New Sparseness Results on Graph Spanners

Barun Chandra[*] Gautam Das[†] Giri Narasimhan[†] José Soares[‡]

March 13, 1992

Abstract

Let $G = (V, E)$ be an n-vertex connected graph with positive edge weights. A subgraph $G' = (V, E')$ is a *t-spanner* of G if for all $u, v \in V$, the weighted distance between u and v in G' is at most t times the weighted distance between u and v in G. We consider the problem of constructing sparse spanners. Sparseness of spanners is measured by two criteria, the *size*, defined as the number of edges in the spanner, and the *weight*, defined as the sum of the edge weights in the spanner. In this paper, we concentrate on constructing spanners of small weight.

For an arbitrary positive edge-weighted graph G, for any $t > 1$, and any $\epsilon > 0$, we show that a t-spanner of G with weight $O(n^{\frac{2+\epsilon}{t-1}}) \cdot wt(MST)$ can be constructed in polynomial time. We also show that $(\log^2 n)$-spanners of weight $O(1) \cdot wt(MST)$ can be constructed.

We then consider spanners for complete graphs induced by a set of points in d-dimensional real normed space. The weight of an edge xy is the norm of the $\overrightarrow{xy}$ vector. We show that for these graphs, t-spanners with total weight $O(\log n) \cdot wt(MST)$ can be constructed in polynomial time.

1 Introduction

Let $G = (V, E)$ be an n-vertex connected graph with positive edge weights. The distance in G between two vertices $u, v \in V$ is the length of the shortest weighted path between u and v and it is denoted by $d_G(u, v)$. A subgraph $G' = (V, E')$ is a *t-spanner* of G if for every $u, v \in V$, $d_{G'}(u, v) \leq t \cdot d_G(u, v)$. The value of t is called the *stretch factor* of G'.

Sparseness of spanners is measured by two criteria, the size and the weight. The *size* of a graph G, $size(G)$, is defined as the number of edges in G and the *weight* of G, $wt(G)$, is defined as the sum of the edge weights of G. The minimum spanning tree (MST) of G is obviously the sparsest possible connected spanner both in terms of size and weight, but its stretch factor can be as bad as $n - 1$ [1]. The sparseness of a spanner G' is judged by comparing it to the size and weight of the MST.

Spanners have numerous applications. Spanners for *Euclidean graphs,* are of special interest for robotics researchers, and have been considered in several works [1, 7, 8, 9, 13, 20, 21, 22, 24]. Unit edged spanners appear in distributed systems and communications network design [2, 16, 17], and in genetics [4]. A similar concept, *dilation,* appears in the design of universal parallel machines [5]. In [15], spanners have been investigated for chordal graphs and directed graphs. Recently, spanners of graphs with *arbitrary* positive edge weights have been investigated [1, 3, 8].

In all the above research, fairly tight sparseness results have been achieved in the size of spanners, but not in the weight. This paper concentrates mainly on the weight criterion, and presents significantly improved results.

We first consider arbitrary positive edge-weighted graphs. Peleg and Schäffer [15] studied this problem for the special case of unit edge-weighted graphs. For these graphs, the size and weight represent the same quantity. They showed that for every t there are infinitely many n such that there exists an n-vertex graph with the property that every t-spanner for this graph requires $\Omega(n \cdot n^{\frac{1}{t}})$ edges. For arbitrary positive edge-weighted graphs, this gives a lower bound of $\Omega(n \cdot n^{\frac{1}{t}}) = \Omega(n^{\frac{1}{t}}) \cdot size(MST)$ on the size of a t-spanner, and a lower bound of $\Omega(n^{\frac{1}{t}}) \cdot wt(MST)$ on the weight. They also proved an upper bound of $O(n \cdot n^{\frac{4}{t}})$ on the size of spanners of unit edge-weighted graphs. However, the techniques used to derive the upper bound for unit edge-weighted graphs do not generalize to arbitrary edge-weighted graphs.

Spanners for arbitrary edge-weighted graphs were considered by Althöfer et al. [1], and it was shown that a natural *greedy algorithm* constructs t-spanners with size

[*]Dept. of Computer Science, University of Chicago, Chicago,IL 60637.

[†]Mathematical Sciences Dept. , Memphis State University

[‡]University of Chicago, and Universidade de São Paulo, Brazil.
Supported by CNPq proc 203039/87.4 (Brazil), and NSF grant CCR-9014562.

8th Annual Computational Geometry, 6/92, Berlin, Germany
©1992 ACM 89791-518-6/92/0006/0192 .... $1.50

$O(n^{1+\frac{2}{t-1}})$, and weight less than $(1+\frac{n}{t-1}) \cdot wt(MST)$. While the size bound is optimal within a constant factor in the exponent of n, the weight bound is suboptimal. Recently [23], the upper bound on the weight was reduced to $O(n^{\frac{1}{2}+\frac{1}{t-1}}) \cdot wt(MST)$, which is still suboptimal. No spanners of smaller weight were known to exist for arbitrary edge-weighted graphs.

We are able to improve the weight bound of the spanner constructed by the same greedy algorithm (described in Section 2) through improved analysis. Our results show not only the existence, but also the polynomial time constructibility, of spanners which are almost optimal in both size and weight.

Theorem 1.1 *Let G be any n-vertex connected graph with positive edge weights. Let $t > 1$ be any real number. For all $\epsilon > 0$ the greedy algorithm constructs a t-spanner of G with weight $O(n^{\frac{2+\epsilon}{t-1}}) \cdot wt(MST)$. The constant implicit in the big O notation depends only on t and ϵ.*

The above result and the size bound of $O(n^{\frac{2}{t-1}}) \cdot size(MST)$ from [1] are clearly optimal within a factor of 2 in the exponent of n, but are likely to be optimal even in the exponent of n; as we observe in Section 3, any improvement in the exponent of n in the expression above would imply a better bound for an open extremal graph theory problem.

We next consider the problem of finding sparse spanners for *geometric graphs*. A *geometric graph* is a complete graph induced by a set of points in d-dimensional real normed space. The weight of an edge xy is the norm of the $\overrightarrow{xy}$ vector. Various schemes have generated $O(1)$-spanners for the cases of Euclidean (L_2 metric) and other L_p metrics [1, 7, 9, 11, 12, 13, 14, 20, 21, 22, 24]. The primary focus of most of these papers was to design efficient algorithms for spanner construction. However, though the spanners in these results had $O(n)$ edges (which is optimal), they could be *arbitrarily large* in total edge weight. A few papers considered weight sparseness of spanners of geometric graphs in two-dimensional Euclidean space [1, 9, 14], and for this special case these papers show that there exist $O(1)$-spanners with size $O(n)$ and weight $O(1) \cdot wt(MST)$. However, the techniques used cannot be extended to higher dimensions, for which the only weight bounds known were bounds for arbitrary edge-weighted graphs. In this paper we use two different techniques to prove much smaller weight bounds on spanners for geometric graphs.

For geometric graphs under any norm and in any dimension, we prove the existence of (and the polynomial time constructibility of) spanners of both optimal size and small weight. The weight result may not be optimal since the best known lower bound is the trivial one of $\Omega(1) \cdot wt(MST)$.

Theorem 1.2 *Let $G = (V, E)$ be any n-vertex geometric graph, where V is a set of points in d-dimensional space under an arbitrary norm. Let $t > 1$ be any real number. There exists a polynomial time algorithm that constructs a t-spanner of G of size $O(n)$ and weight $O(\log n) \cdot wt(MST)$. The constant implicit in the big O notation depends on t, d and the norm.*

We show that the greedy algorithm produces the desired spanner. There is a new contribution here in terms of size sparseness. Although various schemes [1, 20, 21, 24] have produced spanners with bounded *average* vertex degree (this follows since these spanners have size $O(n)$), it was not known whether bounded degree spanners exist for geometric graphs. We show that the spanners built by the greedy algorithm have bounded degree under any norm and for any dimension. The bound on the maximum degree depends on t, the dimension, and the norm. Constructing bounded degree spanners has applications in the problem of selecting the k^{th} smallest inter-distance among n points in space [21].

We obtain an alternative proof about the constructibility of t-spanners with weight $O(\log n) \cdot wt(MST)$ for geometric graphs under an L_p norm. We present, for complete graphs from arbitrary metric spaces, a transformational technique for converting spanners of small size into spanners of small size and small weight. Using Vaidya's [24] or Salowe's algorithm [21] for $O(n)$ size spanners, and then the transformational technique, we obtain the small weight spanner. The advantage of this method is that it is much faster than the greedy algorithm.

One unifying aspect of this paper is that the proof structures of most of the theorems are similar, although additional geometric properties are exploited to give stronger results for geometric graphs. To bound the weight of the spanner G', we cover the edges in G' by a collection of artificially constructed graphs, each of which has edges of approximately the same weight. The weight of any of these graphs can be bounded by first estimating its size and scaling the size by its maximum edge weight. Finally we cumulate the weights of all these graphs to obtain the desired bound. In constructing these graphs, we use a technique of *collapsing* clusters of original vertices into single vertices. For the spanners constructed by the greedy algorithm this clustering is purely for the purposes of analysis, but in the case of the transformational technique (Lemma 4.4) we actually construct the spanner using these clusters.

In Section 2 we introduce some definitions and notation, and summarizes results of previous papers which will be used in the proofs. Section 3 deals with the proof of Theorems 1.1. Section 4 deals with the two proofs of Theorem 1.2. We conclude with some open problems in Section 5.

2 Preliminaries

For simplicity of notation, we assume that n is a power of 2 and omit all floor and ceiling signs; all proofs can be repeated without this assumption with no real complications, at the cost of an increase in the constants.

We begin by stating some definitions and properties that will be used throughout the paper.

A *metric space* (V, d) is a set V of points and a function $d : V \times V \rightarrow \mathbb{R}$, called *distance*, satisfying the following properties for all $x, y, z \in V$:

(i) $d(x, y) \geq 0$ and $d(x, y) = 0$ if and only if $x = y$;

(ii) $d(x, y) = d(y, x)$;

(iii) $d(x, z) \leq d(x, y) + d(y, z)$.

A *real normed vector space* is a vector space V over $\mathbb{R}$ and a function $||\cdot|| : V \rightarrow \mathbb{R}$, called the *norm*, satisfying the the following properties for all $x, y \in V$:

(i) $||x|| \geq 0$ and $||x|| = 0$ if and only if $x = 0$;

(ii) $||cx|| = |c| \cdot ||x||$ for every $c \in \mathbb{R}$;

(iii) $||x + y|| \leq ||x|| + ||y||$.

A real normed vector space generates a metric space where the distance function is defined as $d(x, y) = ||x - y||$. If the norm of a d-dimensional vector $x = (x_1, x_2, \ldots, x_d)$ is defined as $\sqrt[p]{|x_1^p| + |x_2^p| + \cdots + |x_d^p|}$, where p is a positive integer, then the corresponding metric is called the L_p metric. The L_2 metric is called *Euclidean metric.*

A *geometric graph* is a complete graph in which the vertex set is a set of points in d-dimensional real normed space and the edge weights are the distances between the points, according to the metric generated by the given norm. When the norm considered is the Euclidean norm, the graph is called *Euclidean graph.*

We now summarize some useful results. Let G be an n-vertex connected graph with positive edge weights.

We reproduce here a greedy algorithm which is described in [1]. This is a simple generalization of Kruskal's algorithm to build a minimum spanning tree. It takes as input a weighted graph $G = (V, E)$, and a real number $t > 1$. It produces as output a subgraph $G' = (V, E')$.

```
Greedy Algorithm
Input: a graph $G = (V, E)$ and a real number $t > 1$.
Output: a $t$-spanner $G' = (V, E')$ of $G'$.
begin
order the edges in $E$ in nondecreasing order of weights;
$E' \leftarrow \emptyset$; $G' \leftarrow (V, E')$;
for each edge $(x, y) \in E$ (from the sorted list) do
    if $d_{G'}(x, y) > t \cdot d_G(x, y)$ then
            $E' \leftarrow E' \cup \{xy\}$;
            $G' \leftarrow (V, E')$;
output $G'$;
end.
```

Let T be a minimum spanning tree (MST) for G built by Kruskal's algorithm which considers edges in the same order as the Greedy Algorithm.

Let P be the Hamiltonian path drawn by taking the preorder traversal of T. The weight of an edge uv in P is defined as $wt(uv) = d_T(u, v)$. Let $L = wt(P)$. It is well known that $L \leq 2wt(T)$.

The following lemmas (proven in [1]) describe some properties of the output graph G'.

Lemma 2.1 *G' is a t-spanner of G.*

Lemma 2.2 *T is a subgraph of G'.*

The next is an extremal graph theoretic lemma that is an easy consequence of a theorem proven in [6, Theorem 3.7, Chapter III]. Define the *girth* of a graph as the number of edges in its smallest cycle. This lemma provides an upper bound on the size of a graph with a given girth.

Lemma 2.3 *Every n-vertex graph G with girth at least g has at most $n \left\lceil n^{\frac{2}{g-2}} \right\rceil$ edges.*

The following facts are obvious:

Fact 2.4 *For all $u, v \in V$, $d_P(u, v) \geq d_T(u, v)$.*

3 Graphs with Arbitrary Edge Weights

In this section we consider graphs with positive edge weights. We first prove a technical lemma from which we obtain as immediate corollaries Theorem 1.1 and a result about $(\log^2 n)$-spanners.

Lemma 3.1 *Let G be any n-vertex connected graph with positive edge weights. For every $t > 1$, the Greedy Algorithm builds a t-spanner $G' = (V, E')$ of G, such that*

$$wt(G') \leq (3 + \frac{16t}{\delta^2}) \cdot n^{\frac{2+\delta}{t-1-\delta}} \cdot wt(MST),$$

where $t-1 > \delta > 0$ can be chosen to be arbitrarily small.

Proof : From Lemma 2.1, G' is a t-spanner of G so we only need to bound the weight of G'.

The basic method used here is the partition of the edges added by the Greedy Algorithm into phases; in each phase we consider edges of approximately the same weight. We will show that the phases in which the large weight edges are added have only a few edges.

Let T, P, and L be as defined in Section 2. Let $E(T)$ denote the set of edges of T.

Partition the interval $[0, L]$ into $\log n+1$ subintervals:
$I_0 = [0, \frac{L}{n}]$
$I_j = (2^{j-1} \cdot \frac{L}{n}, 2^j \cdot \frac{L}{n}]$ for $j = 1, \ldots, \log n$.

Let $E_j = \{e \in E' \setminus E(T) | wt(e) \in I_j\}$. We are going to estimate the weight of E_j for $j = 0, 1, \ldots, \log n$.

Lemma 3.2 $wt(E_0) \leq 2n^{\frac{2}{t-1}} \cdot wt(MST)$.

Proof : Let C be a cycle containing only edges in E_0. Let the weight of the edges in C in nondecreasing order be $w_1, w_2, \ldots, w_g$. Since the last edge with weight w_g was added to the spanner by the greedy algorithm, $\sum_{i=1}^{g-1} w_i > t \cdot w_g$. Since $w_g \geq w_i$, for all $i < g$, we have that $g > t + 1$. Hence, the girth of the subgraph with edges only from E_0 is greater than $t + 1$, and by Lemma 2.3 the number of edges in E_0 is at most $n^{1+\frac{2}{t-1}}$. Since the weight of each edge in E_0 is at most $L/n \leq 2wt(MST)/n$, we are done. $\square$

Fix $j \geq 1$ and consider E_j. Let G_{j-1} be the graph with edges $E(T) \cup E_0 \cup \cdots \cup E_{j-1}$. Let $a = 2^{j-1}\frac{L}{n}$, so $I_j = (a, 2a]$. Thus, in this phase, each edge added by the Greedy Algorithm has weight between a and $2a$.

Let $c' = 2/\delta$. Divide the path P into $\frac{c'L}{a}$ adjacent intervals (or "clusters"), each of size a/c'. Let n_j be the number of cluster containing at least one vertex from V. Then $n_j \leq c'L/a = c'n/2^{j-1}$. Call an edge uv an *intercluster edge* if u and v lie in different clusters.

Lemma 3.3 *Every edge $uv \in E_j$ is an intercluster edge.*

Proof : By definition of E_j, $uv \notin E(T)$. Consider the path Q in T connecting u to v. By Lemma 2.2 every edge in Q is added to the spanner. The fact that T is a minimum spanning tree of G implies that uv has weight larger than or equal to the weight of each edge in Q. Since the edges are added by the Greedy Algorithm in nondecreasing order of weight, when the edge uv is added to the spanner, each edge in Q had already been added. Since the edge uv did get added to the spanner, we have that $wt(Q) > t \cdot wt(uv)$. Since $c' = 2/\delta$ and $\delta < t - 1$, we have that $t > 1 + 2/c'$. By Fact 2.4, $d_P(u, v) \geq d_T(u, v) = wt(Q)$. By the definition of E_j, $wt(uv) > a$. Combining the inequalities above we obtain

$$d_P(u, v) > t \cdot wt(uv) \geq (1 + 2/c')wt(uv) > 2a/c'.$$

Since the distance in P between two points in the same cluster is at most a/c', u and v lie in different clusters. $\square$

Lemma 3.4 $|E_j| \leq n_j^{\min\{2,1+\frac{2+\delta}{t-1-\delta}\}}$

Proof : Since $|E_j|$ is obviously less than n_j^2, we just have to consider the other term.

Consider the graph H (possibly a multigraph) where all the vertices from V in a single cluster are merged into a single vertex, and the edge-set of H is E_j. By Lemma 3.3 every edge in E_j is an edge in H. Let C be an arbitrary cycle in H with g edges. We will show that $g > \frac{t+1}{1+1/c'}$.

Let the weights of the edges in C nondecreasing order be $w_1, \ldots, w_g$. Before the last (and heaviest) edge was added, there was a path of weight $\leq g \cdot (a/c') + \sum_{i=1}^{g-1} w_i$ (the first term comes from the maximum distance traveled within clusters, and the second from the intercluster distances). Since the last edge did get added by the Greedy Algorithm, we have $g \cdot (a/c') + \sum_{i=1}^{g-1} w_i > w_g \cdot t$. Since $w_g \geq a, w_1, \ldots, w_{g-1}$, we obtain $g > \frac{t+1}{1+1/c'}$.

Hence, in H the girth is larger than $\frac{t+1}{1+1/c'} = \frac{t+1}{1+\delta/2}$. We now invoke Lemma 2.3 to obtain the result. $\square$

Lemma 3.5 *For $j = 1, \ldots, \log n$ we have that*

$$wt(E_j) \leq \frac{16}{\delta^2} \cdot n^{\frac{2+\delta}{t-1-\delta}} \cdot wt(MST) \cdot \left(\frac{t-1}{t}\right)^{j-1}.$$

Proof : In E_j, each edge has weight $\leq 2 \cdot a$, so by Lemma 3.4

$$\begin{aligned}
wt(E_j) &\leq 2 \cdot a \cdot n_j^{\min\{2,1+\frac{2+\delta}{t-1-\delta}\}} \\
&\leq 2 \cdot 2^{j-1}\frac{L}{n} \cdot (c'n/2^{j-1})^{\min\{2,1+\frac{2+\delta}{t-1-\delta}\}} \\
&\leq 2 \cdot c'^2 \cdot L \cdot n^{\frac{2+\delta}{t-1-\delta}} \cdot \left(2^{\frac{2+\delta}{t-1-\delta}}\right)^{-(j-1)} \\
&\leq 4 \cdot c'^2 \cdot wt(MST) \cdot n^{\frac{2+\delta}{t-1-\delta}} \cdot \left(\frac{1}{4^{\frac{1}{t-1}}}\right)^{j-1} \\
&\leq 4 \cdot c'^2 \cdot wt(MST) \cdot n^{\frac{2+\delta}{t-1-\delta}} \cdot \left(\frac{t-1}{t}\right)^{j-1}.
\end{aligned}$$

The last inequality follows from the fact that $4^{\frac{1}{t-1}} \geq 1 + \frac{1}{t-1}$ for every $t > 1$. $\square$

Since $E' = E(T) \cup E_0 \cup E_1 \cup \cdots \cup E_{\log n}$, we use Lemmas 3.2 and 3.5 to obtain

$$\begin{aligned}
wt(G') &\leq wt(MST) + 2n^{\frac{2}{t-1}} \cdot wt(MST) + \\
&\quad \frac{16}{\delta^2} \cdot n^{\frac{2+\delta}{t-1-\delta}} \cdot wt(MST) \cdot \sum_{j=1}^{\log n} \left(\frac{t-1}{t}\right)^{j-1} \\
&\leq wt(MST) + 2n^{\frac{2}{t-1}} \cdot wt(MST) \\
&\quad + \frac{16}{\delta^2} \cdot n^{\frac{2+\delta}{t-1-\delta}} \cdot wt(MST) \cdot t \\
&\leq 3n^{\frac{2}{t-1}} \cdot wt(MST) + \\
&\quad \frac{16t}{\delta^2} \cdot n^{\frac{2+\delta}{t-1-\delta}} \cdot wt(MST) \\
&\leq (3 + \frac{16t}{\delta^2}) \cdot n^{\frac{2+\delta}{t-1-\delta}} \cdot wt(MST).
\end{aligned}$$

This proves Lemma 3.1. □

Theorem 1.1 follows as immediate corollary from Lemma 3.1:

Proof of Theorem 1.1: Setting $\delta = \frac{\epsilon(t-1)}{t+1+\epsilon} < t-1$ we are done, since $\frac{2+\delta}{t-1-\delta} = \frac{2+\epsilon}{t-1}$.

Remark: As mentioned in the Introduction, this is optimal to within a constant factor in the exponent of n. If we could reduce the the exponent of the n-term i.e. if there always exists a t-spanner with weight $\leq O(n^{\frac{2-\epsilon'}{t-1}}) \cdot wt(MST)$, for some $\epsilon' > 0$, this would lead to an improvement in a bound from extremal graph theory [6], namely the bound in Lemma 2.3 would improve to $O(n^{1+\frac{2-\epsilon'}{g-2}})$. The same observation also holds for the size bound of $O(n^{1+\frac{2}{t-1}})$.

Peleg and Schäffer [15] also consider the construction of $O(\log n)$-spanners, and show that for unit edge-weighted graphs, $O(\log n)$-spanners exist with weight $O(1) \cdot wt(MST)$. While we are not able to prove the same result for arbitrary edge-weighted graphs (we obtain $(\log n)$-spanners of weight $O(\log n) \cdot wt(MST)$), we are able to prove a weaker version as another corollary of Lemma 3.1.

Theorem 3.6 *Let G be any n-vertex connected graph with positive edge weights. The greedy algorithm constructs a $(\log^2 n)$-spanner of G with weight $O(1) \cdot wt(MST)$.*

Proof : Setting $t = \log^2 n$, and $\delta = \log n$, we have from From Lemma 3.1

$$\begin{aligned} wt(G') &\leq (3 + \frac{16 \log^2 n}{\log^2 n}) \cdot n^{\frac{2+\log n}{\log^2 n - 1 - \log n}} \cdot wt(MST) \\ &= O(1) \cdot wt(MST). \end{aligned}$$

□

Implementation and time analysis: A naive implementation of the greedy algorithm maintains an all pair shortest paths matrix for the graph G', and updates it (from scratch) each time a new edge is added to G' by the greedy algorithm. The total time taken is the time taken to calculate the all pairs shortest paths for the initial graph G, and the time to do these updates. The all pairs shortest path for G can be calculated in time $O(n^3)$ using Dijkstra's algorithm [10]. Let $f(n,m)$ be the time needed to find all pairs shortest paths in a graph with n vertices and m edges. Let M be the size of the spanner outputted by the greedy algorithm. Then the total time for all the updates is $\sum_{m=0}^{M} f(n,m)$. The best known upper bound on $f(n,m)$ is $O(nm \log_{(2+m/n)} n)$ [10]. For the case of arbitrary edge-weight graphs, we know that $M = O(n^{1+\frac{2}{t-1}})$ edges [1]. Hence the total running is $O(n^{3+\frac{4}{t-1}})$.

4 Geometric Graphs

We look at two different techniques which produce t-spanners for geometric graphs, the Greedy Algorithm and the transformational method. Both spanners have $O(n)$ edges and weight $O(\log n) \cdot wt(MST)$. If we compare the two techniques, the Greedy Algorithm is simpler, produces spanners with bounded degree, and works for all norms (the transformational method works only for L_p norms). On the other hand, the transformational method is much faster; $O(n \log n)$ as opposed to $O(n^3 \log n)$ for the Greedy Algorithm.

4.1 The Greedy Algorithm

We begin by stating a well known property about norms and introducing a few definitions.

Consider a finite dimensional real normed vector space with norm N. Let $d_N(x,y)$ denote the distance between x and y in the metric generated by N. Let $d(x,y)$ denote the Euclidean distance between x and y. It is well known that there exist $k_1, k_2 > 0$ (which depend on the norm N) such that for every x and y

$$k_1 \cdot d_N(x,y) \leq d(x,y) \leq k_2 \cdot d_N(x,y) \qquad (1)$$

By *geometric graph under norm N* we mean a geometric graph in which the edge-weights are the distances in the metric generated by N.

We use in this section the concept of angles. As usual, angles are defined by the inner product implicit in the Euclidean norm. The angles are *unoriented angles,* i.e. every angle is between 0 and π radians.

For a real $t > 1$, let θ be an angle such that $\tan \theta = \frac{k_1(t-1)}{k_2(2t+1)}$. Two vectors $\overrightarrow{ux}$ and $\overrightarrow{vy}$ are said to be *Similar-directional* with respect to N and t if, upon translating $\overrightarrow{vy}$ in space such that v coincides with u, the two vectors form an angle at most θ. More intuitively, Similar-directional means that the two vectors point in almost the same direction, since the angle θ is small. Note that for $\theta < \pi/2$ if $\overrightarrow{ux}$ and $\overrightarrow{vy}$ are Similar-directional then $\overrightarrow{ux}$ and $\overrightarrow{yv}$ *are not* Similar-directional. The norm N and t will be implicit when we write *Similar-directional* instead of *Similar-directional with respect to N and t.*

Let V be a finite set of points in d-dimensional normed space with norm N. Let G be the geometric graph induced by V. Let $G' = (V, E')$ be the output of the Greedy Algorithm on input G and t. The distances considered by the Greedy Algorithm are the distances in the metric generated by N.

Each edge uv in E' corresponds to a pair of vectors, $\overrightarrow{uv}$ and $\overrightarrow{vu}$. Consequently E' corresponds to a set of vectors $\boldsymbol{E}'$. We now present an important technical lemma. Intuitively, this lemma says that if there are two Similar-directional vectors $\overrightarrow{uw}$ and $\overrightarrow{vx}$ in $\boldsymbol{E}'$, then u and v must be separated by a distance greater than

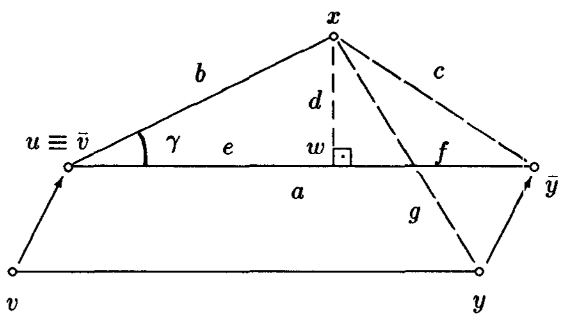

Figure 1: Illustration for Lemma 4.1.

a constant factor times the length of the shorter of the two vectors. Consequently the originating points of two Similar-directional vectors cannot be too close to each other. It may also be noted that as a consequence of this lemma, if there are two vectors originating from the same vertex then they cannot be Similar-directional.

Lemma 4.1 *Let $G = (V, E)$ be a geometric graph in d-dimensional space under norm N, and let $t > 1$ be a real number. Let $G' = (V, E')$ be the t-spanner of G produced by the Greedy Algorithm. Let $k_1, k_2 > 0$ be the constants defined in (1) and $\theta = \arctan \frac{k_1(t-1)}{k_2(2t+1)}$. Let $\overrightarrow{ux}$ and $\overrightarrow{v y}$ be any two Similar-directional vectors in E'. If $d_N(u,x) \leq d_N(v,y)$ then $d_N(u,v) > r \cdot d_N(u,x)$, where $r = \frac{t-1}{6t}$.*

Proof : Let γ be the angle formed by vectors $\overrightarrow{ux}$ and $\overrightarrow{vy}$ upon translating $\overrightarrow{vy}$ in space such that v coincides with u. To prove the lemma we assume that

$$d_N(u,x) \leq d_N(v,y) \quad \text{and} \quad d_N(u,v) \leq r \cdot d_N(u,x) \quad (2)$$

and we prove that under these assumptions $\overrightarrow{ux}$ and $\overrightarrow{vy}$ are not Similar-directional, i. e. $\gamma > \theta$.

Consider the configuration obtained by translating $\overrightarrow{vy}$ in space such that v coincides with u. Let $\bar{v}$ $[\bar{y}]$ be the point corresponding to the translation of v $[y]$. This is a planar configuration in which the plane is defined by the points $u \equiv \bar{v}$, x, and $\bar{y}$. This configuration is illustrated by Figures 1 and 2.

Note that $\tan\theta = \frac{k_1(t-1)}{k_2(2t+1)} < 1$. So, if $\gamma \geq \pi/4$ the vectors $\overrightarrow{ux}$ and $\overrightarrow{vy}$ are not Similar-directional. Thus, we may assume that $\gamma < \pi/4$.

Throughout this proof we use primed lower case letters, $a', b', c', \ldots$, to denote distances in the N metric, while non-primed lower case letters denote distances in the Euclidean metric. For example, if $a' = d_N(x,y)$, then $a = d(x,y)$, and vice-versa.

Let $a' = d_N(v,y)$, $b' = d_N(u,x)$, $c' = d_N(x,\bar{y})$, and $g' = d_N(x,y)$. ($a = d(v,y)$, $b = d(u,x)$, $c = d(x,\bar{y})$, and $g = d(x,y)$ are the corresponding Euclidean distances.)

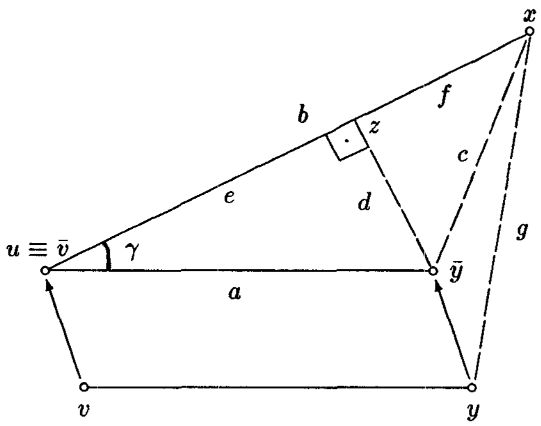

Figure 2: Illustration for Lemma 4.1.

Using this notation, (2) becomes

$$b' \leq a' \quad \text{and} \quad d_N(u,v) \leq rb'. \quad (3)$$

We consider two cases: $g' \geq a'$ and $g' < a'$. Each one of these cases is subdivided into two subcases: $a \geq b$ and $a < b$.

The case in which $a \geq b$ is illustrated in Figure 1, where z is the orthogonal projection of x in the segment $\bar{v}\bar{y}$, $d' = d_N(x,z)$, $e' = d_N(\bar{v},z)$, and $f' = d_N(z,\bar{y})$. Since $\gamma < \pi/4$ and $a \geq b$, z does belong to the segment $\bar{v}\bar{y}$.

The case in which $a < b$ is illustrated in Figure 2, where z is the orthogonal projection of $\bar{y}$ in the segment ux, $d' = d_N(\bar{y},z)$, $e' = d_N(u,z)$, and $f' = d_N(z,x)$. Since $\gamma < \pi/4$ and $a < b$, z does belong to the segment ux.

Case 1: $g' \geq a'$.
Note that by the triangle inequality we have that $g' \leq c' + d_N(u,v)$. The term $d_N(u,v) = d_N(y,\bar{y})$ corresponds to the translation of y.

Subcase 1.1: $a \geq b$ (a and b are Euclidean distances). See Figure 1.

Using (3) and (several times) the triangle inequality, we obtain

$$g' \leq c' + d_N(u,v) \leq c' + rb' \leq d' + f' + rb'$$

implying

$$\begin{aligned} d' &\geq g' - rb' - f' \geq a' - f' - rb' \\ &= e' - rb' \geq e' - r(d' + e') \end{aligned}$$

implying

$$d'(1+r) \geq e'(1-r).$$

Using (1) we have that

$$\frac{1-r}{1+r} \leq \frac{d'}{e'} \leq \frac{k_2 d}{k_1 e} = \frac{k_2}{k_1}\tan\gamma,$$

implying

$$\begin{aligned}\tan\gamma &\geq \frac{k_1(1-r)}{k_2(1+r)} = \frac{k_1(5t+1)}{k_2(7t-1)} \\ &> \frac{k_1(t-1)}{k_2(2t+1)} = \tan\theta.\end{aligned}$$

Since $\tan\gamma > \tan\theta$, we have that $\gamma > \theta$, as desired.
Subcase 1.2: $a < b$ (a and b are Euclidean distances). See Figure 2.

Using (3) and (several times) the triangle inequality, we obtain

$$g' \leq c' + d_N(u,v) \leq c' + rb' \leq d' + f' + rb'$$

implying

$$\begin{aligned} d' &\geq g' - rb' - f' \geq b' - f' - rb' \\ &= e' - rb' \geq e' - ra' \geq e' - r(d'+e')\end{aligned}$$

implying

$$d'(1+r) \geq e'(1-r).$$

Similarly to Subcase 1.1, we obtain $\gamma > \theta$.
Case 2: $g' < a'$.
Among the four edges of G, uv, ux, vy, and xy, the edge vy is the largest (under norm N), and consequently will be examined by the algorithm last. In the spanner being built by the Greedy Algorithm there are paths $P(u,v)$ and $P(x,y)$ such that the lengths of $P(u,v)$ and $P(x,y)$ are at most $t \cdot d_N(u,v)$ and $t \cdot d_N(x,y)$ respectively. This means that when (v,y) is being examined, there exists a path from v to y (via $P(v,u)$, (u,x), and $P(x,y)$). Let $P(v,y)$ refer to this path. Since the edge vy was added to the spanner by the algorithm, we have that

$$\begin{aligned} ta' &< wt(P(v,y)) \\ &= wt(P(v,u)) + wt(ux) + wt(P(x,y)) \\ &\leq t \cdot d_N(v,u) + b' + tg' \\ &\leq t \cdot d_N(v,u) + b' + t(c' + d_N(v,u)) \\ &= 2t \cdot d_N(v,u) + b' + tc' \\ &\leq 2trb' + b' + tc' \end{aligned}$$

implying

$$b'(1+2tr) > t(a'-c'). \qquad (4)$$

We again inspect two subcases:
Subcase 2.1: $a \geq b$. See Figure 1.
Using the triangle inequality, we obtain from (4) that

$$(e'+d')(1+2tr) \geq b'(1+2tr)$$

$$> t(a'-c') \geq t(a'-d'-f') = t(e'-d')$$

or

$$\frac{d'}{e'} > \frac{t(1-2r)-1}{t(1+2r)+1}.$$

Using (1) we have that

$$\frac{t(1-2r)-1}{t(1+2r)+1} < \frac{d'}{e'} \leq \frac{k_2 d}{k_1 e} = \frac{k_2}{k_1}\tan\gamma,$$

implying

$$\tan\gamma > \frac{k_1(t(1-2r)-1)}{k_2(t(1+2r)+1)} = \tan\theta.$$

Since $\tan\gamma > \tan\theta$, we have that $\gamma > \theta$, as desired.
Subcase 2.2: $a < b$. See Figure 2.

Using the triangle inequality, we obtain from(3) and (4) that

$$a'(1+2tr) \geq b'(1+2tr) > t(a'-c') \geq t(b'-c')$$

$$= t(e'+f'-c') \geq t(e'+f'-f'-d') = t(e'-d'),$$

implying

$$td' > te' - a'(1+2tr) \geq te' - (d'+e')(1+2tr)$$

implying

$$\frac{d'}{e'} > \frac{t(1-2r)-1}{t(1+2r)+1}.$$

Similarly to Subcase 2.1, we obtain $\gamma > \theta$.

This completes the proof of Lemma 4.1. □

As a consequence of Lemma 4.1, if there are two vectors originating from the same vertex then they cannot be Similar-directional: the angle formed by these vector is at least $\theta = \arctan\frac{k_1(t-1)}{k_2(2t+1)}$. Let $A(d,\theta)$ be the maximum number of rays (half-lines) from a point in d-dimensional Euclidean space such that each pair of rays forms an angle at least θ. It follows that $A(d,\theta)$ is a bound on the degree of the t-spanner constructed by the Greedy Algorithm. This generalizes an observation made in [22] that the greedy algorithm builds bounded degree spanners for the special case of Euclidean graphs.

Let σ_d denote the unit sphere with center v in $\mathbb{R}^d$. A finite set of points on σ_d is called a *spherical code*. It is easy to see that $A(d,\theta)$ is the maximum cardinality of a spherical code V such that the $\angle xvy \geq \theta$ for each $x \neq y \in V$. This *packing problem* has been extensively studied. We mention the following upper bound due to Rankin [18]:

$$A(d,\theta) = O(d^{3/2}(\sqrt{2}\sin(\theta/2))^{-d}. \qquad (5)$$

The following theorem follows from the above observations:

Theorem 4.2 *Let $G = (V,E)$ be an n-vertex geometric graph in d-dimensional space under norm N. For every $t > 1$, the t-spanner of G produced by the Greedy Algorithm has maximum degree bounded by a constant that depends on d, t, and N.*

As a corollary, this shows that if d, t, and N are fixed, t-spanners of geometric graphs produced by the Greedy Algorithm have only $O(n)$ edges. We now analyze the weight of the t-spanners.

Fix an angle $\theta > 0$. Consider a cover of $\mathbb{R}^d$ by $B(d,\theta)$ circular (overlapping) cones, all having the same focus, such that two points in the same cone form an angle at most θ. We use in Theorem 4.3 the well known fact that $B(d,\theta)$ is finite for every $\theta > 0$ and every d. This *covering problem*, related to the packing problem mentioned above, has also been extensively studied. We mention the following upper bound due to Rogers [19]:

$$B(d,\theta) = O(d^{3/2} \log \frac{d}{\sin(\theta/2)} \sin^{-d}(\theta/2)).$$

Theorem 4.3 *Let $G = (V,E)$ be an n-vertex geometric graph in d-dimensional space under norm N. Let $G' = (V,E')$ be the t-spanner of G produced by the Greedy Algorithm. Then the weight of the spanner is at most $O(\log n) \cdot wt(MST)$. The constant implicit in the big O depends on d, t, and N.*

Proof : Let $\theta = \arctan \frac{k_1(t-1)}{k_2(2t+1)}$, where the constants $k_1, k_2 > 0$ are according to (1). At some arbitrary d-dimensional point, we cover the space by a constant number of circular cones $C_1, C_2, \ldots, C_k$, such that every two lines incident on that point and lying within the same cone subtend an angle at most θ. As noted, k depends only on d and θ.

Call these the *original cones*. One could imagine similar k cones around each of the n vertices. The cones around each of the n vertices could be obtained by making copies of the original cones and then translating them to each of the vertices. Hence each vertex has exactly one cone corresponding to each of the original cones $C_1, C_2, \ldots, C_k$.

Let $\boldsymbol{E}'$ be the set of vector corresponding to E'. Let $\boldsymbol{E}'_i$ be the set of vectors in $\boldsymbol{E}'$ that appears in the cones corresponding to the same cone C_i. We claim that the sum of the weights of the vectors in $\boldsymbol{E}'_i$ is bounded by $O(\log n) \cdot wt(MST)$, for $1 \leq i \leq k$. It may be noted that since the sets $\boldsymbol{E}'_i$ cover the set $\boldsymbol{E}'$, proving this claim is enough to prove the lemma.

Clearly all the vectors in $\boldsymbol{E}'_i$ are Similar-directional. Hence by Lemma 4.1, if $\overrightarrow{u_1 v_1}$ and $\overrightarrow{u_2 v_2}$ are two vectors in $\boldsymbol{E}'_i$, and if the former vector is shorter, then $wt(u_1 u_2)$ must be larger than $r \cdot wt(u_1 v_1)$, where $r = \frac{t-1}{6t}$.

At this stage we can use the same proof technique as before; partition the edges of the spanner into sets of edges of roughly the same weight and then analyse each of these separately. However, it turns out that for the geometric case, there is a much simpler proof.

Consider an optimal *traveling salesman circuit, TSC*, of the n points. Let $L = wt(TSC)$. It is well known that $L \leq 2wt(MST)$. We are now going to account for the length of the vectors in $\boldsymbol{E}'_i$ using the length of the edges in the TSC. Imagine a walk along the circuit TSC starting from some vertex v. As we walk along the circuit we encounter the originating points of the vectors in $\boldsymbol{E}'_i$. Let the order of the vectors encountered from $\boldsymbol{E}'_i$ be $\overrightarrow{e_1}, \ldots, \overrightarrow{e_q}$.

We claim that there exist $\lfloor q/2 \rfloor$ vectors in $\boldsymbol{E}'_i$ with total length at most L/r. Consider a pair of consecutive vectors e_j, e_{j+1}. By Lemma 4.1, the distance between the originating points of $\overrightarrow{e_j}$ and $\overrightarrow{e_{j+1}}$ is longer than r times the length of the shorter of the two vectors. Hence the distance along TSC between the originating points of $\overrightarrow{e_j}$ and $\overrightarrow{e_{j+1}}$ is also longer than r times the length of the shorter vector. We may charge the length of the shorter vector to the length of the path between the originating points along TSC. The charge is at most $1/r$ times the length of the path. Taking the $\lfloor q/2 \rfloor$ disjoint consecutive pair of vectors, $\{\overrightarrow{e_1}, \overrightarrow{e_2}\}, \{\overrightarrow{e_3}, \overrightarrow{e_4}\}, \ldots$, and charging the shorter vector of each pair to the correspondent path in TSC between the originating points, each segment of the circuit TSC is charged at most once. Thus, the total length of the shorter vectors of the chosen pairs is at most L/r.

Now we consider only edges in $\boldsymbol{E}'_i$ that have no been charged yet, and repeat the same process. After $O(\log n)$ steps, each edge in $\boldsymbol{E}'_i$ is charged, giving the bound of $O(\log n \cdot L)$ for the total weight of the edges in $\boldsymbol{E}'_i$.

Since k is constant (dependent only on d, t, and N), we conclude that $wt(\boldsymbol{E}')$ is bounded by $O(\log n \cdot L) = O(\log n) \cdot wt(MST)$. □

Remark: The bound on the maximum degree that can be obtained from Theorem 4.2 is not tight in the following sense. Since the larger the angle θ the smaller the bound on the degree, we want a large angle to be used in (5). For the purpose of obtaining a better bound on the degree, we can redo Lemma 4.1 using $r = 0$. (This value of r is not convenient in Theorem 4.3.) We would obtain θ equal to $\arctan \frac{k_1(t-1)}{k_2(t+1)}$ instead of $\arctan \frac{k_1(t-1)}{k_2(2t+1)}$.

Implementation and time analysis: For the case of geometric graphs, by Theorem 1.2 the number of edges added by the Greedy Algorithm is $O(n)$. Doing the same type of analysis as for arbitrary graphs, we get a total running time of $O(n^3 \log n)$.

4.2 The Transformational Method

The following lemma presents, for complete graphs from *an arbitrary* metric space, a general technique for converting spanners of small size and arbitrary weight into spanners of small size and small weight.

Lemma 4.4 *Let M be a metric space such that for every n-vertex complete graph G on this metric space,*

1. *there exists an $O(g(n))$ time algorithm $\mathcal{A}$ which builds a t-spanner for G with $O(f(n))$ edges, where f and g are functions such that for every $m \geq 1$, $f(m)/2 \geq f(\lfloor m/2 \rfloor)$, and $g(m)/2 \geq g(\lfloor m/2 \rfloor)$.*

2. *there exists an $O(h(n))$ time algorithm $\mathcal{B}$ which builds a spanning tree T for G, such that $wt(T) = O(1) \cdot wt(MST)$.*

Then there exists an $O(\max\{g(n), h(n), n \log n\})$ time algorithm $\mathcal{A}'$ which builds, for every complete graph in M, for every $\epsilon > 0$, a $(t+\epsilon)$-spanner with $O(f(n))$ edges and weight $O\left(\frac{f(n)}{n} \log n\right) \cdot wt(MST)$.

The algorithm $\mathcal{A}'$: The algorithm $\mathcal{A}'$ works as follows. Let $G = (V, E)$ be an n-vertex graph on M. Let T be a spanning tree for G built by algorithm $\mathcal{B}$. Consider the Hamiltonian path P built from T, as is done in Section 2. Let $L = wt(P)$. Note that $L \leq 2 \cdot wt(T) = O(wt(MST))$.

Let $V_0 = V$, and for $j = 1, \ldots, \log n$, define $V_j \subset V$ as follows: Let $a = 2^{j-1}\frac{L}{n}$. Divide L into $n_j = \frac{cL}{a} = \frac{cn}{2^{j-1}}$ intervals of length a/c each, where $c = \frac{2(t+1)}{\epsilon}$. These intervals induce a partition of the vertices in P. Define V_j to be a set containing exactly one vertex arbitrarily chosen from each non-empty set of the partition. Call the vertex chosen the *representative* of the vertices in the set. Note that the distance in P (and hence in T) between every two vertices in the same set of the partition is at most a/c. Let G_j be the induced subgraph on V_j.

$\mathcal{A}'$ construct the $(t+\epsilon)$-spanner in phases. For $j = 0, 1, \ldots, \log n$, consider the t-spanner for G_j built by $\mathcal{A}$, and call this $G'_j = (V_j, E'_j)$. Note that, by hypothesis, $|E'_j|$ is $O(f(|V_j|))$. For $j = 0, 1, \ldots, \log n$, let $E''_j = \{e \in E'_j | wt(e) \leq 2^j \frac{L}{n} \cdot t \cdot (1 + \frac{1}{c})\}$. Let $E'' = T \cup E''_0 \cup E''_1 \cup \cdots \cup E''_{\log n}$ and let $G'' = (V, E'')$.

Claim 4.5 *G'' is a spanner for G with the desired properties.*

Proof : Omitted from this version.

Lemma 4.6 *$\mathcal{A}'$ takes $O(\max\{g(n), h(n), n \log n\})$ time.*

Proof : The spanning tree T can be built in time $O(h(n))$. The Hamiltonian path P can be obtained from T in time $O(n)$. Each of the sets V_j can be built in time $O(n)$ by following edges in P, so the time required to get all the V_j's is $O(n \log n)$.

The time required to obtain all the t-spanners, using algorithm $\mathcal{A}$ repeatedly is

$$\begin{aligned} &= O(|V|) + O(g(|V_0|)) + \sum_{j=1}^{\log n} O(g(|V_j|)) \\ &= O(g(n)) + \sum_{j=1}^{\log n} O(g(\frac{n}{2^{j-1}})) \\ &= O(g(n)). \end{aligned}$$

We have used $g(m)/2 \geq g(\lfloor m/2 \rfloor)$ to obtain the last inequality.

Finally, since there are a total of $O(f(n))$ edges and $f(n)$ is clearly no bigger than $g(n)$, the total time to do the union of the spanning tree and all the spanners to obtain E'' is $O(g(n))$. $\square$

This completes the proof of Lemma 4.4. $\square$

Applying the transformation method to geometric graphs: Let G be an n-vertex geometric graph in $\mathbb{R}^d$ under norm L_p. There are known $O(n \log n)$ time algorithms for constructing t-spanners for G (for $t > 1$) with $O(n)$ edges [21, 24]. Also, using $O(n \log n)$ time algorithms from [21, 25], we can construct spanning trees whose weight is within a factor of 2 of the optimal. Using lemma 4.4 and choosing an appropriately small ϵ, we see that in time $O(n \log n)$ we can build t-spanners for geometric graphs under an L_p norm, which satisfy the conditions of Theorem 1.2.

5 Open Problems

1. Can the gap between the upper and lower bounds for geometric graphs (a factor of $O(\log n)$) be improved? Does the Greedy Algorithm produce t-spanners with weight $\Omega(\log n) \cdot wt(MST)$ for some family of geometric graphs, or can the upper bound be improved by tighter analysis ?

2. For arbitrary graphs, do there always exist $(\log n)$-spanners with weight $O(wt(MST))$ i.e. can we generalize the result for unit edge-weighted graphs from [15].

3. Is there a more efficient implementation of the greedy algorithm, both for arbitrary graphs and for geometric graphs?

4. Find a polynomial time algorithm $\mathcal{A}$ to build t-spanners such that $\mathcal{A}$ has "performance guarantees," i.e., for every graph G and every t, $\mathcal{A}$ builds a t-spanner G' of G such that $size(G') \leq c \cdot e_{opt}$, $[wt(G') \leq c \cdot wt_{opt}]$ where c is a constant and e_{opt} $[wt_{opt}]$ is the minimum number of edges [minimum weight] over all t-spanners of G.

5. What is the complexity of the following problem:
Instance: A geometric graph $G, t, m > 1$.
Question: Does G have a t-spanner with at most m edges?
This is an open problem even if G is a 2-dimensional Euclidean graph. We observe that Peleg and Schäffer [15] have proved that this problem is NP-complete in the case that G is a unit edge-weighted graph.

Acknowledgement: The first and fourth authors would like to thank Laci Babai for his constant encouragement and helpful suggestions.

References

[1] Althöfer, I.; Das, G.; Dobkin, D; Joseph, D.; Soares, J., Generating sparse spanners for weighted graphs, *Discrete and Comp. Geometry*, to appear.

[2] Awerbuch, B., Complexity of network synchronization, *JACM* (1985), 804-823.

[3] Awerbuch, B.; Peleg, D., Sparse partitions, *FOCS* (1990), 503-513.

[4] Bendelt, H.; Dress, A., Reconstructing the shape of a tree from observed dissimilarity data, *Advances in Appl. Math.*, **7** (1986), 309-343.

[5] Bhatt, S.; Chung, F; Leighton, F.; Rosenberg, A., Optimal simulations of tree machines, *27th IEEE Symposium on the Foundations of Computer Science,* Toronto (1986) 274-282.

[6] Bollobás, B., *Extremal graph theory*, Academic Press, London, 1978.

[7] Chew, P., There is a planar graph almost as good as the complete graph, *ACM Symposium on Computational Geometry* (1986), 169-177 .

[8] Das, G., Approximation schemes in computational geometry, PhD Thesis, CS Dept, Univ of Wisconsin-Madison, 1990.

[9] Das, G.; Joseph, D., Which triangulations approximate the complete graph? *International Symposium on Optimal Algorithms* (1989), (LNCS, Springer-Verlag).

[10] Dijkstra, E.W., A note on two problems in connection with graphs, *Numerische Mathematik*, **1** (1959), 269-271 .

[11] Dobkin, P.D.; Friedman, S.J.; Supowit, K.J., Delaunay graphs are almost as good as complete Graphs, *Discrete Comp. Geom.*, **5** (1990), 399-407.

[12] Keil, J.M., Approximating the complete Euclidean graph, *SWAT* (1988), (LNCS, Springer-Verlag).

[13] Keil, J.M.; Gutwin, C.A., Classes of graphs which approximate the complete Euclidean graph, *Discrete Comp. Geom.* **7**, (1992), 13-28.

[14] Levcopoulos, C.; Lingas, A., There are planar graphs almost as good as the complete graphs and as short as the minimum spanning trees, *Symposium on Optimal Algorithms* (1989), 9-13, (LNCS, Springer-Verlag).

[15] Peleg, D.; Schäffer. A., Graph spanners, *Journal of Graph Theory,* **13** (1989), 99-116

[16] Peleg, D.; Ullman, J., An optimal synchronizer for the hypercube, *SIAM J. Comp.*, **18** (1989), 740-747.

[17] Peleg, D.; Upfal, E., A tradeoff between space and efficiency for routing tables, *STOC* (1988), 43-52.

[18] Rankin, R.A., The closest packing of spherical caps in n dimensions, *Proc. Glasgow Math. Assoc.* **2** (1955), 139-144.

[19] Rogers, C.A., Covering a sphere with spheres, *Mathematika* **10** (1963), 157-164.

[20] Ruppert, J.; Seidel, R., Approximating the d-dimensional complete Euclidean graph, *Canadian Conference on Computational geometry* (1991), 207-210.

[21] Salowe, J.S., Construction of multidimensional spanner graphs with applications to minimum spanning trees, *ACM Symposium on Computational Geometry* (1991), 256-261.

[22] Soares, J., Approximating Euclidean distances by small degree graphs, *Technical Report 92-05,* University of Chicago, (1992).

[23] Soares, J., Spanners, *Unpublished Manuscript*, 1991.

[24] Vaidya, P.M., A sparse graph almost as good as the complete graph on points in K dimensions, *Discrete and Comp. Geom.* **6**, (1991), 369-381.

[25] Vaidya, P.M., Minimum spanning trees in k-dimensional space, *SIAM J. Comput.* **17** (1988), 572-582.