth(u) + depth(v)$ of depths of u and v.
Basis: $h = 0$. In this case both u and v are leaves, and so each one of them contains at most one point. If either u or v is empty, then $WSPD(u, v)$ is an empty set, and so $W(u, v) = 0 < \beta \cdot side(u)$. Otherwise, $WSPD(u, v) = \{\{P(u), P(v)\}\}$, and so $W(u, v) = \|\varphi(u) - \varphi(v)\| \le 2\sqrt{2} \cdot side(u) < \beta \cdot side(u)$.
Induction Step: We assume the correctness of the statement for all smaller values of h, $h \ge 1$, and prove it for h. If either u or v is empty, then $WSPD(u, v)$ is an empty set, and so $W(u, v) = 0 < \beta \cdot side(u)$. Otherwise, if $P(u)$ and $P(v)$ are well-separated then $WSPD(u, v) = \{\{P(u), P(v)\}\}$, and so $W(u, v) = \|\varphi(u) - \varphi(v)\| \le 2\sqrt{2} \cdot side(u) < \beta \cdot side(u)$. We henceforth assume that $P(u)$ and $P(v)$ are not well-separated. In this case $WSPD(u, v) = \bigcup_{1 \le i \le 4} WSPD(u_i, v)$, and so $W(u, v) = \sum_{1 \le i \le 4} W(u_i, v)$. Since u and v are diagonal, the intersection of v and exactly one child of u consists of a single point, whereas all the other children of u are disjoint from v. Suppose without loss of generality that the child of u that intersects v is u_1. (See Figure 7.a for an illustration.) Observe that $dist(u_2, v) = dist(u_4, v) = \frac{1}{2} \cdot side(v)$ and $dist(u_3, v) = \frac{1}{\sqrt{2}} \cdot side(v)$. Hence, by Lemma A.3, for each $2 \le i \le 4$, $W(u_i, v) \le \alpha \cdot side(v)$. Next, we bound $W(u_1, v)$. If u_1 is empty, then $WSPD(u_1, v)$ is an empty set, and so $W(u_1, v) = 0$. Also, if $P(u_1)$ and $P(v)$ are well-separated, then $WSPD(u_1, v) = \{\{P(u_1), P(v)\}\}$, and so $W(u_1, v) = \|\varphi(u_1) - \varphi(v)\| \le 2\sqrt{2} \cdot side(v)$. Otherwise, $WSPD(u_1, v) = \bigcup_{1 \le i \le 4} WSPD(u_1, v_i)$, and so $W(u_1, v) = \sum_{1 \le i \le 4} W(u_1, v_i)$. Observe that $dist(u_1, v_2) = dist(u_1, v_4) = side(u_1)$ and $dist(u_1, v_1) = \sqrt{2} \cdot side(u_1)$. Hence, by Lemma A.3, for each $i \ne 3$, $W(u_1, v_i) \le \alpha \cdot side(u_1) = \frac{\alpha}{2} \cdot side(u)$. Notice that u_1 and v_3 are diagonal, and so by the induction hypothesis, $W(u_1, v_3) \le \beta \cdot side(u_1) = \frac{\beta}{2} \cdot side(u)$. Set $\beta = 9\alpha$. Altogether,

$$\begin{aligned} W(u, v) &= \sum_{2 \le i \le 4} W(u_i, v) + W(u_1, v) \ \le\ 3\alpha \cdot side(v) + \sum_{1 \le i \le 4, i \ne 3} W(u_1, v_i) + W(u_1, v_3) \\ &\le 3\alpha \cdot side(v) + 3 \cdot \frac{\alpha}{2} \cdot side(u) + \frac{\beta}{2} \cdot side(u) \ =\ side(u) \cdot \left(\frac{9\alpha}{2} + \frac{\beta}{2}\right) \ =\ \beta \cdot side(u). \end{aligned}$$

□

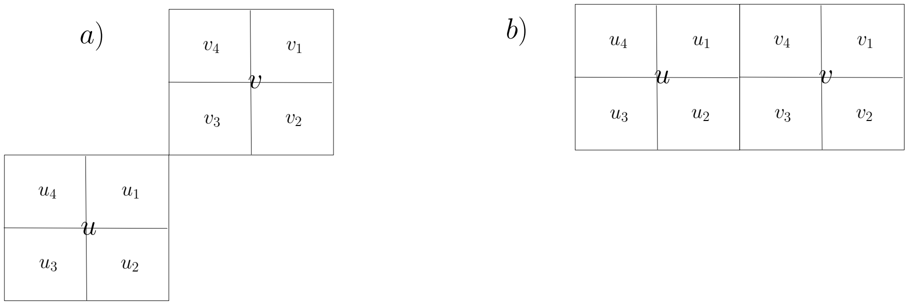

Figure 7: a) An illustration of two diagonal vertices u and v. The vertex u_1 is diagonal to both u_3 and v_3. The vertices u_1 and v_3 intersect at the same point as their respective parents u and v do. b) An illustration of two adjacent vertices u and v. The vertex u_1 is adjacent to u_2, u_4 and v_4. The vertices u_1 and v_4 intersect at a single side, which is the upper half of the side at which their respective parents u and v intersect.

We say that two vertices u and v in $\mathcal{T}$ of the same level are *adjacent* if their boundaries intersect at a single side. (See Figure 6.b for an illustration.) For example, for any vertex v in $\mathcal{T}$, its two children v_1 and v_2 are adjacent. Consider two adjacent vertices u and v in $\mathcal{T}$. Since by definition they are at the same level in $\mathcal{T}$, it holds that $side(u) = side(v)$. Also, notice that $dist(u, v) = 0$ and $\|\varphi(u) - \varphi(v)\| \leq distMax(P(u), P(v)) \leq \sqrt{5} \cdot side(u)$. For a vertex v in $\mathcal{T}$, define $N(v) = |P(v)|$.

Lemma A.5 *For any two adjacent vertices u and v in $\mathcal{T}$ such that $N(u) + N(v) \geq 1$, $W(u, v) \leq \gamma \cdot side(u) \cdot \log(N(u) + N(v))$, where $\gamma = \gamma_s$ is a sufficiently large constant that depends only on s.*

Proof: The proof is by induction on the sum $h = depth(u) + depth(v)$ of depths of u and v.
Basis: $h = 0$. In this case both u and v are leaves, and so each one of them contains at most one point. If either u or v is empty, then $WSPD(u, v)$ is an empty set, and so $W(u, v) = 0 = \gamma \cdot side(u) \cdot \log 1$. Otherwise, $WSPD(u, v) = \{\{P(u), P(v)\}\}$, and so $W(u, v) = \|\varphi(u) - \varphi(v)\| \leq \sqrt{5} \cdot side(u) < \gamma \cdot side(u) \cdot \log 2$.
Induction Step: We assume the correctness of the statement for all smaller values of h, $h \geq 1$, and prove it for h. If either u or v is empty, then $WSPD(u, v)$ is an empty set, and so $W(u, v) = 0 \leq \gamma \cdot side(u) \cdot \log(N(u) + N(v))$. Otherwise, if $P(u)$ and $P(v)$ are well-separated then $WSPD(u, v) = \{\{P(u), P(v)\}\}$, and so $W(u, v) = \|\varphi(u) - \varphi(v)\| \leq \sqrt{5} \cdot side(u) < \gamma \cdot side(u) \cdot \log(N(u) + N(v))$. We henceforth assume that $P(u)$ and $P(v)$ are not well-separated. In this case $WSPD(u, v) = \bigcup_{1 \leq i \leq 4} WSPD(u_i, v)$, and so $W(u, v) = \sum_{1 \leq i \leq 4} W(u_i, v)$. Since u and v are adjacent, exactly two adjacent children u_i and u_{i+1} of u intersect v, $i \in [3]$, each at a single side. Suppose without loss of generality that these children of u are u_1 and u_2. (See Figure 7.b for an illustration.) Observe that $dist(u_3, v) = dist(u_4, v) = \frac{1}{2} \cdot side(v)$. Hence, by Lemma A.3, $W(u_3, v), W(u_4, v) \leq \alpha \cdot side(v)$. Next, we bound $W(u_1, v)$. If u_1 is empty, then $WSPD(u_1, v)$ is an empty set, and so $W(u_1, v) = 0$. Also, if $P(u_1)$ and $P(v)$ are well-separated, then $WSPD(u_1, v) = \{\{P(u_1), P(v)\}\}$, and so $W(u_1, v) = \|\varphi(u_1) - \varphi(v)\| \leq \sqrt{5} \cdot side(v)$. Otherwise, $WSPD(u_1, v) = \bigcup_{1 \leq i \leq 4} WSPD(u_1, v_i)$, and so $W(u_1, v) = \sum_{1 \leq i \leq 4} W(u_1, v_i)$. Observe that $dist(u_1, v_1) = dist(u_1, v_2) = side(u_1)$. Hence, by Lemma A.3, $W(u_1, v_1), W(u_1, v_2) \leq \alpha \cdot side(u_1) = \frac{\alpha}{2} \cdot side(u)$. Notice that u_1 and v_3 are diagonal, whereas u_1 and v_4 are adjacent. Hence, by Lemma A.4, $W(u_1, v_3) \leq \beta \cdot side(u_1) = \frac{\beta}{2} \cdot side(u)$. Recall that u_1 is non-empty, and so $N(u_1) + N(v_4) \geq 1$. By the induction hypothesis, $W(u_1, v_4) \leq \gamma \cdot side(u_1) \cdot \log(N(u_1) + N(v_4)) = \frac{\gamma}{2} \cdot side(u) \cdot \log(N(u_1) + N(v_4))$. We get that

$$\begin{aligned} W(u_1, v) &= \sum_{1 \leq i \leq 4} W(u_1, v_i) \; \leq \; \alpha \cdot side(u) + \frac{\beta}{2} \cdot side(u) + \frac{\gamma}{2} \cdot side(u) \cdot \log(N(u_1) + N(v_4)) \\ &= side(u) \cdot \left(\alpha + \frac{\beta}{2} + \frac{\gamma}{2} \cdot \log(N(u_1) + N(v_4)) \right). \end{aligned}$$

A symmetric argument yields $W(u_2, v) \ \leq \ side(u) \cdot \left(\alpha + \frac{\beta}{2} + \frac{\gamma}{2} \cdot \log(N(u_2) + N(v_3))\right)$.
Observe that $N(u_1) + N(u_2) \leq N(u)$ and $N(v_3) + N(v_4) \leq N(v)$, implying that $2(N(u_1) + N(v_4)) \cdot (N(u_2) + N(v_3)) \leq (N(u) + N(v))^2$. Set $\gamma = 2(4\alpha + \beta)$. Altogether,

$$\begin{aligned} W(u,v) \ &= \ \sum_{1 \leq i \leq 4} W(u_i, v) \ = \ [W(u_1,v) + W(u_2,v)] + [W(u_3,v) + W(u_4,v)] \\ &\leq \ \left[side(u) \cdot \left(2\alpha + \beta + \frac{\gamma}{2} \cdot (\log(N(u_1) + N(v_4)) + \log(N(u_2) + N(v_3)))\right)\right] + [2\alpha \cdot side(v)] \\ &= \ \gamma \cdot side(u) \cdot \left(\frac{4\alpha}{\gamma} + \frac{\beta}{\gamma} + \frac{1}{2} \cdot (\log(N(u_1) + N(v_4)) + \log(N(u_2) + N(v_3)))\right) \\ &= \ \gamma \cdot side(u) \cdot \left(\frac{1 + \log(N(u_1) + N(v_4)) + \log(N(u_2) + N(v_3))}{2}\right) \\ &= \ \gamma \cdot side(u) \cdot \log\sqrt{2(N(u_1) + N(v_4)) \cdot (N(u_2) + N(v_3))} \ \leq \ \gamma \cdot side(u) \cdot \log(N(u) + N(v)). \end{aligned}$$

□

We use the following claim to prove Lemma A.7.

Claim A.6 *For any positive integers $n_1, n_2, \ldots, n_k$, k and n, such that $\sum_{i=1}^{k} n_i = n$,*

$$\sum_{i=1}^{k} \left(\sqrt{n_i} - \frac{\ln n_i}{8}\right) \ \leq \ f(n,k) \ = \ k \cdot \left(\sqrt{n/k} - \frac{\ln(n/k)}{8}\right).$$

Proof: The proof is by induction on k, for $k \in [n]$. The basis $k = 1$ is trivial.
Induction Step: We assume the correctness of the statement for all smaller values of k, $k \geq 2$, and prove it for k. By the induction hypothesis,

$$\sum_{i=1}^{k-1} \left(\sqrt{n_i} - \frac{\ln n_i}{8}\right) \ \leq \ f(n - n_k, k-1) \ = \ (k-1) \cdot \left(\sqrt{\frac{n - n_k}{k-1}} - \frac{\ln\left(\frac{n-n_k}{k-1}\right)}{8}\right).$$

It follows that

$$\sum_{i=1}^{k} \left(\sqrt{n_i} - \frac{\ln n_i}{8}\right) \ \leq \ (k-1) \cdot \left(\sqrt{\frac{n - n_k}{k-1}} - \frac{\ln\left(\frac{n-n_k}{k-1}\right)}{8}\right) + \sqrt{n_k} - \frac{\ln n_k}{8}. \tag{5}$$

Define $g_{n,k}(x) = (k-1) \cdot \left(\sqrt{\frac{n-x}{k-1}} - \frac{\ln\left(\frac{n-x}{k-1}\right)}{8}\right) + \sqrt{x} - \frac{\ln x}{8}$. Since $n_1, n_2, \ldots, n_k \geq 1$ are positive integers and $\sum_{i=1}^{k} n_i = n$, we have that $1 \leq n_k \leq n - k + 1$. Hence, the maximum value of the function $g_{n,k}(x)$ in the range $1 \leq x \leq n - k + 1$ provides an upper bound on the right-hand side of (5). It is easy to verify that the function $g_{n,k}(x)$ in the range $1 \leq x \leq n - k + 1$ is maximized at $x = n/k$. Hence, in the range $1 \leq x \leq n - k + 1$, $g_{n,k}(x) \leq g_{n,k}(n/k) = k \cdot \left(\sqrt{n/k} - \frac{\ln(n/k)}{8}\right)$, and we are done. □

The next lemma implies that $w(E^*) = W(rt) \leq c^* \cdot side(rt) \cdot \left(\sqrt{n} - \frac{\ln(n)}{8}\right) \leq c^* \cdot side(rt) \cdot \sqrt{n}$. Hence, for any set of n points in the unit square, the weight of the WSPD-spanner is $O(\sqrt{n})$, thus proving Proposition A.2.

Lemma A.7 *For any non-empty vertex u in $\mathcal{T}$, $W(u) \leq c^* \cdot side(u) \cdot \left(\sqrt{N(u)} - \frac{\ln(N(u))}{8}\right)$.*

Proof: The proof is by induction on the depth $h = depth(u)$ of u. The basis $h = 0$ is trivial.
Induction Step: We assume the correctness of the statement for all smaller values of h, $h \geq 1$, and prove it for h. First, suppose that $1 \leq N(u) < 20$. In this case, we have $|WSPD(u)| \leq c$, for a sufficiently large constant c. Also, the weight of every edge in the edge set that corresponds to $WSPD(u)$ is at most $\sqrt{2} \cdot side(u)$, and so

$$W(u) \leq c \cdot \sqrt{2} \cdot side(u) < c^* \cdot side(u) \cdot \left(\sqrt{N(u)} - \frac{\ln(N(u))}{8} \right).$$

We henceforth assume that $N(u) \geq 20$. Hence,

$$WSPD(u) = \bigcup_{1 \leq i \leq 4} WSPD(u_i) \cup \bigcup_{1 \leq i < j \leq 4} WSPD(u_i, u_j),$$

and so

$$W(u) = \sum_{1 \leq i \leq 4} W(u_i) + \sum_{1 \leq i < j \leq 4} W(u_i, u_j). \tag{6}$$

To bound $W(u)$, we start with bounding the left sum $\sum_{1 \leq i \leq 4} W(u_i)$ in the right-hand side of (6). Denote by I the set of indices in $[4]$ for which $N(u_i) \geq 1$. By the induction hypothesis, for each index $i \in I$, $W(u_i) \leq c^* \cdot side(u_i) \cdot \left(\sqrt{N(u_i)} - \frac{\ln(N(u_i))}{8} \right)$. Also, for each index $i \in [4] \setminus I$, we have $W(u_i) = 0$. It follows that

$$\begin{aligned} \sum_{1 \leq i \leq 4} W(u_i) &= \sum_{i \in I} W(u_i) \leq \sum_{i \in I} c^* \cdot side(u_i) \cdot \left(\sqrt{N(u_i)} - \frac{\ln(N(u_i))}{8} \right) \\ &= \frac{c^*}{2} \cdot side(u) \cdot \sum_{i \in I} \left(\sqrt{N(u_i)} - \frac{\ln(N(u_i))}{8} \right). \end{aligned} \tag{7}$$

Observe that $\sum_{i \in I} N(u_i) = N(u)$ and $1 \leq |I| \leq 4$. By Claim A.6,

$$\sum_{i \in I} \left(\sqrt{N(u_i)} - \frac{\ln(N(u_i))}{8} \right) \leq f(N(u), |I|) = |I| \cdot \left(\sqrt{N(u)/|I|} - \frac{\ln(N(u)/|I|)}{8} \right).$$

It is easy to verify that the function $f_{N(u)}(x) = f(N(u), x) = x \cdot \left(\sqrt{N(u)/x} - \frac{\ln(N(u)/x)}{8} \right)$ is monotone increasing with x in the range $x > 0$. (The derivative $f'_{N(u)}(x)$ is strictly positive for all $x > 0$.) Since $|I| \leq 4$, we thus have

$$\begin{aligned} \sum_{i \in I} \left(\sqrt{N(u_i)} - \frac{\ln(N(u_i))}{8} \right) &\leq f(N(u), |I|) \leq f(N(u), 4) = 4 \cdot \left(\sqrt{N(u)/4} - \frac{\ln(N(u)/4)}{8} \right) \\ &= 2 \cdot \sqrt{N(u)} - \frac{\ln(N(u)/4)}{2}. \end{aligned} \tag{8}$$

Plugging (8) into (7) yields

$$\sum_{1 \leq i \leq 4} W(u_i) \leq \frac{c^*}{2} \cdot side(u) \cdot \left(2 \cdot \sqrt{N(u)} - \frac{\ln(N(u)/4)}{2} \right) = c^* \cdot side(u) \cdot \left(\sqrt{N(u)} - \frac{\ln(N(u)/4)}{4} \right). \tag{9}$$

We proceed with bounding the right sum $\sum_{1 \leq i < j \leq 4} W(u_i, u_j)$ in the right-hand side of (6). Observe that the two pairs (u_1, u_3) and (u_2, u_4) of children of u are diagonal, whereas the four other pairs $(u_1, u_2), (u_1, u_4), (u_2, u_3)$ and (u_3, u_4) are adjacent. By Lemma A.4, $W(u_1, u_3), W(u_2, u_4) \leq \beta \cdot side(u_1) = \frac{\beta}{2} \cdot side(u)$. Consider a pair (u_i, u_j) among the four pairs of adjacent children of u. If both u_i and u_j are empty, then $W(u_i, u_j) = 0$. Otherwise, we have $N(u_i) + N(u_j) \geq 1$, and so by Lemma A.5,

$$W(u_i, u_j) \leq \gamma \cdot side(u_i) \cdot \log(N(u_i) + N(u_j)) \leq \gamma \cdot side(u_i) \cdot \log(N(u)) = \frac{\gamma}{2} \cdot side(u) \cdot \log(N(u)).$$

Recall that $\gamma = 2(4\alpha + \beta)$, and so $\beta \leq \frac{\gamma}{2}$. Altogether

$$\begin{aligned} \sum_{1 \leq i < j \leq 4} W(u_i, u_j) &\leq \beta \cdot side(u) + 2\gamma \cdot side(u) \cdot \log(N(u)) \\ &\leq \gamma \cdot side(u) \cdot \left(\frac{1}{2} + 2\log(N(u)) \right) \ \leq \ 4\gamma \cdot side(u) \cdot \ln(N(u)). \end{aligned} \tag{10}$$

Plugging (9) and (10) into (6) yields

$$\begin{aligned} W(u) &= \sum_{1 \leq i \leq 4} W(u_i) + \sum_{1 \leq i < j \leq 4} W(u_i, u_j) \\ &\leq c^* \cdot side(u) \cdot \left(\sqrt{N(u)} - \frac{\ln(N(u)/4)}{4} \right) + 4\gamma \cdot side(u) \cdot \ln(N(u)). \end{aligned} \tag{11}$$

It is easy to verify that for a sufficiently large constant c^* and all $n \geq 20$, the right-hand side of (11) is no greater than $c^* \cdot side(u) \cdot \left(\sqrt{N(u)} - \frac{\ln(N(u))}{8} \right)$, and we are done. □