GENERATING SPARSE SPANNERS FOR WEIGHTED GRAPHS

by

Ingo Althöfer, Gautam Das,
David Dobkin & Deborah Joseph

Computer Sciences Technical Report #882

October 1989

GENERATING SPARSE SPANNERS FOR WEIGHTED GRAPHS [1]

INGO ALTHÖFER – Fakultät fur Mathematik, Universität Bielefeld

GAUTAM DAS – Department of Computer Sciences, University of Wisconsin

DAVID DOBKIN – Department of Computer Science, Princeton University

DEBORAH JOSEPH – Department of Computer Sciences, University of Wisconsin

---

ABSTRACT. Given a graph G, a subgraph G' is a t-spanner of G, if for every $u,\ v\ \in\ V$, the distance from u to v in G' is at most t times longer than the distance in G. In this paper we give a very simple algorithm for constructing sparse spanners for arbitrary weighted graphs. We then apply this algorithm to obtain specific results for planar graphs and Euclidean graphs. We discuss the optimality of our results and present several nearly matching lower bounds.

---

1. INTRODUCTION.

Let $G\ =\ (V,\ E)$ be a connected n-vertex graph with arbitrary positive edge weights. A subgraph $G'\ =\ (V,\ E')$ is a *t-spanner* if, between any pair of vertices the distance in G' is at most t times longer than the distance in G. The value of t is the *stretch factor* associated with G'. We consider the problem of determining t-spanners for graphs where the spanners are sparse and t is a constant independent of the size of the graph. Sparsity will be measured according to two criteria. Let $Weight(G)$ denote the sum of all edge weights of graph G, and $Size(G)$ denote the number of edges. A graph is sparse in *size* if it has few edges. Similarly, a graph is sparse in *weight* if its total edge weight is small. Our results separate graphs into classes where spanners with linearly many edges achieve constant stretch factors, and classes where a non-linear number of edges are necessary.

Problems of this type appear in numerous applications. Spanners appear to be the underlying graph structure in various constructions in distributed systems and communication networks [Aw, PUl, PU]. They also appear in biology in the process of reconstructing phylogenetic trees from matrices, whose entries represent genetic distances among contemporary living species [BD]. Robotics researchers have studied spanners

---

[1] The work of the second and fourth authors was supported by NSF PYI grant DCR-8402375. The work of the third author was supported by NSF grant CCR-8700917.

This paper will be presented at the Second Scandinavian Workshop on Algorithm Theory, Bergen, Norway, 1990, and proceedings will be published in the Lecture Notes in Computer Science, Springer-Verlag. The paper also appears as Princeton University Technical Report CS-TR-261-90.

under the constraints of *Euclidean* geometry, where vertices of the graph are points in space, and edges are line segments joining pairs of points [C, DFS, DJ, K, KG, LL].

In the above applications, previous research has focussed on graphs with specific constraints. In distributed computation, the design of synchronizers [Aw, PUl] and the design of succinct routing tables [PU] implicitly generated spanners for graphs with *unit edge weights.* For example, in designing routing tables [PU], the routes follow the edges of a sparse spanner. For any stretch factor $O(t)$, the size of these spanners are $O(n^{1 + 1/t})$. These designs are based upon a *clustering algorithm*, which is more complex than the algorithm in this paper. Moreover, it is not easy to generalize the clustering algorithm to graphs with arbitrary edge weights. Recently the problem of designing succinct routing tables has been considered for graphs with arbitrary edge weights [ABLP, AP]. For any t, the scheme in [ABLP] routes along paths which are at most $O(t^2 9^t)$ longer than the shortest paths, while the total memory required for the tables is $O(tn^{1 + 1/t} logn)$. In [AP], the routes are $O(t^2)$ longer, while the memory required is $O(n^{1 + 1/t} log^2 n log D)$, where D is the *diameter* of the graph. Spanners have been considered for special classes of graphs in [PS], however these graphs have unit edge weights.

In all the above research, sparseness has been achieved in the size of spanners, but not in the weight. In robotics, graphs with varying edge weights have been examined, but the weights are Euclidean distances and not arbitrary. Because of this restriction, it has been possible to construct linearly sized spanners, unlike the above examples. For instance, the *Delaunay* and other triangulations approximate complete straight-edged graphs on the plane [C, DFS, DJ, LL, KG]. A few papers have considered weight sparseness of spanners for these graphs [DJ, LL]. For any t, there exist spanners for the complete graph on the plane with stretch factor $O(t)$, and weight within a $O(1 + 1/t)$ multiple of the weight of the minimum spanning tree. Weight sparseness has also been considered in [A, D, S] for the special case of 1-spanners under a general model where spanners may have auxiliary vertices.

In this paper we approach the problem from a very broad perspective. Our graphs have no special embeddings, and we allow arbitrary positive edge weights. For any t, we show that every such graph has a spanner with $O(t)$ stretch factor, and $O(n^{1 + 1/t})$ size. We also provide weight bounds for our spanners. The contributions of this paper are: a very simple polynomial time algorithm for constructing sparse spanners (in *both* size and weight) of arbitrary weighted graphs, using these ideas for constructing spanners of *planar* graphs, some lower bound results, and some results on spanners in Euclidean spaces of arbitrary dimensions and norms. Since any spanner with appropriate sparseness and stretch factor can be used for constructing synchronizers [PUl], our algorithm provides a simple alternative to the clustering algorithm previously used for constructing synchronizers. Similarly, we hope that our results will simplify the construction of succinct routing tables for arbitrary weighted graphs.

The next section describes how to construct sparse spanners for general as well as planar graphs with arbitrary edge weights. Section 3 deals with several lower bound

results. Section 4 discusses spanners for Euclidean graphs. We conclude with some open problems. Due to lack of space, we omit details of some of the proofs, which may be found in the full paper [ADDJ].

2. CONSTRUCTION OF SPARSE SPANNERS.

Let G be a n-vertex, connected, weighted graph. The minimum spanning tree of G, denoted $MST(G)$, is obviously the sparsest spanner, however, it is not hard to see that its stretch factor can be as bad as $\Omega(n)$. Instead, we would like to look for spanners whose stretch factors are constants, independent of n. Our results are encouraging, because we show that any graph has spanners with constant stretch factors, whose sparseness can be made arbitrarily close to that of $MST(G)$. They are summarized by the following two theorems.

Theorem 1: Given a n-vertex graph G and a $t \geq 1$, there is a polynomially constructible $(2t + 1)$-spanner G' such that

1) $Size(G') < n \cdot \lceil n^{1/t} \rceil$,
2) $Weight(G') < Weight(MST(G)) \cdot (1 + n/2t)$.

Note that the stretch factor is independent of the number of vertices in the graph, and of the edge weights. Thus, even for a dense graph with $\Omega(n^2)$ edges, arbitrarily sparse spanners exist which have good stretch factors. Our lower bound results will show that the size bound is almost tight, though the weight bound is loose. Previous research [Aw, PS] has produced spanners with the same size and stretch factors, but the graphs considered there were unweighted, and sparseness was measured by size only. Also, our algorithm is simpler than the clustering algorithm employed. Recently the problem of designing succinct routing tables has been considered for graphs with arbitrary edge weights [ABLP, AP], again based on the clustering algorithm. For any t, the scheme in [ABLP] routes along paths which are at most $O(t^2 9^t)$ longer than the shortest paths, while the total memory required for the tables is $O(tn^{1 + 1/t} logn)$. In [AP], the routes are $O(t^2)$ longer, while the memory required is $O(n^{1 + 1/t} log^2 n log D)$, where D is the *diameter* of the graph.

Theorem 2: Given a n-vertex planar graph G and a $t \geq 1/2$, there is a polynomially constructible $(2t + 1)$-spanner G' such that

1) $Size(G') \leq (n - 2) \cdot (1 + 2/\lfloor 2t \rfloor)$,
2) $Weight(G') < Weight(MST(G)) \cdot (1 + 1/t)$.

Here we again observe that the stretch factor is independent of the number of vertices in the planar graph, or the edge weights. Thus, arbitrarily sparse spanners exist which have good stretch factors. (Note that the maximum size of a planar graph can be $3n - 6$, so in effect we demonstrate that even the multiplicative constant in the size can be reduced). Theorem 2 is stronger in flavor than our first theorem, because our lower bound results will show that both size and weight bounds are tight. We later give an interesting application of this in connection with Euclidean graphs.

Before we prove our results, we introduce an algorithm for constructing spanners. It constructs a sparse subset of edges so that a required stretch factor is achieved. The algorithm, called $SPANNER(G,\ r)$, takes as input a weighted graph G, and a positive parameter r. The weights need not be unique. It produces as output a subgraph G'.[2]

Algorithm $SPANNER(G\ =\ (V,\ E),\ r)$;

begin

Sort E by nondecreasing weight;
$G'\ :=\ \phi$;

for every edge $e\ =\ [u,\ v]$ in E **do**
begin
Compute $P(u,\ v)$, the shortest path from u to v in G';
if $(r \cdot Weight(e)\ <\ Weight(P(u,\ v)))$ **then** add e to G';
end;

Output G';

end;

While our algorithm is very simple, the subgraph it generates has many interesting properties. The following lemmas describe these properties.

Lemma 1: G' is a r-spanner of G.

Proof. Consider any edge $[u,\ v]$ in $G\ -\ G'$. At the instant it is examined by the algorithm, there is a path $P(u,\ v)$ from u to v in the current graph of length $\leq\ r \cdot Weight([u,\ v])$. Thus, in the final output each deleted edge $[u,\ v]$ is associated with a short path $P(u,\ v)$. Now consider any shortest path in G of length l. For every deleted $[u,\ v]$ edge along this path we replace by $P(u,\ v)$, and thus obtain an alternate path in G' of length $\leq\ l \cdot r$, which proves the lemma. •

Lemma 2: Let C be any simple cycle in G'. Then $Size(C)\ >\ r\ +\ 1$.

Proof. Assume that a cycle C remained with size $\leq\ r\ +\ 1$. Let $[u,\ v]$ be the last edge in C to be examined by the algorithm. Clearly, it is one of the (possibly many) edges in the cycle with the largest weight. When $[u,\ v]$ is being considered by the algorithm, there is a clearly path from u to v (the remaining portion of the cycle) whose length is $\leq\ r \cdot Weight([u,\ v])$. Thus, $[u,\ v]$ should not have been added, which is a contradiction. •

Lemma 3: Let C be any simple cycle in G', and let e be any edge in C. Then $Weight(C\ -\ \{e\})\ >\ r \cdot Weight(e)$.

---

[2] We understand that this algorithm has been independently discovered by Bern, [Be].

Proof. Assume that a cycle C remained which violates the above weight condition for some edge e in C. Clearly, it violates this condition for the last edge in C to be examined by the algorithm, say $[u, v]$. As in the above lemma, it is one of the (possibly many) edges in the cycle with the largest weight. Thus, the remaining portion of the cycle has length $\leq r \cdot Weight([u, v])$. Thus, $[u, v]$ should not have been added, which is a contradiction. •

Lemma 4: $MST(G)$ is contained in G'.

Proof. Before we prove this lemma, we observe that our algorithm is essentially a generalized minimum spanning tree algorithm, because for an infinite r, it outputs $MST(G)$. In fact, for an infinite r its behavior is exactly like Kruskal's minimum spanning tree algorithm [T].

Our proof (sketch) is by induction on the order in which edges are examined. Let the sequence $\phi = G'_0, G'_1, \cdots, G'_{|E|} = G'$ represent the growth of G', where G'_i represents the partially constructed subgraph after the i^{th} edge has been examined. Now let us consider Kruskal's algorithm, which also examines edges by increasing weight. In this case, let the sequence $\phi = M_0, M_1, \cdots, M_{|E|} = MST(G)$ represent the growth of the minimum spanning tree, where M_i represents the partially constructed tree after the i^{th} edge has been examined. It is not hard to prove by induction that, for all i, M_i is contained in G'_i, which in turn proves the lemma. •

The algorithm is clearly polynomial and easy to implement. In proving the two theorems, Lemma 2 will be used in proving size sparseness, while Lemma 3 and Lemma 4 will be used in proving weight sparseness of the resulting subgraphs.

We still need two more lemmas. Let the *size* of a face of a planar graph be the number of edges encountered while traversing the face (with repetitions allowed). Lemma 5 bounds the size of a planar graph, given a minimum face size.

Lemma 5: If all the faces of an n-vertex planar graph G have sizes $\geq r$, then $Size(G) \leq (n - 2) \cdot (1 + 2/(r - 2))$.

Proof. Let m be the size of the graph. Euler's formula for planar graphs states that $n - m + f = 2$, where f is the number of faces in the graph. Thus, $f = m + 2 - n$. If we traverse each face and mark the edges encountered, every edge in the graph will eventually be marked twice. We thus have $f \cdot r \leq 2m$. Substituting for f from above and simplifying, we have $m \leq (n - 2) \cdot (1 + 2/(r - 2))$, which proves the lemma. •

Our next lemma is from extremal graph theory (which is derivable from Theorem 3.7, Chapter III, [B]). Let the *girth* of a graph be the size (number of edges) of its smallest simple cycle. This lemma provides an upper bound on the size of a graph with a given girth.

Lemma 6: Let G be an n-vertex graph with girth $> r$. Then $Size(G) < n \cdot \lceil n^{2/(r-1)} \rceil$.

We first prove Theorem 2, then use some of the ideas in proving Theorem 1.

Proof of Theorem 2. We run the *SPANNER* algorithm on a given n-vertex planar graph G and a $t \geq 1/2$, after setting $r = 2t + 1$. By Lemma 1, the resulting graph has stretch factor $2t + 1$. Also, by Lemma 2, the girth of the output is $> 2t + 1$, that is, $\geq \lfloor 2t + 2 \rfloor$. This clearly implies that the minimum face size is $\geq \lfloor 2t + 2 \rfloor$. Thus, by Lemma 5, the size of the output is $\leq n \cdot (1 + 2/\lfloor 2t \rfloor)$. In the next section we will show that this size bound is tight.

We now prove the weight bound, which requires a different approach (the outline is similar to the method in [LL]). By Lemma 4, the output subgraph has to contain $MST(G)$. Consider a planar drawing of the subgraph, with $MST(G)$ embedded in the subgraph. If we walk around the tree (counting each edge twice), our path will resemble a "skinny" polygon with perimeter being $2 \cdot Weight(MST(G))$. Our accounting strategy will be to "grow" this polygon outwards by absorbing neighboring edges of the subgraph, until it becomes the outer face of the graph. At any stage, an edge is selected which, along with a portion of the polygon's current boundary circumscribes a face adjacent to the polygon. Consider Stage i. Let the length of the polygon be W_i (with $W_0 = 2 \cdot Weight(MST(G))$). Let T_i be the total length of all non minimum spanning tree edges encountered so far. Thus $T_0 = 0$. Let the edge selected to be added be $[u, v]$, and the portion of the polygon's boundary from u to v be $P(u, v)$. By Lemma 3, $Weight([u, v]) < (1/(2t + 1)) \cdot Weight(P(u, v))$. Thus,

1) $W_{i+1} < W_i - Weight([u, v]) \cdot (1/(2t + 1) - 1)$,
2) $T_{i+1} = T_i + Weight([u, v])$.

We state without proof that T converges to at most $Weight(MST(G)) \cdot (1/t)$. Thus $Weight(G') < Weight(MST(G)) \cdot (1 + 1/t)$. We later show that this weight bound is tight because there are graphs for which one can do no better. •

Proof of Theorem 1. We are given an n-vertex graph G and a $t \geq 1$ as input. We set $r = 2t + 1$ and run the *SPANNER* algorithm. By Lemma 1, the resulting graph has stretch factor $\leq 2t + 1$. Also, by Lemma 2, the girth of the output is $> 2t + 1$. Thus, by Lemma 6, the size of the output is $< n \cdot \lceil n^{1/t} \rceil$. In the next section we will show that this size bound is quite tight because there exist graphs for which one cannot do much better.

We now prove the weight bound. By Lemma 4, $MST(G)$ is contained in the subgraph. For each vertex v, consider the graph G_v, composed of $MST(G)$ and the edges in G' incident to v and not in $MST(G)$. Let the set of latter edges be denoted as E_v. It is easy to see that this graph is planar. Hence by methods similar to the proof of Theorem 2, we can show that $Weight(E_v) < Weight(MST(G)) \cdot 1/t$. Thus $\Sigma_{v \in V} Weight(E_v) < Weight(MST(G)) \cdot n/t$. In the above summation each edge in the E_v's have been counted twice, thus $Weight(G') < Weight(MST(G)) \cdot (1 + n/2t)$, which proves the weight bound. In the next section we shall see that this bound is not as tight as our other results. •

3. LOWER BOUNDS.

In this section we will show various lower bounds for spanners of graphs.

Our first result concerns general graphs with arbitrary positive weights. In [PS], using results from extremal graph theory [B], it has already been shown that, for every $t \geq 1$, there exist infinitely many n-vertex graphs with unit edge weights such that every $(2t+1)$-spanner requires $\Omega(n^{1+1/(2t+3)})$ edges. Thus the size result of Theorem 1 is tight up to a constant factor in the exponent of n. The above lower bound gives a corresponding lower bound for weights of spanners, that is, for every $t \geq 1$, there exist infinitely many n-vertex weighted graphs G such that every $(2t+1)$-spanner has weight $\Omega(Weight(MST(G)) \cdot n^{1/(2t+3)})$. Thus the weight result of Theorem 1 is not as tight as the size result.

Our next result concerns planar graphs with arbitrary positive weights.

Theorem 3: For infinitely many n and t, there exists a n-vertex planar graph G with unit edge weights such that every $(2t+1)$-spanner requires $\Omega(n \cdot (1+1/t))$ edges, and has weight $\Omega(Weight(MST(G)) \cdot (1+1/t))$.

Proof. For infinitely many n and t, we state without proof that it is possible to construct a planar n-vertex graph with unit edge weights, such that each face is a regular $(2t+4)$-gon, and such that the girth is also $\geq 2t+4$. From Euler's formula it can be derived that the size of this graph is $\Omega(n \cdot (1+1/t))$. Clearly any proper subgraph will have stretch factor $\geq 2t+3$, and the result follows. •

This shows that the size and weight results of Theorem 2 are tight.

In what follows we shall state lower bound results for various generalizations of spanners. Consider graphs with arbitrary positive weights. Let $Dist(u, v, G)$ be the distance from vertex u to vertex v in graph G. Let $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ be two graphs with V_1 a subset of V_2. G_2 is called a *generalized* t-spanner of G_1 if, for all $u, v \in V_1$, $Dist(u, v, G_1) \leq Dist(u, v, G_2) \leq t \cdot Dist(u, v, G_1)$. Thus, the spanner is not simply a subgraph of G_1, rather it may contain auxiliary vertices and edges. An important point is, the spanner is not allowed to "cheat", that is, paths in G_2 are never shorter than those in the original graph G_1, though they may use auxiliary vertices and edges.

For special graphs generalized spanners can be substantially smaller than simple spanners. For example, consider the complete n-vertex graph with unit edge weights. Clearly every simple 1-spanner requires all edges. But the *star graph* with one auxiliary vertex attached to all n original vertices via n additional edges of weight $1/2$ is a generalized 1-spanner with only n edges. However, the following lower bound result shows that such constructions are not always possible for all graphs. In fact, there exist graphs such that generalized spanners cannot be much smaller than simple spanners.

Theorem 4: For infinitely many n and t, there exists a n-vertex graph G with unit edge weights such that every generalized t-spanner requires $\Omega(\frac{1}{logn} \cdot n^{1+1/(t+2)})$ edges.

Before we prove the theorem, we need some definitions and lemmas. Let $t \geq 1$, and m be a positive integer. Let g be a function that maps each unweighted n-vertex graph to a string of m bits. We say g is an (n, t, m)-*compressor* if the following is true. For all pairs of graphs G_1, G_2, if $g(G_1) = g(G_2)$, then for all pairs of vertices u, v, $Dist(u, v, G_1) \leq t \cdot Dist(u, v, G_2)$. Informally, the compressor is just like a hash function. It partitions all graphs into groups, such that in each group the graphs have approximately the same distances between any given pair of vertices. Clearly, as t increases, it should be possible to construct compressors with fewer groups. The following lemma provides a lower bound on the number of groups, given any t.

Lemma 7: For an (n, t, m)-compressor to exist, $m \geq 1/4 \cdot n^{1 + 1/(t+2)}$.

Proof. For infinitely many n and t, there exist n-vertex graphs with girth $\geq t + 2$ and at least $1/4 \cdot n^{1 + 1/(t+2)}$ edges [B, pp 104]. Let $G = (V, E)$ be such a graph, and let G_1, G_2 be two different subgraphs of G. By the girth condition of G, both subgraphs have to belong to different groups of the compressor. Thus the total number of groups is $\geq 2^{|E|}$, thus $m \geq |E|$. •

Two further observations will be useful in proving the theorem. First, existence of spanners (even generalized spanners) for all graphs implies the existence of a compressor. For example, consider a collection of t-spanners, one for each n-vertex graph. Let the set of spanners (without repetitions) be $\{G'_1, \cdots, G'_k\}$. Then, setting $g(G) = binary(i)$ if G'_i is a spanner of G, yields an $(n, t, \lceil logk \rceil)$-compressor.

Second, we show how to encode an unweighted graph as a bit string. Every n-vertex graph with m edges can be encoded by $2 \cdot \lceil logn \rceil \cdot m$ bits, by representing every edge as a pair of $\lceil logn \rceil$ bit strings, each in turn representing a vertex.

Proof of Theorem 4. We shall first prove the theorem for unweighted spanners, then sketch the proof for weighted spanners. To prove the first part, consider any collection of t-spanners, one for each n-vertex graph. Since we are interested in sparse generalized spanners of n-vertex graphs, even with auxiliary vertices these spanners should have no more than $n + 2 \cdot n(n-1)/2 = n^2$ vertices. Let m be the size of the largest spanner in this set. Each spanner can be encoded in $\leq 4 \cdot \lceil logn \rceil \cdot m$ bits. Thus by our previous observation, these spanners imply the existence of an $(n, t, 4 \cdot \lceil logn \rceil \cdot m)$-compressor. But by Lemma 7, $4 \cdot \lceil logn \rceil \cdot m \geq 1/4 \cdot n^{1 + 1/(t+2)}$. Solving for m proves the theorem.

To prove the second part, the idea is to encode the edge weights of the spanners as bit strings. The problem arises because weights could be arbitrarily large, or require many precision bits. However, it can be shown that by ignoring very large weights, and rounding off the remaining in a certain way, we can get suitably encodable spanners with slightly larger stretch factors. •

Note that we have no lower bounds on the weight sparseness of generalized spanners. We now generalize spanners in another direction. Let $t \geq 1$. $G' = (V, E')$ is a t-*approximator* of $G = (V, E)$ if, for all $u, v \in V$, $1/t \cdot Dist(u, v, G) \leq Dist(u, v, G') \leq t \cdot Dist(u, v, G)$. The following theorem (without proof) provides a lower bound on the size of approximators.

Theorem 5: For infinitely many n and t, there exists a n-vertex graph G with unit edge weights such that every t-approximator has $\Omega(\frac{1}{logn} \cdot n^{1 + 1/(t+2)})$ edges.

We finish this section by stating a lower bound for *complete* graphs with weights that are *proper*. The latter mean that the weight of any edge is never more than the length of any path between its end vertices.

Theorem 6: For infinitely many n and t, there exists a n-vertex complete graph G with proper edge weights, such that every $(2t - 1)$-spanner requires $\Omega(1/t \cdot n^{1 + 1/(2t+1)})$ edges.

All the above lower bound results emphasize robustness, that is, they hold even after allowing more general spanners, or more restrictive graphs. In the next section we discuss spanners of Euclidean graphs.

4. SPANNERS IN EUCLIDEAN GRAPHS.

In Euclidean graphs, the weights assigned to the edges are not arbitrary, hence the lower bounds of Section 3 do not apply. It has been possible to construct spanners for such graphs with a linear number of edges.

Our first result is on Euclidean graphs on the plane. The vertices of such a graph are a set V of n points on the plane. The edges are line segments joining pairs of vertices and each edge weight is the Euclidean distance (measured in some norm, $||\cdot||$) between the vertices. These graphs may be either planar or nonplanar. We let $K(V)$ be the complete Euclidean graph (which is clearly nonplanar). We pose the question: are there sparse *planar* spanners of $K(V)$? This problem has been extensively studied in the past. In [C, DFS] it was shown that *Delaunay triangulations* in the $||\cdot||_1$ and $||\cdot||_2$ norms are spanners with constant stretch factors. Using a general framework, [DJ] showed that other planar graphs such as *greedy triangulations* and *minimum weight triangulations* also have constant stretch factors (with different constant values). In [KG] the stretch factor for Delaunay triangulations in the $||\cdot||_2$ norm was improved to 2.42 (current best). In [DJ, LL] it was shown that there exist extremely short Euclidean planar graphs (that is, almost as short as the minimum spanning tree) that have constant stretch factors. The algorithm in [LL] produces arbitrarily short graphs based upon a parameter, though it is not obvious how small are the *sizes* of these graphs. Using Theorem 2 we can produce planar spanners that are both short, as well as small, in size.

Let $t \geq 1/2$, and consider the Delaunay triangulation over V in the $||\cdot||_2$ norm. It is known that this triangulation contains $MST(K(V))$. We now apply the *SPANNER* algorithm on the triangulation with $r = 2t + 1$. Because spanners are transitive, it is easy to see that the output (denoted as G') satisfies the following property.

Theorem 7: G' is a $(2.42) \cdot (2t + 1)$-spanner of $K(V)$, such that
1) $Size(G') \leq (n - 2) \cdot (1 + 2/\lfloor 2t \rfloor)$,
2) $Weight(G) < Weight(MST(K(V))) \cdot (1 + 1/t)$.

In higher dimensions planarity of spanners is not an issue. Our next result is on constructing linear sized spanners for complete graphs of point sets V in $(R^d, ||\cdot||)$, for all dimensions $d \geq 2$, and all norms $||\cdot||$. Fix some angle $\delta > 0$. The key idea is to cover R^d by finitely many open cones $C_1, ..., C_{s(\delta)}$, all with the same focus at the origin O, such that for all points u, v in the same cone, $\angle\, u\, O\, v < \delta$. Such a covering exists for every d and every $||\cdot||$ by the theorem of Heine-Borel [CS]. Similar ideas for constructing spanners for fixed norms have been considered in [K].

We construct the spanner G' as follows. For every $v \in V$, consider the covering of R^d by the cones $C_1 + v, ..., C_{s(\delta)} + v$, where $C + v$ represents a shifting of the cone C to a new origin v, in the spirit of Minkowski. For every cone $C_i + v$, let u be the vertex in the cone such that $||u - v||$ is minimized. We add $[u, v]$ to G', and u is known as the i^{th} *neighbor* of v.

Clearly, the size of G' is $\leq s(\delta) \cdot n$, and is therefore linear. It remains to show that its stretch factor is small. Consider $u, v \in V$. A short path between them is constructed in the following way. Let u be inside the cone $C_i + v$. Go from v to its i^{th} neighbor, and proceed from there in the same way. We show that this path is not too long with respect to $||v - u||$.

Lemma 8: For every $\epsilon < 1/(2 + \sqrt{2})$, there is an angle $\delta(\epsilon) > 0$ such that $Distance(v, u, G') < ||v - u||/(1 - \epsilon(2 + \sqrt{2}))$.

The estimation of the angle $\delta(\epsilon)$ is rather technical to prove and we omit it from this version of the paper. The basic idea is to bound distances in $||\cdot||$ from above and below by constant multiples of corresponding distances in $||\cdot||_2$. The lemma leads to the following theorem.

Theorem 8: For every $t > 0$, dimension d, and norm $||\cdot||$ of R^d, there exists a constant $c(t, d, ||\cdot||)$ such that every finite set V has a t-spanner with at most $c \cdot n$ edges.

5. OPEN PROBLEMS.

We conclude with some open problems.

1) In Theorem 1, the bound in the weight does not agree with the lower bound as nicely as in the other results. Can it be improved? We feel that our strategy of dividing the graph into planar components cannot be extended to yield the optimal answer.

2) Even in the other results, there are gaps between upper and lower bounds. For instance, in Theorem 1, the upper bound for the size is $O(n^{1 + 1/t})$, while the lower bound is $\Omega(n^{1 + 1/(2t + 3)})$. Only for $t = 1$, the known bounds, $\Theta(n^{3/2})$, are of the same order. The lower bound was proved in [L], and the upper bound is mentioned in [PU].

3) For dimensions higher than 2, Euclidean spanners with linear sizes exist. Do Euclidean spanners exist with weights within a constant multiple of the weight of the minimum spanning tree?

4) What spanners do random graphs have? The Euclidean random case has been examined in [SV].

5) Consider R^d, some fixed norm, and $t > 1$. What are the worst (or at least bad) point configurations V with respect to the number of edges in optimal t-spanners?

6) Do spanners have other applications?

6. ACKNOWLEDGEMENTS.

Thanks are due to Torsten Sillke for his help in simplifying the proof of Lemma 8.

7. REFERENCES.

[A] Althöfer: On Optimal Realizations of Finite Metric Spaces by Graphs: Discrete and Computational Geometry 3, 1988, 103-122.

[Aw] Awerbuch: Complexity of Network Synchronization: JACM, 1985, 804-823.

[ABLP] Awerbuch, Bar-Noy, Linial, Peleg: Compact Distributed Data Structures for Adaptive Routing: STOC, 1989, 479-489.

[ADDJ] Althöfer, Das, Dobkin, Joseph: Generating Sparse Spanners for Weighted Graphs: submitted to Discrete and Computational Geometry.

[AP] Awerbuch, Peleg: Routing with Polynomial Communication-Space Tradeoff: Manuscript, 1989.

[B] Bollobas: Extremal Graph Theory: Academic Press, 1978.

[BD] Bandelt, Dress: Reconstructing the Shape of a Tree from Observed Dissimilarity Data: Advances in Appl. Maths, 7, 1986, 309-343.

[Be] Bern: private communication to David Dobkin, 1989.

[C] Chew: There is a Planar Graph Almost as Good as the Complete Graph: ACM Symposium on Computational Geometry, 1986, 169-177.

[CS] Conway, Sloane: Sphere Packing, Lattices, and Groups: Springer, New York, 1988.

[D] Dress: Trees, Tight Extensions of Metric Spaces: Adv. in Math. 53, 1984, 321-402.

[DFS] Dobkin, Friedman, Supowit: Delaunay Graphs are Almost as Good as Complete Graphs: FOCS, 1987, 20-26.

[DJ] Das, Joseph: Which Triangulations Approximate the Complete Graph?: International Symposium on Optimal Algorithms, 1989 (LNCS, Springer-Verlag).

[K] Keil: Approximating the Complete Euclidean Graph: SWAT, 1988 (LNCS, Springer-Verlag).

[KG] Keil, Gutwin: The Delaunay Triangulation Closely Approximates the Complete Euclidean Graph: WADS, 1989 (LNCS, Springer-Verlag).

[L] Longyear: Regular d-valent Graphs of Girth 6 and $2(d * d - d + 1)$ Vertices: Journal of Combin. Theory 9, 1970, 420-422.

[LL] Levcopoulos, Lingas: There are Planar Graphs Almost as Good as the Complete Graphs and as Short as Minimum Spanning Trees: International Symposium on Optimal Algorithms, 1989 (LNCS, Springer-Verlag).

[PS] Peleg, Schäffer: Graph Spanners: Journal of Graph Theory, Vol 13 No 1, 1989, 99-116.

[PU] Peleg, Upfal: A Tradeoff Between Space and Efficiency for Routing Tables: STOC, 1988, 43-52.

[PUl] Peleg, Ullman: An Optimal Synchronizer for the Hypercube: SIAM Journal of Computing, Aug 1989, 740-747.

[S] Simoes-Pereira: A Note on the Tree Realizability of a Distance Matrix: Journal of Combin. Theory 6, 1969, 303-310.

[SV] Sedgewick, Vitter: Shortest Paths in Euclidean Graphs: Algorithmica 1, 1986, 31-48.

[T] Tarjan: Data Structures and Network Algorithms: Society for Industrial and Applied Mathematics, 1983.