

Fault-Tolerant Geometric Spanners

Artur Czumaj and Hairong Zhao

Department of Computer Science, New Jersey Institute of Technology,
Newark, NJ 07102, USA
{czumaj, hairong}@cis.njit.edu

Abstract. We present two new results about vertex and edge fault-tolerant spanners in Euclidean spaces.

We describe the first construction of vertex and edge fault-tolerant spanners having optimal bounds for maximum degree and total cost. We present a greedy algorithm that for any $t > 1$ and any non-negative integer k , constructs a k -fault-tolerant t -spanner in which every vertex is of degree $\mathcal{O}(k)$ and whose total cost is $\mathcal{O}(k^2)$ times the cost of the minimum spanning tree; these bounds are asymptotically optimal.

Our next contribution is an efficient algorithm for constructing good fault-tolerant spanners. We present a new, sufficient condition for a graph to be a k -fault-tolerant spanner. Using this condition, we design an efficient algorithm that finds fault-tolerant spanners with asymptotically optimal bound for the maximum degree and almost optimal bound for the total cost.

1. Introduction

In this paper we consider a geometric optimization problem of connecting a collection of sites by a “good” network. Problems of this type arise in applications in VLSI design, telecommunication, clustering, robotics, graph theory, and distributed systems. In all these areas it is often important to construct high quality networks. Typical quality measures of good networks include the *cost* of the network, its *stretch factor* (dilation), *connectivity* (resistance to failures), minimum and maximum *degree*, and its *diameter*. These networks and their quality can be modeled by geometric graphs. The sites correspond to points in Euclidean space and the connections can be represented by edges (the straight-line segments) connecting the endpoints.

In this paper we consider various classes of Euclidean graphs. Our main focus is on *spanners* [4], [8], [15], [19] and *fault-tolerant spanners* [20]. Spanners are important geometric structures that provides a sparse or economic representation of a given graph.

They were introduced by Peleg and Schäffer [25] in the context of distributed computing and, later, by Chew [8] in the context of computational geometry. Spanners have many applications in robotics, graph theory, network topology design, distributed systems; the recent $\mathcal{O}(n \log n)$ -time PTAS for Euclidean TSP [27] is heavily based on the use of spanners, and so is the recent PTAS for Euclidean biconnectivity [10]. Spanners are also very extensively used in recent advances on topological issues in ad hoc networks (see, e.g., [16], [17], [26], and the reference therein). Survey expositions [7], [15], [22], [24], and [29] contain an extensive encounter on spanners and their applications.

We first introduce some notations and then give formal definitions of spanners and fault-tolerant spanners. Let V be a set of n points in a Euclidean space \mathbb{R}^d . Let $G = (V, E)$ be a weighted (either *undirected* or *directed*) graph, where V is a vertex set, E is a subset of the (unordered or ordered, respectively) pairs of points in V , and the *length (cost)* of edge (p, q) , denoted by $|pq|$, is equal to the Euclidean distance between points p and q . Any such a graph G will be called a (*Euclidean*) *graph on V* . The *cost of the graph* is the sum of the costs of its edges. Consistent with our definition, the edges of a Euclidean graph $G = (V, E)$ are in one-to-one correspondence with the straight-line segments (in \mathbb{R}^d) connecting the incident vertices.

For a point set V in \mathbb{R}^d , we denote by K_V the *complete Euclidean graph on V* . A graph G on a set of points V is called a (*Euclidean*) *minimum spanning tree (MST)* of V if it is a minimum-cost spanning tree of K_V . For any pair of vertices v and u in a graph, any path between v and u is called a *vu -path*.

Definition 1.1 (Spanners). Let $G = (V, E)$ be a Euclidean graph and let $t \geq 1$. Graph G is called a *t -spanner for V* if for every pair of vertices $v, u \in V$ there is a *vu -path* in G of length upper bounded by $t \cdot |vu|$.

A path in a Euclidean graph G between two vertices v and u is said to be a *t -spanner path* if its length is at most $t \cdot |vu|$. We next define fault-tolerant spanners, which is a class of Euclidean graphs where one requires the existence of short paths between all pairs of vertices even if some vertices or edges of the graph are deleted.

Definition 1.2 (Vertex Fault-Tolerant Spanners [20]). Let V be a set of n points in a metric space, let $t \geq 1$ be a real, and let k be a non-negative integer. A graph $G = (V, E)$ is called a *k -vertex fault-tolerant t -spanner for V* , denoted by (k, t) -VFTS, if for any subset V' of V of size at most k , the graph $G \setminus V'$ is a *t -spanner* for the point set $V \setminus V'$.

Definition 1.3 (Edge Fault-Tolerant Spanners [20]). Let V be a set of n points in a metric space, let $t \geq 1$, and let k be an integer. A graph $G = (V, E)$ is called a *k -edge fault-tolerant t -spanner for V* , denoted by (k, t) -EFTS, if for any subset E' of E of size at most k and for any pair of points p and q in V , the graph $G \setminus E'$ contains a *pq -path* of total length at most t times the length of a shortest path between p and q in the graph $K_V \setminus E'$.

One can show (see [20] and [21]) that a (k, t) -VFTS is also a (k, t) -EFTS. Therefore, from now on we focus our attention mostly on vertex fault-tolerant spanners.

1.1. Previous Works on Spanners and Fault-Tolerant Spanners

Traditionally, the main measures of the quality of spanners are their *number of edges*, *maximum degree*, and the *total cost*. In this context, in any Euclidean space \mathbb{R}^d with constant d , for every positive constant ε , one can construct in time $\mathcal{O}(n \log n)$ a $(1 + \varepsilon)$ -spanner in which every vertex has constant degree and whose total cost is of order of the cost of the MST for the input point set [2], [18]; all these bounds are asymptotically optimal. (See also [1], [3], [7], [12]–[14], and [19] for other related results on spanners.)

The problem of constructing efficient fault-tolerant spanners in Euclidean spaces has been proposed recently by Levcopoulos et al. [20]. They presented in [20] three algorithms constructing k -vertex fault-tolerant spanners. Their first algorithm is based on the observation that the $(k + 1)$ -power of a t -spanner is a (k, t) -VFTS (an s -power of a graph G is a graph with the same vertex set as G and it contains an edge between any pair of vertices that are connected by a path in G with at most s edges). Therefore, if one starts with the spanner construction from [18], then in time $\mathcal{O}(n \log n) + n2^{\mathcal{O}(k)}$ one can obtain a (k, t) -VFTS of maximum degree $2^{\mathcal{O}(k)}$ and the total cost of $2^{\mathcal{O}(k)}$ times the cost of an MST. For a constant k , this construction leads to a fault-tolerant spanner with asymptotically optimal parameters. The other two algorithms described by Levcopoulos et al. [20] use the *well-separated pair decomposition* [6]. It is shown that a k -vertex fault-tolerant spanner can be constructed (i) in $\mathcal{O}(n \log n + k^2 n)$ time with $\mathcal{O}(k^2 n)$ edges, or (ii) in $\mathcal{O}(nk \log n)$ time with $\mathcal{O}(nk \log n)$ edges. Neither the maximum degree nor the total cost of the fault-tolerant spanner is bounded in these two algorithms.

In a follow-up paper, Lukovszki [21] gave a construction of (k, t) -VFTSs with the optimal number of edges $\mathcal{O}(nk)$; the running time of this algorithm is $\mathcal{O}(n \log^{d-1} n + nk \log \log n)$. Lukovszki also presented a construction of (k, t) -VFTSs with maximum degree $\mathcal{O}(k^2)$ and investigated fault-tolerant spanners that allow the use of Steiner points.

1.2. New Contributions

The main open problem left in [20] and [21] is whether there exist fault-tolerant spanners having good bounds for the maximum degree and the total cost. The best bounds obtained in the prior constructions were the k -fault-tolerant spanners having maximum degree $\mathcal{O}(k^2)$ by Lukovszki [21], and one having the total cost $2^{\mathcal{O}(k)}$ times the cost of an MST by Levcopoulos et al. [20].

Our first result resolves that problem and gives a construction of *optimal* (k, t) -VFTSs.

Theorem 1. *Let V be a set of points in \mathbb{R}^d . Let $t > 1$ and let k be a positive integer. Then one can construct a (k, t) -VFTS for V that has maximum degree $\mathcal{O}(k)$ and whose total cost is $\mathcal{O}(k^2 \cdot W_{\text{MST}})$, where W_{MST} denotes the cost of an MST of V . The constants implicit in the \mathcal{O} -notation depend on t and d .*

Notice that by our arguments above this result implies the identical result for k -edge fault-tolerant spanners.

It is not difficult to see that the spanner promised in Theorem 1 has asymptotically *optimal bounds both for the maximum degree and the total cost*. Indeed, since a (k, t) -

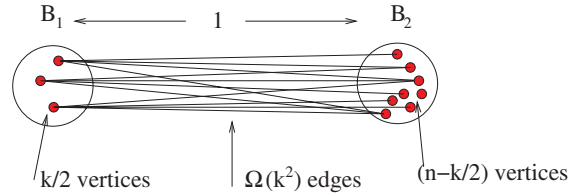


Fig. 1. Any VFTS for points in B_1 and B_2 must have weight at least $\Omega(k^2)$, while the MST has weight $\mathcal{O}(1)$.

VFTS ((k, t) -EFTS) has to be $(k + 1)$ -edge connected, every vertex must have degree $\Omega(k)$. To see that the total cost must be $\Omega(k^2 \cdot W_{\text{MST}})$ in the worst-case, we consider the following construction (see also [20]), see Fig. 1. Suppose that k is even. Let c_1 and c_2 be two points with $|c_1 c_2| = 1$ and let $r \ll 1/n$. We consider n points such that $k/2$ of them are contained in a ball B_1 of radius r centered at c_1 and the remaining $n - k/2$ points are contained in another ball B_2 of radius r with the center at c_2 . Since any MST of these n points has only a single edge between B_1 and B_2 , $W_{\text{MST}} = \mathcal{O}(1)$. However, since the minimum degree of any k -vertex (or k -edge) fault-tolerant spanner is $k + 1$, every vertex in B_1 has to be connected to at least $k/2 + 2$ vertices contained in ball B_2 . Therefore, there are $\Omega(k^2)$ edges between B_1 and B_2 , and the cost of any k -vertex (or k -edge) fault-tolerant spanner is $\Omega(k^2 \cdot W_{\text{MST}})$.

Our construction in Theorem 1 is a generalization of the greedy algorithm (which is our algorithm with $k = 0$) that has been used before to construct spanners [1], [2], [7], [13], [18]. Our main contribution in this context is the first, precise analysis of the fault-tolerant spanners obtained in that construction.

Our next contribution gives an efficient construction of good fault-tolerant spanners. The construction from Theorem 1 gives fault-tolerant spanners having optimal parameters, but it does not lead to an efficient algorithm for constructing the spanners. The following theorem shows that we can construct very efficiently a fault-tolerant spanner whose total cost is just slightly larger than optimal (and the maximum degree remains optimal).

Theorem 2. *Let V be a set of n points in \mathbb{R}^d . Let $t > 1$ and let k be a positive integer. Then, in time $\mathcal{O}(nk \log^d n + nk^2 \log k)$, one can construct a directed (k, t) -VFTS that has maximum degree $\mathcal{O}(k)$ and whose total cost is $\mathcal{O}(k^2 \cdot \log n \cdot W_{\text{MST}})$, where W_{MST} denotes the cost of an MST of V . The constants implicit in the \mathcal{O} -notation depend on t and d .*

Our efficient algorithm from Theorem 2 is based on a new, interesting property of Euclidean graphs that gives a sufficient condition (characterization) for graphs to be (k, t) -VFTSs.

2. Preliminaries

In this paper many algorithms investigated consider pairs of points (edges) in some sequential order. Therefore, we slightly abuse the notation and introduce the total order

on the costs of the edges in E , such that for two edges $(x, x'), (y, y') \in E$, edge (x, x') is *shorter* than edge (y, y') if either $|xx'| < |yy'|$ or $|xx'| = |yy'|$ and edge (x, x') is taken by the algorithm before edge (y, y') .

As we mentioned before, we consider *both directed and undirected Euclidean graphs*. Throughout the paper we abuse notation and denote *both* an undirected and a directed edge from x to y by (x, y) . In the first part of the paper, in Section 3, we analyze *undirected* graphs and undirected fault-tolerant spanners, while in the second part of the paper, in Section 4, for convenience, we consider *directed* graphs and their spanners. Notice, however, that this distinction is really for convenience only, since any undirected spanner can be converted to a directed one by replacing each edge with two directed edges; similarly, any directed spanner can be transformed to be undirected by making every edge undirected and then removing the parallel edges between all pairs of vertices.

2.1. Menger's Theorem and Its Consequences—as Needed in Some Proofs

We use the following lemma that follows easily from Menger's theorem (see Theorem 5 in Chapter III of [5]).

Lemma 2.1. *Let $G = (V, E)$ be an undirected graph. Let $v, u \in V$. Let $X \subseteq (V \setminus \{v, u\})$ be a set of s vertices such that for each $x \in X$,*

- *either $(v, x) \in E$ or there are s internally vertex-disjoint vx -paths in G , and*
- *either $(x, u) \in E$ or there are s internally vertex-disjoint xu -paths in G .*

Then, there are s internally vertex-disjoint vu -paths in G . In particular, if one removes any $s - 1$ vertices in $V \setminus \{v, u\}$ from G , then the obtained graph still contains a vu -path.

One can easily extend the above lemma to obtain the following.

Lemma 2.2. *Let $G = (V, E)$ be an undirected graph. Let $v, u \in V$. Let $E^+ = \{(v_1, u_1), \dots, (v_s, u_s)\}$ be such that*

- *for every i , $1 \leq i \leq s$, either $(v_i, u_i) \in E$ or $v_i = u_i$ and $v_i \neq v, u$,*
- *all u_i and v_i that are neither u nor v are pairwise distinct,*
- *for every i , $1 \leq i \leq s$, either $(v, v_i) \in E$ or there are s internally vertex-disjoint vv_i -paths in G , and*
- *for every i , $1 \leq i \leq s$, either $(u_i, u) \in E$ or there are s internally vertex-disjoint u_iu -paths in G .*

Then there are s internally vertex-disjoint vu -paths in G . In particular, if one removes any $s - 1$ vertices in $V \setminus \{v, u\}$ from G , then the obtained graph still contains a vu -path.

3. k -Vertex Fault-Tolerant Spanners of Low Degree and Low Cost

In this section we analyze the following algorithm and show it constructs (k, t) -VFTSs of both low degree and low cost:

***k*-Greedy Algorithm**

Input: A complete *undirected* Euclidean graph $G = (V, E)$, integer $k \geq 0$,
 real $t > 1$
Output: A (k, t) -VFST $G' = (V, E')$ for V

$E' = \emptyset$
 $G' = (V, E')$
for each edge $(u, v) \in E$ taken in nondecreasing order by length **do**
 if $G' = (V, E')$ does not have $k + 1$ internally vertex-disjoint t -spanner
 uv -paths
 then $E' = E' \cup \{(u, v)\}$
 $G' = (V, E')$
output $G' = (V, E')$

We begin with the following claim.

Claim 3.1. *The k -Greedy Algorithm constructs a (k, t) -VFST.*

Proof. Let V' be any subset of V having size at most k . We prove that $G' \setminus V'$ is a t -spanner for $V \setminus V'$.

We pick any pair of points u, v in $V \setminus V'$. We have to show that $G' \setminus V'$ has a t -spanner uv -path. Clearly, if $(u, v) \in E'$, then this is true. So we suppose that $(u, v) \notin E'$. Then, according to the algorithm, the only reason for not including edge (u, v) in E' is that there are $k + 1$ internally vertex-disjoint t -spanner uv -paths in G' . Therefore, since $|V'| \leq k$, there is at least one such path in $G' \setminus V'$. \square

Remark 3.2. Notice that Claim 3.1 holds even if the edges are taken in an arbitrary order (that is, not necessarily nondecreasing). Furthermore, this claim holds for general, not only for Euclidean graphs and the assumption that G is a complete graph can be weakened. For general graphs, we say a graph $G^* = (V^*, E^*)$ is a (k^*, t^*) -VFST of a weighted graph $G = (V, E)$ if $V^* = V$, $E^* \subseteq E$, and for any $V' \subseteq V$ with $|V'| \leq k^*$, for any pair of vertices $v, u \in V \setminus V'$, the graph $G^* \setminus V'$ contains a vu -path of total length at most t^* times the length of the shortest vu -path in $G \setminus V'$. The proof of Claim 3.1 implies that if we begin the k -Greedy Algorithm with an arbitrary $(k + 1)$ -connected weighted graph $G = (V, E)$, then the obtained graph G' will be a (k, t) -VFST of G .

3.1. Analyzing the Maximum Degree

In this section we prove that the (k, t) -VFST constructed by the k -Greedy Algorithm has maximum degree $\Theta(k)$. Our analysis is in a similar spirit to the analysis of the greedy algorithm for normal spanners, see, e.g., [1] and [7].

We begin with an auxiliary claim.

Claim 3.3. *Let $0 < \theta < \pi/4$ and suppose that t is chosen such that $t \geq 1/(\cos \theta - \sin \theta)$. Let $G' = (V, E')$. Let u, v, x be three points in V with $(u, v), (u, x) \in E'$ and*

$\angle(vux) \leq \theta$. Suppose further that $|uv| \leq |ux|$. Then, for each t -spanner vx -path p in G' , the path consisting of the edge (u, v) followed by the path p is a t -spanner ux -path.

Proof. Let p be any t -spanner vx -path in G' . Let q be the ux -path obtained by taking the edge (u, v) followed by p . Then we have

$$\text{length}(q) = |uv| + \text{length}(p) \leq |uv| + t \cdot |vx|. \quad (1)$$

Furthermore, if we denote by z the point on the segment ux such that $\angle(uzv) = \angle(vzx) = \pi/2$, then we have

$$|vx| \leq |vz| + |zx| = |ux| + (|vz| - |uz|). \quad (2)$$

Next, we observe that $|vz| = |uv| \cdot \sin(\angle(zuv)) \leq |uv| \cdot \sin \theta$ and $|uz| = |uv| \cdot \cos(\angle(zuv)) \geq |uv| \cdot \cos \theta$. Therefore, combining these two identities with (1) and (2), we obtain

$$\begin{aligned} \text{length}(q) &\leq |uv| + t \cdot |vx| \leq |uv| + t \cdot (|ux| + |uv| \cdot (\sin \theta - \cos \theta)) \\ &= t \cdot |ux| + |uv| \cdot (1 - t(\cos \theta - \sin \theta)). \end{aligned}$$

Hence, by the assumption that $t \geq 1/(\cos \theta - \sin \theta)$, we can conclude the claim that $\text{length}(q) \leq t \cdot |ux|$. \square

This claim can be used to prove the following result.

Claim 3.4. *Let $0 < \theta < \pi/4$ and $t \geq 1/(\cos \theta - \sin \theta)$. Let $G' = (V, E')$ be the output of the k -Greedy Algorithm for V . For any $u \in V$, let C_u be any cone in \mathbb{R}^d with the apex at u and the angular diameter¹ at most θ . Then G' has at most $k + 1$ edges incident to u that are contained in the cone C_u .*

Proof. Let E'_{C_u} be the set of edges in G' incident to u that are contained in the cone C_u . Let (u, v) be the longest edge in E'_{C_u} . We prove that if there are more than k edges in E'_{C_u} that are shorter than (u, v) , then there are $k + 1$ t -spanner uv -paths in G' , each using only edges shorter than $|uv|$. This implies that the k -Greedy Algorithm would not add edge (u, v) to E' , which contradicts the fact that (u, v) is an edge of G' .

Suppose there are $k + 1$ edges in E'_{C_u} that are shorter than (u, v) . We prove the existence of $k + 1$ internally vertex-disjoint uv -paths such that each path uses edges in G' that are shorter than (u, v) . We consider first any edge $(u, z) \in E'_{C_u}$ such that $(v, z) \in E'$. By Claim 3.3, the uv -path consisting of the edges (u, z) , (z, v) is of length upper bounded by $t \cdot |uv|$.

Next, we consider any edge $(u, z) \in E'_{C_u}$ such that $(v, z) \notin E'$. Then, since the k -Greedy Algorithm has not taken edge (v, z) to E' , there must exist $k + 1$ internally vertex-disjoint t -spanner vz -paths, and each edge on these paths is shorter than (v, z) ,

¹ The angular diameter of a cone C in \mathbb{R}^d having its apex at point $p \in \mathbb{R}^d$ is defined as the maximum angle between any two vectors $\vec{p}\vec{x}$ and $\vec{p}\vec{y}$, $x, y \in C$.

which is less than (u, v) . Therefore, by Claim 3.3, there exist $k + 1$ t -spanner uv -paths between u and v such that all paths begin with edge (u, z) and then are internally vertex-disjoint.

Summarizing, there are $k + 1$ vertices z_1, z_2, \dots, z_{k+1} such that for each i , $1 \leq i \leq k + 1$, (i) $(u, z_i) \in E'_{C_u}$ and (ii) either $(z_i, v) \in E'$ and $|uz_i| + |z_iv| \leq t \cdot |uv|$, or G' contains $k + 1$ t -spanner uv -paths between u and v such that all paths begin with edge (u, z_i) and then are internally vertex-disjoint, and each edge on each path is shorter than (u, v) . Therefore, we can apply Lemma 2.1 to conclude that G' contains at least $k + 1$ internally vertex-disjoint t -spanner uv -paths, each path using only edges shorter than (u, v) . This, however, contradicts the fact that (u, v) is an edge of G' , and hence, this completes the proof of the claim. \square

In [30] it was shown that there is a constant $c > 0$ such that for any point $p \in \mathbb{R}^d$ and any angle θ , $0 < \theta < \pi$, there is always a collection \mathcal{C} of $\mathcal{O}((c/\theta)^{d-1})$ cones in \mathbb{R}^d having the apex at point p such that (i) $\bigcup_{C \in \mathcal{C}} C = \mathbb{R}^d$, and (ii) each cone $C \in \mathcal{C}$ has angular diameter at most θ . We can incorporate this upper bound for the number of cones in \mathcal{C} with Claims 3.1 and 3.4 to obtain the following lemma.

Lemma 3.5. *Let V be any point set in a Euclidean space \mathbb{R}^d . Let k be any non-negative integer and let t be any real number, $t > 1$. Then the k -Greedy Algorithm returns a (k, t) -VFTS for V having maximum degree $\mathcal{O}((c/\theta)^{d-1} k)$, where $\cos \theta - \sin \theta \geq 1/t$. In particular, if the dimension d and t are constant, then the maximum degree is $\Theta(k)$.*

3.2. Upper Bound for the Cost of Spanners Generated by the k -Greedy Algorithm

The most difficult and challenging part of the proof of Theorem 1 is the analysis of the total cost of the spanner generated by the k -Greedy Algorithm. In order to bound the cost of that spanner, we first introduce the concept of “leapfrog property” [12], which yields a bound for the total cost of a set of edges in terms of the relative position of these edges in Euclidean space.

Definition 3.6 (Leapfrog Property [12]). Let $t \geq 1$. Let $G = (V, E)$ be a Euclidean graph. A set $E^* \subseteq E$ satisfies the t -leapfrog property if, for every $s \geq 2$, for every subset $E^+ = \{(p_1, q_1), \dots, (p_s, q_s)\} \subseteq E^*$ it holds that

$$t \cdot |p_1 q_1| < \sum_{i=2}^s |p_i q_i| + t \cdot \left(|q_s p_1| + \sum_{i=1}^{s-1} |q_i p_{i+1}| \right).$$

The following result is shown in [12] and [14] (see also Theorem 3 of [13]).

Claim 3.7. *Let t be a constant greater than 1. Let $G = (V, E)$ be a Euclidean graph. If a set $E^* \subseteq E$ satisfies the t -leapfrog property, then the total cost of the edges in E^* is $\mathcal{O}(MST_{E^*})$, where MST_{E^*} is the cost of an MST connecting the endpoints of E^* . The constant implicit in the \mathcal{O} -notation depends on t and d .*

The following result gives a tight upper bound for the cost of (k, t) -VFTSs generated by the k -Greedy Algorithm. This is one of the main results of this paper; together with Lemma 3.5, it directly yields Theorem 1.

Lemma 3.8. *Let $G = (V, E)$ be a (k, t) -VFTS of a point set V generated by the k -Greedy Algorithm. Then the cost of G is at most $\mathcal{O}(k^2 \cdot W_{\text{MST}})$. The constant implicit in the \mathcal{O} -notation depends on t and d .*

Proof. The idea of the proof is to partition the edges of G into $\mathcal{O}(k^2)$ groups and then show that the cost of edges in each group is upper bounded by $\mathcal{O}(W_{\text{MST}})$.

We first partition E into disjoint sets E_1, E_2, \dots such that each E_i is a maximal set of edges and any two edges in the set have an angle at most θ . We will require that θ satisfies $t \geq 1/(\cos \theta - \sin \theta)$. By our discussion before (see also [30]), there are $\mathcal{O}((c/\theta)^{d-1})$ such disjoint sets E_1, E_2, \dots .

Next, we partition each E_i into sets E_{i1}, E_{i2}, \dots such that each E_{ij} satisfies the t -leapfrog property. For each edge $e = (v, u) \in E$, let $V_e \subseteq V$ be a minimum vertex set such that after removal of all vertices of V_e from G , there will be no t -spanner vu -paths consisting only of edges shorter than e . Since G is a (k, t) -VFTS generated by the k -Greedy Algorithm, besides e itself, there are at most k internally vertex-disjoint t -spanner vu -paths having all edges shorter than e . Therefore, by Menger's theorem we have $|V_e| \leq k$.

Fix a set E_i . Let S_e be a subset of E_i containing edges that are shorter than e and that are incident to a vertex in V_e . Then we define the sets E_{i1}, E_{i2}, \dots by picking the edges $e \in E_i$ one by one and adding e to any set E_{ij} that does not contain any edge in S_e . We claim that $\mathcal{O}(k^2)$ sets E_{ij} are sufficient in our construction and that each set E_{ij} satisfies the t -leapfrog property. Indeed, by Claim 3.4, each vertex is incident to at most $k + 1$ edges in E_i . Therefore, since $|V_e| \leq k$, we have $|S_e| \leq k(k + 1)$. Thus, $k(k + 1) + 1$ sets E_{ij} are sufficient for each i .

We fix a set E_{ij} . Now we prove that E_{ij} satisfies the t -leapfrog property. Let $E' = \{(u_0, v_0), (u_1, v_1), \dots, (u_m, v_m)\}$ be any subset of E_{ij} . Our goal is to show that

$$t \cdot |u_0 v_0| < \sum_{s=1}^m |u_s v_s| + t \cdot \left(|v_m u_0| + \sum_{s=0}^{m-1} |v_s u_{s+1}| \right).$$

Observe first that this inequality is trivial if either $|u_0 v_0| \leq |v_m u_0|$ or $|u_0 v_0| \leq |v_s u_{s+1}|$ for certain $s, 0 \leq s \leq m - 1$. Furthermore, it is enough to consider the case when (u_0, v_0) is the longest edge in E' . Therefore, from now on, we assume that (u_0, v_0) is the longest edge in $E' \cup \{(v_0, u_1), \{v_1, u_2\}, \dots, \{v_{m-1}, u_m\}, \{v_m, u_0\}\}$.

For convenience, let $e = (u_0, v_0)$. Let G_e be the graph obtained from G after removing all vertices in V_e together with their incident edges and then by removing all edges not shorter than (u_0, v_0) . By our discussion above and by Menger's theorem, any $u_0 v_0$ -path in G_e must have a total length greater than $t \cdot |u_0 v_0|$. Our goal is to show that $\sum_{s=1}^m |u_s v_s| + t \cdot (|v_m u_0| + \sum_{s=0}^{m-1} |v_s u_{s+1}|)$ is at least the length of certain $u_0 v_0$ -path in G_e .

For each $\{x, y\}$ in $\{(v_0, u_1), \{v_1, u_2\}, \dots, \{v_{m-1}, u_m\}, \{v_m, u_0\}\}$, either $(x, y) \in E$ or there are $k + 1$ internally vertex-disjoint t -spanner xy -paths in G consisting only of

edges shorter than (x, y) , which is shorter than (u_0, v_0) by our assumption. Furthermore, none of the points $u_0, u_1, \dots, u_m, v_0, v_1, \dots, v_m$ is in V_e . (Indeed, if, for example, $u_s \in V_e$, then by our assumption no edge incident to u_s should be included in E_{ij} , but this contradicts the fact that $(u_s, v_s) \in E_{ij}$.)

Therefore, by Menger's theorem there must be a t -spanner xy -path in G_e . Hence, we can create a u_0v_0 -path π in G_e as follows: π starts with a t -spanner v_0u_1 -path, then uses edge (u_1, v_1) , then uses a t -spanner v_1u_2 -path, then uses edge $(u_2, v_2), \dots$, then uses edge (u_m, v_m) , and finally terminates with a t -spanner v_mu_0 -path. By our discussion above, π is a u_0v_0 -path in G_e . Moreover, we have

$$\begin{aligned} \text{length}(\pi) &\leq t \cdot |v_0u_1| + |u_1v_1| + t \cdot |v_1u_2| + |u_2v_2| + \dots + |u_mv_m| + t \cdot |v_mu_0| \\ &= \sum_{s=1}^m |u_s v_s| + t \cdot \left(|v_mu_0| + \sum_{s=0}^{m-1} |v_s u_{s+1}| \right). \end{aligned}$$

However, by our arguments above, we know that G_e contains no t -spanner u_0v_0 -path. Therefore,

$$t \cdot |u_0v_0| < \text{length}(\pi) \leq \sum_{s=1}^m |u_s v_s| + t \cdot \left(|v_mu_0| + \sum_{s=0}^{m-1} |v_s u_{s+1}| \right),$$

which implies the t -leapfrog property of set E_{ij} .

Now we can summarize our discussion. We can partition the set of edges of G into $\mathcal{O}((c/\theta)^{d-1} \cdot k^2)$ sets of edges, each set satisfying the t -leapfrog property. Therefore, by Claim 3.7, we can conclude the proof. \square

4. Efficient Construction of Fault-Tolerant Spanners

In the previous section we proved that the k -Greedy Algorithm generates (k, t) -VFTSs with low maximum degree and low total cost. The disadvantage of this algorithm, however, is that we do not know how to implement it efficiently. In this section we discuss an alternative approach to construct fault-tolerant spanners. For convenience, in this section we consider *directed* graphs and their spanners.

We first introduce the notion of *gap property* and *near parallel edges*, and then present a new sufficient property for graphs to be (k, t) -VFTSs. Then we use this characterization to design a simple algorithm that generates good (k, t) -VFTSs in polynomial time. Finally, we tune that algorithm to decrease the running time to $\mathcal{O}((nk \log^d n + nk^2 \log k))$.

4.1. Basic Auxiliary Properties

We first present two important notions on directed edges that are heavily used in the paper.

Definition 4.1 (Near Parallel Edges). Two directed edges (p, q) and (x, y) are called α -near parallel² if after translating vector $\vec{x}\vec{y}$ such that x coincides with p , that is, to the vector $\vec{p}\vec{z}$ with $z = y - (x - p)$, the angle between vectors $\vec{p}\vec{q}$ and $\vec{p}\vec{z}$ is upper bounded by α .

Definition 4.2 (Gap Property). Let $\omega \geq 0$. Let $G = (V, E)$ be a directed Euclidean graph. A set of edges $E^* \subseteq E$ satisfies the ω -gap property if for any two edges $(v_1, u_1), (v_2, u_2) \in E^*$ the distance between the heads and the tails of (v_1, u_1) and (v_2, u_2) is greater than ω times the length of the shorter of the two edges, that is,

$$\min\{|v_1v_2|, |u_1u_2|\} > \omega \cdot \min\{|v_1u_1|, |v_2u_2|\}.$$

The following result is shown in [7]. (The constant implicit in the \mathcal{O} -notation does not depend on d .)

Claim 4.3 [7] (see also Lemma 1 of [3]). *Let $\omega > 0$. Let $G = (V, E)$ be a directed Euclidean graph. If a set $E^* \subseteq E$ satisfies the ω -gap property, then the total cost of the edges in E^* is $\mathcal{O}((1/\omega) \cdot \log |E^*| \cdot MST_{E^*})$, where MST_{E^*} is the cost of an MST connecting the endpoints of the edges in E^* .*

Our next claim states, informally, that if two edges in a directed Euclidean graph are near parallel and close to each other, then one can form a spanner path between the endpoints of the “longer” edge by concatenating the “shorter” edge and the spanner paths between endpoints of the two edges. This claim is essentially proven in Lemma 2 of [3].

Claim 4.4. *Let t, α, ω be real numbers such that $0 < \alpha < \pi/4$, $0 \leq \omega \leq \frac{1}{2}(\cos \alpha - \sin \alpha - 1/t)$. Let G be a directed Euclidean graph. Let (u, v) and (x, z) be two edges in G that are α -near parallel to each other. Suppose $|uv| \leq |xz|/\cos \alpha$, and $|ux| \leq |vz|$. If $|ux| \leq \omega \cdot |uv|$, then (i) $|vz| < |xz|$ and (ii) $t \cdot |xu| + |uv| + t \cdot |vz| \leq t \cdot |xz|$.*

Remark 4.5. We emphasize that Claim 4.4 holds not only when the length of edge (u, v) is less than or equal to the length of edge (x, z) , but it also holds when $|xz| < |uv| \leq |xz|/\cos \alpha$. Furthermore, notice that Claim 4.4 still holds if we change the assumption $|ux| \leq \omega \cdot |uv|$ to $|ux| \leq \omega \cdot \min\{|uv|, |xz|\}$ since the latter assumption is stronger. Therefore, this claim is true when edges (u, v) and (x, z) do not satisfy the ω -gap property. These two observations are important for our efficient algorithm described in Section 4.3.

4.2. Sufficient Conditions for Being a k -Vertex Fault-Tolerant Spanner

In this section we present a new sufficient condition for a Euclidean graph to be a (k, t) -VFTS. Later we show how this condition can be used to obtain an efficient algorithm

² pairs of α -near parallel edges are called “similar directional” in [7].

for constructing (k, t) -VFTSs. Our approach is motivated by a similar characterization of spanners developed by Arya and Smid in [3].

Lemma 4.6. *Let t, α, ω be real numbers such that $0 < \alpha < \pi/4$ and $0 \leq \omega \leq \frac{1}{2}(\cos \alpha - \sin \alpha - 1/t)$. Let $G = (V, E)$ be a directed Euclidean graph. Suppose that for any two vertices u and v in V ,*

1. *either $(u, v) \in E$ or*
2. *there are $k + 1$ edges $\{(u_1, v_1), \dots, (u_{k+1}, v_{k+1})\} \subseteq E$ such that*
 - *for each $1 \leq i \leq k + 1$, $|u_i v_i| \leq |uv|/\cos \alpha$,*
 - *all u_i and v_i that are neither u nor v are pairwise distinct,*
 - *for each $1 \leq i \leq k + 1$, edge (u_i, v_i) is α -near parallel to (u, v) , and*
 - *for each $1 \leq i \leq k + 1$, $\min\{|uu_i|, |v_i v|\} \leq \omega \cdot |u_i v_i|$.*

Then G is a (k, t) -VFTS for V .

We emphasize that in Lemma 4.6 we allow some of the u_i 's and v_i 's to be equal to u or v , but, otherwise, all other endpoints of the edges in $\{(u_1, v_1), \dots, (u_{k+1}, v_{k+1})\}$ must be pairwise distinct.

Proof. In order to prove that G is a (k, t) -VFTS, we show that for any two vertices $u, v \in V$, either $(u, v) \in E$ or G contains $k + 1$ disjoint t -spanner uv -paths, each uv -path having all edges shorter than $|uv|/\cos \alpha$. Our proof is by induction on the rank of the distances between the pairs of points in V .

If $|uv|$ has the minimum distance among all pairs of vertices, then (u, v) must be an edge of E , and hence the claim holds for u, v . Next, we proceed by induction. We consider a pair of vertices $u, v \in V$. By induction, for all ordered pairs of vertices $x, y \in V$ with $|xy| < |uv|$, either $(x, y) \in E$ or G contains $k + 1$ disjoint t -spanner xy -paths, each having all edges shorter than $|xy|/\cos \alpha$. Our goal is to prove that either $(u, v) \in E$ or G contains $k + 1$ disjoint t -spanner uv -paths in G , each having all edges shorter than $|uv|/\cos \alpha$.

We only have to consider the case when $(u, v) \notin E$. Then, by the lemma's assumption, we know that there exist $k + 1$ edges $(u_1, v_1), (u_2, v_2), \dots, (u_{k+1}, v_{k+1})$ in E such that

- (i) for each $1 \leq i \leq k + 1$, edge (u_i, v_i) is shorter than $|uv|/\cos \alpha$,
- (ii) all u_i and v_i that are neither u nor v are pairwise distinct,
- (iii) for each $1 \leq i \leq k + 1$, edges (u_i, v_i) and (u, v) are α -near parallel, and
- (iv) for each $1 \leq i \leq k + 1$, $\min\{|uu_i|, |v_i v|\} \leq \omega \cdot |u_i v_i|$.

We pick any edge (u_i, v_i) and assume, without loss of generality, that $|uu_i| = \min\{|uu_i|, |v_i v|\}$. Since (u_i, v_i) and (u, v) are α -near parallel by (iii), $|u_i v_i| \leq |uv|/\cos \alpha$ by (i), and $|uu_i| \leq \omega \cdot |u_i v_i|$ by (iv), Claim 4.4(i) implies that $|v_i v| < |uv|$. Hence, by induction, either $(v_i, v) \in E$ or there are $k + 1$ disjoint t -spanner $v_i v$ -paths in G , all using only edges shorter than $|v_i v|/\cos \alpha$. Similarly, since $|uu_i| \leq |v_i v| < |uv|$, either $(u, u_i) \in E$ or G contains $k + 1$ disjoint t -spanner uu_i -paths that use only edges shorter than $|uu_i|/\cos \alpha$. Furthermore, by Claim 4.4(ii), each uv -path consisting of a t -spanner uu_i -path (or edge (u, u_i)), edge (u_i, v_i) , and a t -spanner $v_i v$ -path (or edge (v_i, v)) is a t -spanner uv -path.

So far we have proven that there are $k + 1$ edges $(u_1, v_1), (u_2, v_2), \dots, (u_{k+1}, v_{k+1})$ such that for each i , $1 \leq i \leq k + 1$, (i) either $(u, u_i) \in E$ or G contains $k + 1$ disjoint t -spanner uu_i -paths that use only edges shorter than $|uu_i|/\cos \alpha$, and (ii) either $(v_i, v) \in E$ or there are $k + 1$ disjoint t -spanner $v_i v$ -paths in G that use only edges shorter than $|v_i v|/\cos \alpha$. Now, the claim follows directly from Menger's theorem (for more details, see Lemma 2.2). \square

Essentially identical arguments can be used to prove the following characterization for edge fault-tolerant spanners.

Lemma 4.7. *Let t, α, ω be real numbers such that $0 < \alpha < \pi/4$, $0 \leq \omega \leq \frac{1}{2}(\cos \alpha - \sin \alpha - 1/t)$. Let $G = (V, E)$ be a Euclidean graph. Suppose that for any two vertices u and v of V , either $(u, v) \in E$ or there are $k + 1$ edges $\{(u_1, v_1), \dots, (u_{k+1}, v_{k+1})\} \subseteq E$ such that*

- for $1 \leq i \leq k + 1$, $|u_i v_i| \leq |uv|/\cos \alpha$,
- each edge (u_i, v_i) is α -near parallel to (u, v) , and
- for each $1 \leq i \leq k + 1$, $\min\{|uu_i|, |v_i v|\} \leq \omega \cdot |u_i v_i|$.

Then G is a (k, t) -EFTS for V .

The characterization of (k, t) -VFTSs in Lemma 4.6 almost immediately implies a simple polynomial-time algorithm for constructing such spanners, which we call the k -Gap-Greedy Algorithm and describe in the form of a meta-algorithm in Fig. 2.

The following central lemma describes the main properties of the k -Gap-Greedy Algorithm.

Lemma 4.8. *The k -Gap-Greedy Algorithm outputs a directed (k, t) -VFTS for $t = 1/(\cos \alpha - \sin \alpha - 2\omega)$ with maximum in-degree and out-degree $\mathcal{O}(k)$ and whose total*

k -Gap-Greedy Algorithm

Input: A directed complete Euclidean graph $G = (V, E)$, α and ω , and integer k such that $k \geq 0$, $0 < \alpha < \pi/4$, and $0 < \omega < \frac{1}{2}(\cos \alpha - \sin \alpha)$

Output: A (k, t) -VFTS $G' = (V, E')$ for V , where $t = 1/(\cos \alpha - \sin \alpha - 2\omega)$.

$E' = \emptyset$

for each edge $(u, v) \in E$ taken in nondecreasing order by length **do**

Let E^* be a maximal (in the sense of inclusion) subset of E' such that:

1. all edges in E' beginning at u or ending at v are contained in E^* ,
2. for every $(x, y) \in E^*$, (x, y) is α -near parallel to (u, v) ,
3. for every $(x, y) \in E^*$, $\min\{|ux|, |yv|\} \leq \omega \cdot |xy|$,
4. for any pair of distinct edges $(x, y), (z, w) \in E^*$, if $x \neq u$ and $y \neq v$, then $x \neq z$ and $y \neq w$.

if $|E^*| < k + 1$ **then** $E' = E' \cup \{(u, v)\}$

output $G' = (V, E')$

Fig. 2. The k -Gap-Greedy Algorithm.

cost is $\mathcal{O}((1/\omega) \cdot k^2 \log n)$ times the cost of an MST for V . The constant implicit in the \mathcal{O} -notation depends on t and d .

Proof. We first prove that G' is a (k, t) -VFTS by showing that for any ordered pair of vertices u, v , one of the two conditions of Lemma 4.6 is satisfied. If $(u, v) \in E'$, then the first condition is obviously true. Otherwise, $(u, v) \notin E'$ and we consider the iteration of the algorithm when (u, v) is chosen. The only reason that (u, v) is not added to E' is that $|E^*| \geq k + 1$. However, this implies that the second condition of Lemma 4.6 is satisfied for (u, v) . Therefore, G' is a (k, t) -VFTS by Lemma 4.6.

Next, we prove that the maximum out-degree and the maximum in-degree of each vertex is $\mathcal{O}(k)$. Let u be any vertex. We first prove the out-degree of u is $\mathcal{O}(k)$. Let C_u be any cone in \mathbb{R}^d with the apex at u and the angular diameter at most α . Let E'_{C_u} be the set of edges in G' beginning at u that are contained in the cone C_u . We prove that $|E'_{C_u}| \leq k + 1$, which immediately implies that the out-degree of u is $\mathcal{O}(k)$. We analyze the behavior of the algorithm at the moment when $|E'_{C_u}| = k + 1$ and the algorithm considers a new edge (u, v) with $v \in C_u$. We observe that in that case we will have $E'_{C_u} \subseteq E^*$, and, hence, E^* will be of size at least $k + 1$. Therefore, the algorithm will not add the edge (u, v) to the spanner. This implies that $|E'_{C_u}| \leq k + 1$, and, hence, the out-degree of u is $\mathcal{O}(k)$. One can use essentially identical arguments to prove the in-degree of u is $\mathcal{O}(k)$. (Similar arguments show that u is the head/tail of at most $k + 1$ pairwise α -near parallel edges.)

Finally, we prove that G' has small cost. We proceed similarly as in the proof of Lemma 3.8 and first partition E' into disjoint sets E'_1, E'_2, \dots such that each E'_i is a maximal set of edges which are α -near parallel. By our discussion in the proof of Lemma 3.8, there are $\mathcal{O}((c/\alpha)^{d-1})$ such disjoint sets E'_1, E'_2, \dots . We fix one set E'_i and divide the edges in E'_i into a minimal number of groups such that the edges in the same group satisfy the ω -gap property. We prove that $\mathcal{O}(k^2)$ groups are sufficient. It is enough to show that for any edge $e \in E'_i$ there are at most $\mathcal{O}(k^2)$ edges $e' \in E'_i$ shorter than e such that $\{e, e'\}$ does not satisfy the ω -gap property.

We fix an edge $(u, v) \in E'_i$. Let $\overline{V}_{(u,v)}^u = \{p \in V : \exists(p, q) \in E'_i, |pq| \leq |uv|, 0 \leq |up| \leq \omega \cdot |pq|\}$ and let $\overline{E}_{(u,v)}^u$ be the set of edges in E'_i that (i) begin at vertices in $\overline{V}_{(u,v)}^u$, and (ii) are shorter than $|uv|$. Note that since $\overline{E}_{(u,v)}^u \subseteq E'_i$, all edges in $\overline{E}_{(u,v)}^u$ are pairwise α -near parallel. We show that $|\overline{E}_{(u,v)}^u| < 2(k + 1)^2$ by contradiction. We suppose that $|\overline{E}_{(u,v)}^u| \geq 2(k + 1)^2$ and consider the iteration in which the edge (u, v) is picked by the algorithm. Let E^* be the set taken by the k -Gap-Greedy Algorithm when the edge (u, v) is considered. We prove that $|E^*| \geq k + 1$, which contradicts the assumption that $(u, v) \in E'$. Let \mathcal{E} be the set of all edges in E' shorter than (u, v) such that for every edge $e \in \mathcal{E}$, (i) e is α -near parallel to (u, v) and (ii) $\{e, (u, v)\}$ does not satisfy the ω -gap property. Note that because the algorithm picks edges in nondecreasing order, condition 3 of the algorithm implies that none of the edges in E^* satisfies the ω -gap property with (u, v) . Thus, E^* can be defined as the sum of certain *maximal matching* in \mathcal{E} and the set of all edges in \mathcal{E} that either begin at u or end at v . Therefore, to prove that $|E^*| \geq k + 1$ it is sufficient to show that every maximal matching in \mathcal{E} contains at least $k + 1$ edges. Furthermore, because of the well known relation between the cardinality of the maximum matching and the minimum cardinality of a maximal matching, it is

enough to show that the maximum matching in \mathcal{E} contains at least $2(k+1)$ edges. We prove this property by considering the edges in $\overline{E}_{(u,v)}^u$. We first observe that by definition we have $\overline{E}_{(u,v)}^u \subseteq \mathcal{E}$. Therefore, we only must show that $\overline{E}_{(u,v)}^u$ has a matching of size at least $2(k+1)$. Note that there are at most $k+1$ edges in E'_i starting at each vertex as we proved above. Since we assumed that $|\overline{E}_{(u,v)}^u| \geq 2(k+1)^2$, we conclude that set $\overline{E}_{(u,v)}^u$ must contain at least $2(k+1)$ disjoint edges. Therefore, there is a matching of size at least $2(k+1)$ in \mathcal{E} . However, this leads to a contradiction, and hence we proved that $|\overline{E}_{(u,v)}^u| < 2(k+1)^2$.

Symmetrically, we can define $\overline{E}_{(u,v)}^v$, and prove that $|\overline{E}_{(u,v)}^v| < 2(k+1)^2$. Therefore, $|\overline{E}_{(u,v)}^u \cup \overline{E}_{(u,v)}^v| < 4(k+1)^2$. Hence we can conclude that we need at most $4(k+1)^2$ groups of edges from E'_i such that the edges in each group satisfy the ω -gap property. Thus, we proved that we can partition the edges in E' into $\mathcal{O}((c/\alpha)^{d-1} k^2)$ groups such that the edges in each group satisfy the ω -gap property. To conclude the proof we apply Claim 4.3 to obtain an upper bound for the total cost of E' to be $\mathcal{O}((c/\alpha)^{d-1} k^2 (1/\omega) \log n)$ times the cost of an MST of V . \square

4.3. Efficient Construction of a k -Fault-Tolerant Spanner

It is easy to implement the k -Gap-Greedy Algorithm in polynomial time, however, direct implementations lead to a running time of $\Omega(n^2)$. In this section we discuss how that algorithm can be modified to achieve a running time of $\mathcal{O}(nk \log^d n + nk^2 \log k)$ while still returning a spanner having the parameters promised in Lemma 4.8. Our approach is similar to the one developed by Arya and Smid [3] to construct spanners with bounded degree and low cost.

The main idea behind our approach is not to consider all the $\Theta(n^2)$ edges but to have an efficient procedure that will “forbid” certain edges without considering them explicitly during the run of the algorithm. Since every vertex is of degree $\mathcal{O}(k)$, our goal is to ensure that only $\mathcal{O}(k)$ edges incident to any vertex are considered by the algorithm. We relax the rules of inserting an edge in the k -Gap-Greedy Algorithm in order to obtain a more efficient implementation. On one hand, we want to maintain the properties of the output spanner required by Lemma 4.6 and, on the other hand, we aim at outputting a spanner having properties described in Lemma 4.8. Below we describe the main idea behind that relaxation, and the algorithm itself is described in Section 4.3.2.

4.3.1. Main Idea Behind the Modified Algorithm. We use the collection of cones of angular diameter α (see Section 3.1) to test whether two edges are α -near parallel to each other. That is, we assume that we are given a collection \mathcal{C} of cones with apexes at the origin and angular diameter α , so that the cones in \mathcal{C} cover \mathbb{R}^d . Then we say an edge (u, v) is in a cone C if the vector $v - u \in C$. Using this notion, each time we consider a pair of points u, v with $(u, v) \in C$, we only look at those edges $(x, y) \in E'$ that are in C . Since if (u, v) and (x, y) are in the same cone, then they are α -near parallel to each other, this notion allows us to relax the testing of α -near parallel edges.

Once we fix the collection \mathcal{C} of cones, we consider all cones from \mathcal{C} separately. We build the spanner by defining the edge set E' which will be the union of the edge sets

constructed for each cone separately. We use E'_C to denote all those edges of E' inside the cone $C \in \mathcal{C}$.

In the k -Gap-Greedy Algorithm, one analyzes the edges in order of their increasing lengths. This is more complicated (in the sense of efficiency) if we want to consider the edges in different cones separately. Therefore, following the approach from [28] (see also [3]), we approximate the distance between the points in a cone. For each cone C , we consider any fixed ray ℓ_C incident to the apex of C and that is included in the cone C . Then the idea of approximating the distance between pairs of points is to use the distance between the projections of the points on the ray ℓ_C . To make this notion more formal, we first need a few definitions. For any cone $C \in \mathcal{C}$ and any point $x \in \mathbb{R}^d$, let C_x be the translation of C so that the apex of C_x is at x , that is, $C_x = \{y \in \mathbb{R}^d : y - x \in C\}$. Similarly, for the ray ℓ_C , let $\ell_{C,x} = \{y \in \mathbb{R}^d : y - x \in \ell_C\}$. Then we approximate the distance between the points in the cone C using the following function dist_C :

$$\text{dist}_C(x, y) = \begin{cases} |x R_y| & \text{if } y \in C_x \text{ and } R_y \text{ is the orthogonal projection of } y \text{ onto } \ell_{C,x}, \\ \infty & \text{if } y \notin C_x. \end{cases}$$

The main reason for using this approximation of the distances between pairs of points inside a cone is that it can be efficiently maintained by a dynamic algorithm (see also [3] and [28]). Furthermore, it is easy to show that if $\text{dist}_C(x, y) \leq \text{dist}_C(u, v)$, then $|xy| \leq |uv|/\cos \alpha$. Therefore, by Claim 4.4 and Lemma 4.6, the algorithm remains correct (in the sense that it satisfies the properties from Lemma 4.6) even if edge (x, y) is considered before edge (u, v) .

Summarizing, our algorithm considers the edges inside each cone in nondecreasing order of their dist_C length.

Consider an edge (x, y) that is to be taken by the algorithm and let (x, y) be in a cone $C \in \mathcal{C}$. Observe that if there are already at least $k + 1$ edges in C with the head at x , then we know that the edge (x, y) will not be taken by the algorithm. Therefore, whenever we have a vertex x already having $k + 1$ outgoing edges in C , then we do not have to consider any further edge starting at x in C . Symmetrically, we only consider at most $k + 1$ incoming edges for any point y .

The other reason for rejecting edge (x, y) is that there are many disjoint edges in C which are shorter than (x, y) with respect to the dist_C length and are very close to either x or y . Let $x \in V$ be any input point. We keep two special data structures to maintain a maximal set of disjoint edges in C that have starting or ending points close to x , respectively. We modify these two data structures not when an edge in C incident to x is considered, but, instead, they are updated each time an edge close to x is inserted into E'_C . To be more precise, at each moment of the algorithm, we maintain a set $F_C(x)$ for each $x \in V$ which contains a set of disjoint edges in E'_C such that for any $(u, v) \in F_C(x)$ we have $|xu| \leq \omega \cdot |uv|$. Similarly, we maintain $H_C(x)$ which contains a set of disjoint edges in E'_C such that for any $(u, v) \in H_C(x)$ we have $|vx| \leq \omega \cdot |uv|$.

Now, suppose an edge (u, v) is inserted to E'_C . Let $N_{(u,v)}^u$ and $N_{(u,v)}^v$ be the set of points that are at distance at most $\omega \cdot |uv|$ from u and v , respectively. By our definitions of $F_C(x)$ and $H_C(x)$, we need to update those points in $N_{(u,v)}^u$ and $N_{(u,v)}^v$ that are ‘‘affected’’ by the edge (u, v) . That is, for every point $x \in N_{(u,v)}^u$, if $F_C(x)$ contains no edge incident

to either u or v , then we can update $F_C(x)$ by inserting (u, v) to it. Similarly, we update $H_C(y)$ for every point $y \in N_{(u,v)}^v$.

Notice that the operation of updating the sets $F_C(x)$ and $H_C(x)$ may be very expensive, because both the vertex sets $N_{(u,v)}^u, N_{(u,v)}^v$ and the edge sets $F_C(x), H_C(x)$ may be very large. Therefore, we relax the definition of the sets $F_C(x)$ and $H_C(x)$ by observing that once $F_C(x)$ or $H_C(x)$ is of size larger than or equal to $k+1$, by Lemma 4.6, we know that no new edge in C beginning or ending at x , respectively, will have to be added to E'_C . Therefore, we maintain two sets of points from V : V_C^1 containing all points that can still be the heads of new edges in E'_C and V_C^2 containing points that can still be the tails of new edges in E'_C . Each time the size of a certain $F_C(x)$ is greater than or equal to $k+1$, we delete x from V_C^1 ; we do not consider any further edges in C that begin at x , we do not update $F_C(x)$ anymore, nor do we consider x in any further sets $N_{(u,v)}^u$. Similarly, if $H_C(x) \geq k+1$, we delete x from V_C^2 , we do not consider any further edges in C ending at x , we do not update $H_C(x)$ or consider x in any further sets $N_{(u,v)}^v$. In this way, since both $F_C(x)$ and $H_C(x)$ have a size between 0 and $k+1$, the total number of operations for updating the sets $F_C(x)$ and $H_C(x)$ in the entire algorithm (for a given cone C) will be $\mathcal{O}(kn)$.

Observe that once a vertex x has been reported $2(k+1)^2$ times in the sets $N_{(u,v)}^u$, then we know that there are $2(k+1)^2$ edges (u, v) such that $|ux| \leq \omega \cdot |uv|$. Since each vertex is the head of at most $k+1$ edges in C , there must be $2(k+1)$ disjoint edges among all these edges. Therefore, in this case, the size of $F_C(x)$ must be greater than or equal to $k+1$ (see Lemma 4.8 for a more detailed discussion on similar arguments). Hence, at this moment x will be deleted and will not be reported in $N_{(u,v)}^u$ for any $u \in V$ any more. For the same reason x will be reported in $N_{(u,v)}^v$ at most $2(k+1)^2$ times. This allows us to conclude that the total size of $N_{(u,v)}^u$ and $N_{(u,v)}^v$ for all vertices u and v in the entire algorithm (for a given cone C) is $4n(k+1)^2$.

To define sets $N_{(u,v)}^u$ we need to find all points that are at a distance at most $\omega \cdot |uv|$ from u . However, since it is difficult to maintain dynamic data structures to find $N_{(u,v)}^u$ in the Euclidean metric, in the definition of $N_{(u,v)}^u$ we assume the distances between points according to the L_∞ metric. That is, we *redefine* $N_{(u,v)}^u$ to be the set of all points x with $|ux|_\infty \leq (\omega/\sqrt{d})|uv|$. (Here, we need to output only the points that have not been deleted, that is, those x for which $|F_C(x)| < k+1$.) Since $|ux|_\infty \leq (\omega/\sqrt{d})|uv|$ implies that $|ux| \leq \omega \cdot |uv|$, the new definition of $N_{(u,v)}^u$ gives a subset of the set $N_{(u,v)}^u$ defined above. Set $N_{(u,v)}^v$ is redefined symmetrically.

4.3.2. Improved k -Gap-Greedy Algorithm. In the previous subsection we presented the main ideas behind our modifications of the k -Gap-Greedy Algorithm that allow us to design a new algorithm that finds good fault-tolerant spanners efficiently. A more formal description of our algorithm is presented on p. 224. We begin with a formal proof of its correctness and then a discussion about its efficient implementation.

Properties of the Improved k -Gap-Greedy Algorithm. This section is devoted to a formal proof of the following lemma.

Lemma 4.9. *Let V be a set of n points in \mathbb{R}^d . Let α, ω be real numbers such that $0 < \alpha < \pi/4$ and $0 < \omega < \frac{1}{2}(\cos \alpha - \sin \alpha)$. Let $t = 1/(\cos \alpha - \sin \alpha - 2\omega)$.*

Improved k -Gap-Greedy Algorithm

Input: Set V of n points in \mathbb{R}^d , an integer $k \geq 0$, and positive α, ω with $\alpha < \pi/4$ and $\omega < \frac{1}{2}(\cos \alpha - \sin \alpha)$

Output: A (k, t) -VFTS $G' = (V, E')$ for V , where $t = 1/(\cos \alpha - \sin \alpha - 2\omega)$.

let \mathcal{C} be any collection of cones with angular diameter α that cover \mathbb{R}^d

for each cone $C \in \mathcal{C}$ **do**

$E_C = \{(u, v) : \text{dist}_C(u, v) < \infty\}$ $\{E_C$ contains all edges (u, v) such that $v - u \in C\}$

$E'_C = \emptyset$ $\{E'_C$ collects the edges in E_C that are to be included in the spanner}

for each $x \in V$ **do** $F_C(x) = \emptyset$ $\{F_C(x)$ is a set of disjoint edges in E'_C that begin at points near to $x\}$

for each $x \in V$ **do** $H_C(x) = \emptyset$ $\{H_C(x)$ is a set of disjoint edges in E'_C that end at points near to $x\}$

$V_C^1 = V$ $\{V_C^1$ contains all points u in V such that $|F_C(x)| < k + 1\}$

$V_C^2 = V$ $\{V_C^2$ contains all points v in V such that $|H_C(x)| < k + 1\}$

while $E_C \neq \emptyset$ **do**

let $(u, v) \in E_C$ be such that $\text{dist}_C(u, v)$ is minimal

if the number of edges beginning at u in E'_C is at least $k + 1$ **then**

delete u from V_C^1 and delete from E_C all edges beginning at u

if the number of edges ending at v in E'_C is at least $k + 1$ **then**

delete v from V_C^2 and delete from E_C all edges ending at v

if (u, v) is still in E_C **then** $\{\text{edge } (u, v) \text{ will be in the spanner}\}$

$E_C = E_C \setminus \{(u, v)\}$

$E'_C = E'_C \cup \{(u, v)\}$

let $N_{(u,v)}^u$ be the set of points $x \in V_C^1$ with $|ux|_\infty \leq (\omega/\sqrt{d})|uv|$

let $N_{(u,v)}^v$ be the set of points $y \in V_C^2$ with $|vy|_\infty \leq (\omega/\sqrt{d})|uv|$

for each $x \in N_{(u,v)}^u$ **do**

if (u, v) is disjoint with all edges in $F_C(x)$ **then**

$F_C(x) = F_C(x) \cup \{(u, v)\}$

if $|F_C(x)| \geq k + 1$ **then**

delete x from V_C^1 and delete from E_C all edges beginning at x

for each $y \in N_{(u,v)}^v$ **do**

if (u, v) is disjoint with all edges in $H_C(y)$ **then**

$H_C(y) = H_C(y) \cup \{(u, v)\}$

if $|H_C(y)| \geq k + 1$ **then**

delete y from V_C^2 and delete from E_C all edges ending at y

$E' = \bigcup_{C \in \mathcal{C}} E'_C$

output $G' = (V, E')$

There is a constant c such that the Improved k -Gap-Greedy Algorithm outputs a directed (k, t) -VFST with maximum in-degree and out-degree $\mathcal{O}(k)$ and whose total cost is $\mathcal{O}((c/\alpha)^{d-1} (\sqrt{d}/\omega) k^2 \log n)$ times the cost of an MST for V .

Proof. We need to prove that the output of the Improved k -Gap-Greedy Algorithm has the same properties as the output of the k -Gap-Greedy Algorithm after we relax some conditions. The proof is similar to that of Lemma 4.8.

Let G' be the output of the Improved k -Gap-Greedy Algorithm. First, we prove that G' is a (k, t) -VFST by showing that G' satisfies the conditions of Lemma 4.6. Let u, v be any ordered pair of vertices, and $v - u \in C$ for some $C \in \mathcal{C}$. If $(u, v) \in E'_C$, then the first condition is obviously true. Otherwise, $(u, v) \notin E'_C$ and there are two possible reasons why (u, v) was not inserted to E'_C :

- (i) (u, v) is considered explicitly in the “while” loop but is not inserted to E'_C .
In this case $\text{dist}_C(u, v)$ is minimum at the beginning of a certain iteration. Since (u, v) is not added to E'_C , there must exist either $k + 1$ edges with heads at u or $k + 1$ edges with tails at u which are already inserted to E'_C . These edges have length at most $|uv|/\cos \alpha$, are α -near parallel to (u, v) , and they have one common endpoint with (u, v) . Therefore, the second condition of Lemma 4.6 is satisfied.
- (ii) (u, v) is deleted when some other pair (x, y) is picked and added to E'_C in the “while” loop.

In this case the only reason that (u, v) is deleted from E_C is either $|F_C(u)| = k + 1$ or $|H_C(v)| = k + 1$ after (x, y) is inserted to $F_C(u)$ or $H_C(v)$, respectively. Suppose that $|F_C(u)| = k + 1$ and consider the edges in $F_C(u) = \{(x_1, y_1), \dots, (x_{k+1}, y_{k+1})\}$. By the Improved k -Gap-Greedy Algorithm, edge (x_i, y_i) , $1 \leq i \leq k + 1$, can be inserted to $F_C(u)$ only if: (a) $|ux_i|_\infty \leq (\omega/\sqrt{d}) |x_i y_i|$, which implies that $|ux_i| \leq \omega \cdot |x_i y_i|$, and (b) (x_i, y_i) is disjoint with all other edges in $F_C(u)$. Furthermore, since (x_i, y_i) is considered before (u, v) by the algorithm, we must have $\text{dist}_C(x_i, y_i) \leq \text{dist}_C(u, v)$, which means $|x_i y_i| \leq |uv|/\cos \alpha$. Therefore, we can combine this with (a) and (b) to see that the second condition of Lemma 4.6 is satisfied.

To summarize, we proved, for each pair (x, y) , either (x, y) is an edge in G' or there are $k + 1$ edges in G' satisfying the second condition of Lemma 4.6. Therefore, G' is a (k, t) -VFST.

Next, we prove that the maximum in-degree and out-degree of each vertex is $\mathcal{O}(k)$. The proof is basically the same as the proof of Lemma 4.8. Let u be any vertex. According to the Improved k -Gap-Greedy Algorithm it is easy to see that at most $k + 1$ edges beginning at u can be added to the E'_C for any given cone $C \in \mathcal{C}$. Therefore the out-degree of u in G' is $\mathcal{O}(k)$. Similarly, we can show that the in-degree is $\mathcal{O}(k)$.

Finally, we prove that G' has small cost. We fix a cone C and divide the edges in E'_C into a minimal number of groups such that the edges in the same group satisfy the (ω/\sqrt{d}) -gap property. We prove that $\mathcal{O}(k^2)$ groups suffice. To prove this claim it is enough to show that for any edge $e \in E'_C$ there are at most $\mathcal{O}(k^2)$ edges $e' \in E'_C$ that are “shorter” than e in terms of the dist_C length and such that $\{e, e'\}$ does not satisfy the (ω/\sqrt{d}) -gap property.

Fix an edge $(u, v) \in E'_C$. Let $E_{(u,v)}^u = \{(x, y) \in E'_C : \text{dist}_C(x, y) < \text{dist}_C(u, v) \text{ and } |ux| \leq \omega/\sqrt{d} \cdot |xy|\}$ and $E_{(u,v)}^v = \{(x, y) \in E'_C : \text{dist}_C(x, y) < \text{dist}_C(u, v) \text{ and } |vy| \leq (\omega/\sqrt{d}) \cdot |xy|\}$. Observe that if an edge (x, y) is “shorter” than (u, v) with respect to the dist_C length and it fails to satisfy the (ω/\sqrt{d}) -gap property with (u, v) , then $(x, y) \in E_{(u,v)}^u \cup E_{(u,v)}^v$. Therefore, to prove the claim it is sufficient to show that $|E_{(u,v)}^u \cup E_{(u,v)}^v| = \mathcal{O}(k^2)$.

We first prove that $|E_{(u,v)}^u| = \mathcal{O}(k^2)$. Let $\mathcal{E}_{(u,v)}^u = \{(x, y) \in E'_C : \text{dist}_C(x, y) < \text{dist}_C(u, v) \text{ and } |ux|_\infty \leq (\omega/\sqrt{d}) \cdot |xy|\}$. Note that since $|ux|_\infty \leq |ux|$, we have $E_{(u,v)}^u \subseteq \mathcal{E}_{(u,v)}^u$. In the following we show that $|\mathcal{E}_{(u,v)}^u| = \mathcal{O}(k^2)$ which directly implies that $|E_{(u,v)}^u| = \mathcal{O}(k^2)$.

We have two observations. First, because all edges in $\mathcal{E}_{(u,v)}^u$ are “shorter” than (u, v) with respect to dist_C , all these edges are picked and inserted to E'_C by the algorithm before (u, v) . Secondly, each time after one of these edges (x, y) is inserted to E'_C , the algorithm inserts (x, y) to $F_C(u)$ if (x, y) is disjoint with edges already in $F_C(u)$. In other words, at the end of each iteration of the Improved k -Gap-Greedy Algorithm $F_C(u)$ keeps a maximal matching of the edges in $\mathcal{E}_{(u,v)}^u$ that have been inserted to E'_C so far.

Now we prove by contradiction that $|\mathcal{E}_{(u,v)}^u| < 2(k+1)^2$. Suppose that $|\mathcal{E}_{(u,v)}^u| \geq 2(k+1)^2$. We have already shown that each vertex has out-degree (in-degree) in E'_C of at most $k+1$. Therefore, the maximum matching of $\mathcal{E}_{(u,v)}^u$ must contain at least $2(k+1)$ edges and thus any maximal matching of $\mathcal{E}_{(u,v)}^u$ must have at least $k+1$ edges. Since $F_C(u)$ is a maximal matching of $\mathcal{E}_{(u,v)}^u$, we have $|F_C(u)| \geq k+1$. Then there must exist an edge $(x, y) \in \mathcal{E}_{(u,v)}^u$ such that during the iteration when (x, y) is inserted to E'_C , (x, y) is also inserted to $F_C(u)$, and $|F_C(u)|$ becomes $k+1$. However, the Improved k -Gap-Greedy Algorithm is designed such that in that moment u must have been deleted from V_C^1 and all edges with the head at u including (u, v) are also deleted from E_C and will not be inserted to E'_C . This contradicts the fact that $(u, v) \in E'_C$. Thus, we proved $|E_{(u,v)}^u| \leq |\mathcal{E}_{(u,v)}^u| < 2(k+1)^2$.

In a similar way, we can prove that $|E_{(u,v)}^v| < 2(k+1)^2$. Therefore, $|E_{(u,v)}^u \cup E_{(u,v)}^v| < 4(k+1)^2$. Hence, we can partition the edges from E'_i into at most $4(k+1)^2$ groups such that the edges in each group satisfy the (ω/\sqrt{d}) -gap property. Thus, the edges in E' can be partitioned into $\mathcal{O}((c/\alpha)^{d-1}k^2)$ groups such that the edges in each group satisfy the (ω/\sqrt{d}) -gap property. To conclude the proof we apply Claim 4.3 to obtain an upper bound for a total cost of E' , which is $\mathcal{O}((c/\alpha)^{d-1}k^2(\sqrt{d}/\omega) \log n)$ times the cost of an MST of V . \square

Details of the implementation of the Improved k -Gap-Greedy Algorithm. Our description of the Improved k -Gap-Greedy Algorithm is on a high level and now we discuss how one can implement that algorithm efficiently. In order to obtain an efficient implementation we must provide efficient data structures that allow us to query for (i) an edge $(u, v) \in E_C$ with minimum dist_C , (ii) the number of edges in E'_C beginning or ending at u , (iii) reporting all points in $N_{(u,v)}^u$ and $N_{(u,v)}^v$, and (iv) for verifying if an edge (u, v) is disjoint to all edges in $F_C(x)$ or $H_C(x)$.

It is easy to see that one can maintain the data structure (ii) reporting the number of edges in E'_C beginning or ending at a given vertex with constant query time and update

time. Similarly, one can easily maintain the data structure (iv) verifying if an edge (u, v) is disjoint to all edges in $F_C(x)$ or $H_C(x)$ with $\mathcal{O}(\log k)$ query time and update time. (The bound for the query time follows immediately from the fact that $F_C(x)$ or $H_C(x)$ contains $\mathcal{O}(k)$ edges and one can use a balanced binary search tree to store the endpoints of these edges.)

Arya and Smid [3] gave an efficient data structure supporting query (i) for an edge in E_C with minimum dist_C . As is demonstrated (and discussed in detail) in [3], the total running time needed to perform all these operations is in our case $\mathcal{O}(nk \log^d n)$. (This bound follows from the fact that we use query (i) $\mathcal{O}(kn)$ times, and, as shown in [3], an efficient data structure can be built in time $\mathcal{O}(n \log^d n)$ that has constant query time and $\mathcal{O}(\log^d n)$ amortized update time per edge.)

To report all points in $N_{(u,v)}^u$ and $N_{(u,v)}^v$ efficiently we use a dynamic data structure for orthogonal range queries (see [23]). In our algorithm we only delete points and therefore the amortized deletion time is $\mathcal{O}(\log^{d-1} n)$ and the query time is $\mathcal{O}(\log^{d-1} n)$ time plus the number of reported points [23]. Since each point is deleted at most once, the total deletion time is $\mathcal{O}(n \log^{d-1} n)$. We perform query operations on the dynamic data structure each time after an edge is inserted, so the total time is $\mathcal{O}(nk \log^{d-1} n)$ for $\mathcal{O}(nk)$ edges. Each vertex is reported at most $\mathcal{O}(k^2)$ times; when reported, we verify whether $F_C(x)$ or $H_C(x)$ can be enhanced by adding an edge; the verification and update takes time $\mathcal{O}(\log k)$; thus the total time for reporting, verification, and updating is $\mathcal{O}(nk^2 \log k)$.

In summary, we can conclude the discussion in this section with the following lemma.

Lemma 4.10. *Let V be a set of n points in \mathbb{R}^d . Let α, ω be real numbers such that $0 < \alpha < \pi/4$ and $0 < \omega < \frac{1}{2}(\cos \alpha - \sin \alpha)$. Let $t = 1/(\cos \alpha - \sin \alpha - 2\omega)$. There is a constant c such that the Improved k -Gap-Greedy Algorithm can be implemented to run in time $\mathcal{O}((c/\alpha)^{d-1}(nk \log^d n + nk^2 \log k))$.*

We can conclude the discussion in this section with the following main theorem that follows immediately from Lemmas 4.9 and 4.10.

Theorem 3. *Let α, ω be real numbers such that $0 < \alpha < \pi/4$, $0 < \omega < \frac{1}{2}(\cos \alpha - \sin \alpha)$. Let V be a set of n points in \mathbb{R}^d . Let $t = 1/(\cos \alpha - \sin \alpha - 2\omega)$. There is a constant c such that in $\mathcal{O}((c/\alpha)^{d-1}(nk \log^d n + nk^2 \log k))$ time the Improved k -Gap-Greedy Algorithm computes a directed (k, t) -VFTS having maximum degree $\mathcal{O}((c/\alpha)^{d-1} k)$ and total cost $\mathcal{O}((c/\alpha)^{d-1} k^2 (\sqrt{d}/\omega) \log n W_{\text{MST}})$.*

To conclude this section, we notice that Theorem 3 directly implies Theorem 2.

5. Conclusions and Final Remarks

It is tempting (and very interesting) to try to extend our results to non-Euclidean graphs. One could extend Claim 3.1 to hold for arbitrary graphs, that is, to show that the graph obtained in the k -Greedy Algorithm is a (k, t) -VFTS for arbitrary (i.e., also for non-Euclidean) graphs (see Remark 3.2). Furthermore, since our k -Greedy Algorithm is an extension of the classical greedy algorithm that has been used extensively in the

construction of t -spanners, see, e.g., [1], it is plausible to ask if it produces good quality spanners for arbitrary graphs or for planar graphs. For example, it follows from [1] that if the k -Greedy Algorithm with $k = 0$ is run on a planar graph, then it produces a t -spanner whose total cost is upper bounded by $\mathcal{O}(1/(t - 1))$ times the MST cost. However, this result cannot be generalized to larger k , as we show below.

We consider the k -Greedy Algorithm for $k = 1$ and $t = 1 + \varepsilon$ for a positive real number ε . Let $G = (V, E)$ be a planar, weighted graph on n vertices with $V = \{v_1, v_2, \dots, v_n\}$ and $E = E_1 \cup E_2 \cup \{(v_n, v_1)\}$, where $E_1 = \{(v_1, v_2), (v_2, v_3), \dots, (v_{n-1}, v_n)\}$ and $E_2 = \{(v_1, v_3), (v_2, v_4), \dots, (v_{n-2}, v_n)\}$. Let $\text{cost}(e)$ denote the cost of edge (u, v) . We define the edge costs as follows:

$$\text{cost}(e) = \begin{cases} 10 & \text{if } e = (v_n, v_1), \\ \varepsilon/n & \text{if } e \in E_1, \\ 10 - \varepsilon & \text{if } e \in E_2. \end{cases}$$

It is easy to see that the minimum-cost 2-vertex connected subgraph of G is $G' = (V, E')$ with $E' = E_1 \cup \{(v_n, v_1)\}$, which is also an optimal $(1, 1 + \varepsilon)$ -VFTS (or $(1, 1 + \varepsilon)$ -EFTS). However, if we run the greedy algorithm on G , it will output G itself. Therefore, the optimal cost of a $(1, 1 + \varepsilon)$ -VFTS (or $(1, 1 + \varepsilon)$ -EFTS) is $10 + (n - 1) \cdot \varepsilon/n$, which for very small ε is about 10, while the cost of the graph obtained by the greedy algorithm is $10 + (n - 2) \cdot (10 - \varepsilon) + (n - 1) \cdot \varepsilon/n$, which for small ε is about $10 \cdot (n - 1)$. Figure 3 gives an example for $n = 8$.

An obvious open question that is left after our paper is whether one can design an efficient algorithm that outputs fault-tolerant spanners having properties from Theorem 1. We believe that it should be possible to design an $\tilde{\mathcal{O}}(nk)$ -time ($\mathcal{O}(nk \text{ polylog } n)$) algorithm for that problem.

There are many algorithmic applications of spanners, perhaps the most appealing being the recent application in an $\mathcal{O}(n \log n)$ -time approximation algorithm for the Euclidean TSP [27]. On the other hand, we do not know of many applications of fault-tolerant spanners. Actually, even in the most natural application to the k -connectivity

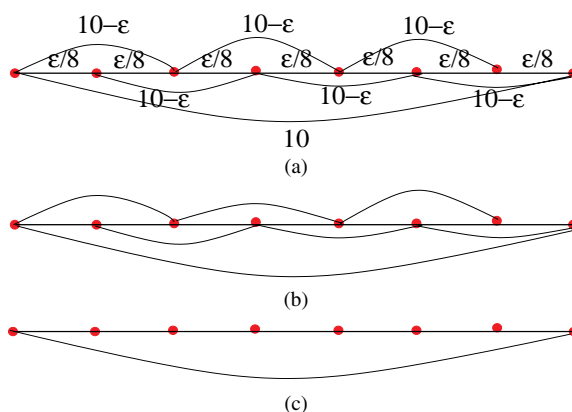


Fig. 3. (a) A weighted planar graph G , (b) $(1, 1 + \varepsilon)$ -VFTS (or $(1, 1 + \varepsilon)$ -EFTS) of G generated by the greedy algorithm with $t = 1 + \varepsilon$, and (c) minimum $(1, 1 + \varepsilon)$ -VFTS (or $(1, 1 + \varepsilon)$ -EFTS) of G .

problem, the fastest approximation algorithms do not use fault-tolerant spanners [9]–[11]. Therefore, in near future we intend to investigate the relationship between the connectivity problems in Euclidean graphs and the notion of fault-tolerant spanners.

References

1. I. Althöfer, G. Das, D. Dobkin, D. Joseph, and J. Soares. On sparse spanners of weighted graphs. *Discrete & Computational Geometry*, 9:81–100, 1993. A preliminary version entitled “Generating sparse spanners for weighted graphs” appeared in J. R. Gilbert and R. G. Karlsson, editors, *Proceedings of the 2nd Scandinavian Workshop on Algorithm Theory*, Bergen, Norway, July 11–14, 1990, pages 26–37. Volume 447 of Lecture Notes in Computer Science. Springer-Verlag, Berlin.
2. S. Arya, G. Das, D. M. Mount, J. S. Salowe, and M. Smid. Euclidean spanners: Short, thin, and lanky. In *Proceedings of the 27th Annual ACM Symposium on Theory of Computing*, Las Vegas, NV, May 29 – June 1, 1995, pages 489–498. ACM Press, New York.
3. S. Arya and M. Smid. Efficient construction of a bounded-degree spanner with low weight. *Algorithmica*, 17(1):33–54, January 1997. A preliminary version appeared in J. van Leeuwen, editor, *Proceedings of the 2nd Annual European Symposium on Algorithms*, Utrecht, The Netherlands, September 26–28, 1994, pages 48–59. Volume 855 of Lecture Notes in Computer Science. Springer-Verlag, Berlin.
4. M. de Berg, M. van Kreveld, M. Overmars, and O. Schwarzkopf. *Computational Geometry – Algorithms and Applications*. Springer-Verlag, Berlin, 1997.
5. M. Bollobas. *Modern Graph Theory*. Springer-Verlag, Berlin, 1998.
6. P. B. Callahan and S. R. Kosaraju. A decomposition of multidimensional point sets with applications to k -nearest-neighbors and n -body potential fields. *Journal of the ACM*, 42(1):67–90, January 1995. A preliminary version appeared in *Proceedings of the 24th Annual ACM Symposium on Theory of Computing*, Victoria, British Columbia, Canada, May 4–6, 1992, pages 546–556. ACM Press, New York.
7. B. Chandra, G. Das, G. Narasimhan, and J. Soares. New sparseness results on graph spanners. In *Proceedings of the 8th Annual ACM Symposium on Computational Geometry*, Berlin, Germany, June 10–12, 1992, pages 192–201. ACM Press, New York.
8. P. L. Chew. There is a planar graph as good as the complete graph. In *Proceedings of the 2nd Annual ACM Symposium on Computational Geometry*, Yorktown Heights, NY, June 2–4, 1986, pages 169–177. ACM Press, New York.
9. A. Czumaj and A. Lingas. On approximability of the minimum-cost k -connected spanning subgraph problem. In *Proceedings of the 10th Annual ACM–SIAM Symposium on Discrete Algorithms*, Baltimore, MD, January 17–19, 1999, pages 281–290. SIAM, Philadelphia, PA.
10. A. Czumaj and A. Lingas. Fast approximation schemes for Euclidean multi-connectivity problems. In U. Montanari, J. D. P. Rolim, and E. Welzl, editors, *Proceedings of the 27th Annual International Colloquium on Automata, Languages and Programming*, Geneva, Switzerland, July 9–15, 2000, pages 856–868. Volume 1853 of *Lecture Notes in Computer Science*. Springer-Verlag, Berlin.
11. A. Czumaj, A. Lingas, and H. Zhao. Polynomial-time approximation schemes for the Euclidean survivable network design problem. In P. Widmayer, F. Triguero, R. Morales, M. Hennessy, S. Eidenbenz, and R. Conejo, editors, *Proceedings of the 29th Annual International Colloquium on Automata, Languages and Programming*, Malaga, Spain, July 8–13, 2002, pages 973–984. Volume 2380 of *Lecture Notes in Computer Science*. Springer-Verlag, Berlin.
12. G. Das, P. Heffernan, and G. Narasimhan. Optimally sparse spanners in 3-dimensional Euclidean space. In *Proceedings of the 9th Annual ACM Symposium on Computational Geometry*, San Diego, CA, May 19–21, 1993, pages 53–62. ACM Press, New York.
13. G. Das and G. Narasimhan. A fast algorithm for constructing sparse Euclidean spanners. *International Journal of Computational Geometry and Applications*, 7(4):293–315, 1997. A preliminary version appeared in *Proceedings of the 10th Annual ACM Symposium on Computational Geometry*, Stony Brook, NY, June 6–8, 1994, pages 132–139. ACM Press, New York.
14. G. Das, G. Narasimhan, and J. Salowe. A new way to weigh malnourished Euclidean graphs. In *Proceedings of the 6th Annual ACM–SIAM Symposium on Discrete Algorithms*, San Francisco, CA, January 22–24, 1995, pages 215–222. SIAM, Philadelphia, PA.

15. D. Eppstein. Spanning trees and spanners. In J.-R. Sack and J. Urrutia, editors, *Handbook of Computational Geometry*, chapter 9, pages 425–461. Elsevier Science, Amsterdam, 1997.
16. J. Gao, L. J. Guibas, J. Hershburger, L. Zhang, and A. Zhu. Geometric spanner for routing in mobile networks. In *Proceedings of the 2nd ACM Symposium on Mobile Ad Hoc Networking & Computing (MobiHoc 2001)*, Long Beach, CA, October 4–5, 2001, pages 45–55.
17. M. Grünewald, T. Lukovszki, C. Schindelhauer, and K. Volbert. Distributed maintenance of resource efficient wireless network topologies. In B. Monien and R. Feldmann, editors, *Proceedings of the 8th Euro-Par*, Paderborn, Germany, August 27–30, 2002, pages 935–946. Volume 2400 of Lecture Notes in Computer Science. Springer-Verlag, Berlin.
18. J. Gudmundsson, C. Levcopoulos, and G. Narasimhan. Fast greedy algorithms for constructing sparse geometric spanners. *SIAM Journal on Computing*, 31(5):1479–1500, August 2002. A preliminary version appeared in M. M. Halldórsson, editor, *Proceedings of the 7th Scandinavian Workshop on Algorithm Theory*, Bergen, Norway, July 5–7, 2000, pages 314–327. Volume 1851 of Lecture Notes in Computer Science. Springer-Verlag, Berlin.
19. C. Levcopoulos and A. Lingas. There are planar graphs almost as good as the complete graphs and almost as cheap as minimum spanning trees. *Algorithmica*, 8:251–256, 1992. A preliminary version appeared in H. Djidjev, editor, *Proceedings of the International Symposium on Optimal Algorithms*, Varna, Bulgaria, 1989, pages 9–13. Volume 401 of Lecture Notes in Computer Science. Springer-Verlag, Berlin.
20. C. Levcopoulos, G. Narasimhan, and M. H. M. Smid. Improved algorithms for constructing fault-tolerant spanners. *Algorithmica*, 32(1):144–156, 2002. A preliminary version appeared in *Proceedings of the 30th Annual ACM Symposium on Theory of Computing*, Dallas, TX, May 23–26, 1998, pages 186–195, ACM Press, New York.
21. T. Lukovszki. New results on fault tolerant geometric spanners. In F. Dehne, A. Gupta, J.-R. Sack, and R. Tamassia, editors, *Proceedings of the 6th Workshop on Algorithms and Data Structures*, Vancouver, Canada, August 11–14, 1999, pages 193–204. Volume 1663 of Lecture Notes in Computer Science. Springer-Verlag, Berlin.
22. T. Lukovszki. New Results on Geometric Spanners and Their Applications. Ph.D. thesis, University of Paderborn, 1999.
23. K. Mehlhorn and S. Näher. Dynamic fractional cascading. *Algorithmica*, 5:215–241, 1990.
24. D. Peleg. *Distributed Computing: A Locality-Sensitive Approach*. Society for Industrial and Applied Mathematics, Philadelphia, PA, 2000.
25. D. Peleg and A. Schäffer. Graph spanners. *Journal of Graph Theory*, 13:99–116, 1989.
26. R. Rajaraman. Topology control and routing in ad hoc networks: a survey. *SIGACT News*, 33:60–73, June 2002.
27. S. B. Rao and W. D. Smith. Approximating geometrical graphs via “spanners” and “banyans”. In *Proceedings of the 30th Annual ACM Symposium on Theory of Computing*, Dallas, TX, May 23–26, 1998, pages 540–550. ACM Press, New York.
28. J. Ruppert and R. Seidel. Approximating the d -dimensional complete Euclidean graph. In *Proceedings of the 3rd Canadian Conference on Computational Geometry*, Simon Fraser University, Vancouver, Canada, August 6–10, 1991, pages 207–210.
29. M. Smid. Closest-point problems in computational geometry. In J.-R. Sack and J. Urrutia, editors, *Handbook of Computational Geometry*, chapter 20, pages 877–935. Elsevier Science Amsterdam, 1997.
30. A. C. Yao. On constructing minimum spanning trees in k -dimensional spaces and related problems. *SIAM Journal on Computing*, 11:721–736, 1982.

Received May 15, 2003, and in revised form January 26, 2004. Online publication June 18, 2004.