Localized Spanner Construction for Ad Hoc Networks with Variable Transmission Range

David Peleg [⋆] and Liam Roditty

Department of Computer Science and Applied Mathematics, The Weizmann Institute of Science, Rehovot 76100, Israel.

Abstract. This paper presents an algorithm for constructing a spanner for ad hoc networks whose nodes have *variable* transmission range. Almost all previous spanner constructions for ad hoc networks assumed that all nodes in the network have the same transmission range. This allowed a succinct representation of the network as a unit disk graph, serving as the basis for the construction. In contrast, when nodes have variable transmission range, the ad hoc network must be modeled by a general disk graph. Whereas unit disk graphs are undirected, general disk graphs are directed. This complicates the construction of a spanner for the network, since currently there are no efficient constructions of low-stretch spanners for general directed graphs. Nevertheless, in this paper it is shown that the class of disk graphs enjoys (efficiently constructible) spanners of quality similar to that of unit disk graph spanners. Moreover, it is shown that the new construction can be done in a localized fashion.

1 Introduction

A wireless ad hoc network is composed of a collection S of n nodes distributed in the two dimensional plane. The nodes can communicate with each other using wireless connections. As opposed to cellular networks, there is no wire infrastructure and the connections between the nodes are restricted by their transmission energy. As nodes often receive their energy from a battery, reducing energy consumption is one of the most fundamental problems in the design of ad hoc networks. A popular approach for coping with the challenge of designing an efficient ad hoc network is to find a topology in which only a linear number of links need to be maintained, while the degradation of paths that connect any pair of nodes is restricted.

In the common wireless network model, the power needed to transmit from p to q is $|pq|^\alpha$, where $|pq|$ is the Euclidean distance between p and q and α is a constant that varies between 2 and 4. The basic assumption adopted in most of the literature on ad hoc networks is that all the nodes have the same transmission range. Consequently, the ad hoc network can be represented using a *unit disk graph*, that is, a graph in which two nodes share an edge if their

[⋆] Supported in part by grants from the Minerva Foundation and the Israel Ministry of Science.

Euclidean distance is at most 1. The size (in edges) of the unit disk graph can be as large as $O(n^2)$.

One fundamental object used in the design of ad hoc network topologies is a *spanner* [19, 18, 16]. A graph H is a t-spanner of a graph G if $\delta_H(u, v) \leq t \cdot \delta_G(u, v)$ for every two nodes u and v, where $\delta_G(u, v)$ denotes the shortest path distance between u and v in the graph G and H is a subgraph of G. The parameter t is referred to as the *stretch factor* of the spanner.

There is an extensive body of literature on spanners in both the geometric setting and the ad hoc setting. In the geometric setting, the graph G to be spanned is the complete graph over a set S of n points, where the weight of each edge of G is the distance between its endpoints in $\mathbb{R}^d$. Yao in [26] , Vaidya [23], Salowe [21] and Callahan and Kosaraju [3] showed how to compute a geometric $(1+\epsilon)$-spanner with $O(n/\epsilon^d)$ edges in $O(n \log n)$ time. In [11], Gao et. al. showed how to maintain a $(1 + \epsilon)$-spanner in a distributed manner in a mobile setting, i.e., when points can move.

In the ad hoc settings, where the graph to be spanned is a unit disk graph, the most popular constructions that are used as underlying network topologies for routing are the *relative neighborhood graph* (RNG) and *Gabriel graph* (GG) which are planar subgraphs (see [12, 2]). These graphs might suffer a very high stretch in the worst-case. Subsequent work by Gao et. al. [10], Wang and Yang-Li [24] and Yang-Li et. al. [14] considered the restricted Delaunay graph, whose worst-case stretch is constant (larger than $1 + \epsilon$). In [25], Wang and Yang-Li showed how to construct a spanner of bounded degree which is also planar. That spanner too has constant stretch.

Spanners in ad hoc networks have crucial role. Not only do they preserve the connectivity of the network but they also guarantee that the distance between every pair of nodes is within some constant factor from the shortest possible distance. Moreover, the size of the spanner is only linear. These properties made the use of spanners an attractive approach for ad hoc networks. To learn more on the tight connection between topology control in ad hoc networks and spanners see [20].

Common to all the papers mentioned above in the ad hoc setting is the assumption that the ad hoc network is represented by a unit disk graph, that is, every node of the network is assumed to have the same transmission range. This model is of significant theoretical appeal, but its accuracy is limited due to the fact that coverage areas are assumed to be disk of equal radius, implying in particular that transmission coverage must be symmetric. The focus on the restricted model of unit disk graph is partially explained by the lack of methods for dealing with more general models on one hand and the attractive properties of unit disk graphs on the other hand. There are few papers which studied more general models than the unit disk graph, such as the Quasi-unit disk graph in [13] and [17], however, these models are still limited. Li, Song and Wang [15] considered a model similar to ours, in which every node has a different transmission range. In their model an edge connects u and v in the communication graph only if u can transmit to v and v can transmit to u. Thus, the resulting graph is still undirected.

The current paper considers a more general and sometimes more natural case in which any node has a different transmission range, taken from the range $[1, M]$, and an edge is placed from u to v if u can transmit to v. This yields an intermediate model between the geometric setting and the usual ad hoc setting, as the transmission graph induced in this case is no longer a unit disk graph but a general disk graph. In such graphs, edges have a direction, since the fact that p can transmit to q does not necessarily imply that q can transmit to p. Thus, the resulting graph is directed and the transmission coverage is no longer assumed to be symmetric. In this respect, our work can be viewed as an intermediate step towards more general coverage models.

The main result of the current paper (in Section 2) is an algorithm for constructing a $(1+\epsilon)$-spanner for a given disk graph with $O(n/\epsilon^{-d} \log M)$ edges. The algorithm can be implemented in $O(m \log n)$ time, where m is the number of edges in the disk graph.

Our result is also of theoretical significance. Finding good spanners for directed graphs is a difficult problem. A general bound, similar to the one available for undirected graphs, cannot exist for the directed case, as indicated by considering the example of a directed bipartite graph in which all the edges are directed from one side to the other; clearly, any spanner for such a graph must contain every edge. In that sense, our spanner construction yields the first result establishing the existence of a directed spanner for a non-trivial class of directed graphs.

Many routing protocols for ad hoc network use only the local information which is stored with every node. In such algorithms a packet is routed out from a node by considering only its neighbors in the topology. See [22, 2, 1, 12] for more information. As our topology is constructed on top of a directed network our result opens a new direction for localized routing algorithms.

In addition, the paper also presents (in Section 3) an algorithm for constructing a linear size ($O(n/\epsilon^{-d})$ edges) $(1+\epsilon)$-spanner for a given unit disk graph. In particular, we show that any geometric $(1+\epsilon)$-spanner can be turned into a $(1+\epsilon')$-spanner for a unit disk graph by applying a simple process.

2 Spanners for general disk graphs

Let S be a set of points in $\mathbb{R}^d$ and assume that any point $p \in S$ has a transmission radius $r(p)$, taken from the range $[1, M]$. The transmission graph of S is a disk graph $I(S, E)$, whose vertices are the points of S and whose edge set includes an edge from p to q if p can transmit to q. Obviously, the resulting graph is directed, as it might happen that p can transmit to q while q cannot transmit to p. In this section we show how to compute a $(1+\epsilon)$-spanner with $O(n/\epsilon^{-d} \log M)$ edges for a given disk graph.

The construction of the spanner is based on hierarchal partition of the points of S that takes into account the variable transmission radii.

Let ϵ be an arbitrarily small positive constant and let α and β be two small constants depending on ϵ, to be fixed later on. Assume that the transmission radii are scaled so that the smallest edge in the disk graph is of weight 1. Let i be an integer from the range $[0, \lfloor \log_{1+\alpha} M \rfloor]$ and let $M_i = M/(1+\alpha)^i$. Let $E(M_{i+1}, M_i) = \{(x,y) \mid M_{i+1} \leq |xy| \leq M_i\}$. Let $\ell(x,y)$ be the level of the edge (x,y), that is, if $(x,y) \in E(M_{i+1}, M_i)$ then $\ell(x,y) = i$. Let p be a point with a transmission radius $r(p) \in [M_{i+1}, M_i]$. It follows that level i is the first level in which p can have outgoing edges. We denote this level with $\ell(p)$.

The spanner construction algorithm receives as input a (directed) disk graph $I(S,E)$ and a desired approximation factor ϵ. It constructs the set of spanner edges $E_{\text{SP}}^{\text{DIR}}$ and returns the graph $H^{\text{DIR}}(S, E_{\text{SP}}^{\text{DIR}})$. The construction is as follows. The edges of $I(S,E)$ are partitioned into classes $E(M_{i+1}, M_i)$ for $i \in [0, \lfloor \log_{1+\alpha} M \rfloor]$. Assume that in each class the edges are sorted by their weight. For every $i \in [0, \lfloor \log_{1+\alpha} M \rfloor]$, starting from $i = 0$, the edges of the class $E(M_{i+1}, M_i)$ are considered in a non-decreasing order. On each stage of the construction we maintain a set of pivots P_i. Let $x \in S$ and let $NN(x, P_i)$ be the nearest neighbor of x among the points of P_i. For a pivot $p \in P_i$, define $\Gamma_i(p) = \{x \mid x \in S, NN(x, P_i) = p, r(x) \geq |xp|\}$, that is, all the points whose nearest neighbor from P_i is p which can transmit to p.

When considering the edge (x,y), the algorithm acts according to the following rule: If $NN(x, P_i) > \beta M_{i+1}$ then x is added to P_i and the edge (x,y) is added to $E_{\text{SP}}^{\text{DIR}}$. If $NN(x, P_i) \leq \beta M_{i+1}$ and there is no edge $(x', y) \in E_{\text{SP}}^{\text{DIR}}$ such that $x' \in \Gamma_i(NN(x, P_i))$ then the edge (x,y) is added to $E_{\text{SP}}^{\text{DIR}}$. When i reaches $\lfloor \log_{1+\alpha} M \rfloor$, the algorithm handles all the edges that belong to $E(M_{\lfloor \log_{1+\alpha} M \rfloor + 1}, M_{\lfloor \log_{1+\alpha} M \rfloor})$. This includes also edges whose weight is 1, the minimal possible weight.

The spanner construction algorithm is given in Figure 1. The algorithm returns the directed graph $H^{\text{DIR}}(S, E_{\text{SP}}^{\text{DIR}})$. In what follows we prove that $H^{\text{DIR}}(S, E_{\text{SP}}^{\text{DIR}})$ is a $(1+\epsilon)$-spanner with $O(n/\epsilon^{-d} \log M)$ edges of the directed graph $I(S,E)$.

2.1 The stretch of the spanner

We start by showing that the stretch of the graph $H^{\text{DIR}}(S, E_{\text{SP}}^{\text{DIR}})$ returned by the algorithm is $1+\epsilon$.

Lemma 1 (Stretch). *Let $\epsilon > 0$ and let $H^{DIR}(S, E_{SP}^{DIR})$ be the graph returned by Algorithm* **disk-spanner**. *If $(x,y) \in E$ then $\delta_G(x,y) \leq (1+\epsilon)|xy|$.*

Proof. Assume that the transmission ranges are scaled such that the shortest edge is of weight 1. Set $\alpha = \beta < \epsilon/6$. We prove that every directed edge of an arbitrary node $x \in S$ is approximated with $1+\epsilon$ stretch. Let $i \in [0, \lfloor \log_{1+\alpha} M \rfloor]$. The proof is by induction on i. For a given node x, the base of the induction is the maximal value of i in which x has an edge in $E(M_{i+1}, M_i)$. Let j be this value for x, that is, the set $E(M_{j+1}, M_j)$ contains the shortest edge that touches x. Every other node is at distance at least M_{j+1} away from x, hence x is a pivot at this stage and every edge that touches x from the set $E(M_{j+1}, M_j)$ is added to $E_{\text{SP}}^{\text{DIR}}$.

```
Algorithm disk-spanner $(I(S,E), \epsilon)$
$E_{\text{SP}}^{\text{DIR}} \leftarrow \phi$
$P_0 \leftarrow \phi$
for $i \leftarrow 0$ to $\lfloor \log_{1+\alpha} M \rfloor$
    for each $(x,y) \in E(M_{i+1}, M_i)$ do
        if $|NN(x,P_i)x| > \beta M_{i+1}$ then
            $P_i \leftarrow P_i \cup \{x\}$
        if $\nexists (x',y) \in E_{\text{SP}}^{\text{DIR}}$ s.t. $x' \in \Gamma_i(NN(x,P_i))$
            $E_{\text{SP}}^{\text{DIR}} \leftarrow E_{\text{SP}}^{\text{DIR}} \cup \{(x,y)\}$
    $P_{i+1} \leftarrow P_i$
return $H^{\text{DIR}}(S, E_{\text{SP}}^{\text{DIR}})$
```

Fig. 1. A high level implementation of the spanner construction algorithm for *general* disk graphs

We now turn to prove the induction hypothesis. Let $(x,y) \in E(M_{i+1}, M_i)$ for some $i < j$ and let $p = NN(x, P_i)$. If the edge (x,y) is not in the spanner, then there must be an edge $(\hat{x}, y) \in E_{\text{SP}}^{\text{DIR}}$, where $\hat{x} \in \Gamma_i(p)$. The crucial observation is that x has a transmission range of at least M_{i+1}. It follows from the algorithm that $|\hat{x}p| \leq \beta M_{i+1}$ and $|xp| \leq \beta M_{i+1}$.

By the choice of β, it follows that $2\beta M_{i+1} < M_{i+1}$ and $(x, \hat{x}) \in E$. Thus, there is a (directed) path from x to y of the form $\langle x, \hat{x}, y \rangle$ whose length is $2\beta M_{i+1} + M_i$. However, only the edge $(\hat{x}, y)$ is in $E_{\text{SP}}^{\text{DIR}}$. By the inductive hypothesis, the edge $(x, \hat{x})$ whose weight is $2\beta M_{i+1}$ is approximated with $1+\epsilon$ stretch. Thus, there is a path in the spanner from x to y whose length is at most $(1+\epsilon)|x\hat{x}| + M_i$, and this can be bounded by

$$(1+\epsilon)2\beta M_{i+1} + M_i = ((1+\epsilon)2\beta + (1+\alpha))M_{i+1}.$$

As the edge $(x,y) \in E(M_{i+1}, M_i)$ it follows that $|xy| \geq M_{i+1}$. It remains to prove that $1 + 2\epsilon\beta + 2\beta + \alpha \leq 1 + \epsilon$, which follows directly from the choice of α and β.

2.2 The size of the spanner

We now prove that the size of the spanner $H^{\text{DIR}}(S, E_{\text{SP}}^{\text{DIR}})$ is $O(n/\epsilon^d \log M)$. As a first step, we state the following well-known lemma, cf. [9].

Lemma 2. *[Packing Lemma] If all points in a set $U \in \mathbb{R}^d$ are at least r apart from each other, then there are at most $(2R/r+1)^d$ points in U within any ball X of radius R.*

The next lemma establishes a bound on the number of incoming spanner edges that a point may be assigned on stage $i \in [0, \lfloor \log_{1+\alpha} M \rfloor]$ of the algorithm.

Lemma 3. *Let $i \in [0, \lfloor \log_{1+\alpha} M \rfloor]$ and let $y \in S$. The total number of incoming edges of y that were added to the spanner on stage i is $O(\epsilon^{-d})$.*

Proof. Let (x, y) be a spanner edge and let $NN(x, P_i) = p$. We associate (x, y) to p. From the spanner construction algorithm it follows that this is the only incoming edge of y whose source is in $\Gamma_i(p)$. Thus, this is the only incoming edge of y which is associated to p. Now consider all the incoming edges of y on stage i. The source of each of these edges is associated to a unique pivot within distance of at most $M_i + \beta M_{i+1}$ away from y and any two pivots are βM_{i+1} apart from each other. Using Lemma 2, we get that the number of edges entering y is $(\frac{M_i + \beta M_{i+1}}{\beta M_{i+1}} + 1)^d = ((1+\alpha)/\beta + 2)^d = O(\epsilon^{-d})$.

It follows from the above lemma that the total number of edges that were added to $E_{\mathrm{SP}}^{\mathrm{DIR}}$ in the main loop is $O(n/\epsilon^d \log M)$.

2.3 The construction time

We now describe how to efficiently implement the algorithm. Let n be the number of vertices and let m be the number of edges in the disk graph $I(S, E)$.

First, the algorithm has to partition the set E into the sets $E(M_{\lfloor \log_{1+\alpha} M \rfloor + 1}, M_{\lfloor \log_{1+\alpha} M \rfloor}), \ldots, E(M_1, M_0)$. This can be done in $O(m)$ time. The algorithm also preforms nearest neighbor queries. It is easy to see that at most $O(m)$ such queries are processed. To obtain an efficient implementation we maintain the set P_i using the dynamic nearest neighbor data structure of Cole and Gottlieb [6]. Every operation is supported in $O(\log n)$ time. However, their data structure is only capable of answering ϵ-approximate nearest neighbor queries. Luckily, it is enough for our purpose. The only effect of using an approximation is that the separation between any two pivots becomes $(1 + \epsilon')\beta M_{i+1}$ for some arbitrarily small $\epsilon' > 0$, instead of βM_{i+1}, which has a negligible effect on our bounds.

Any new pivot is inserted into the data structure in $O(\log n)$ time. The set of pivots on the $(i+1)$st stage is initiated with the set of pivots of the ith stage. Thus, any point is inserted exactly once into that data structure.

By the above discussion it follows that the total cost of the construction algorithm is $O(m \log n)$.

2.4 A localized algorithm

We now turn to describe a localized implementation of the algorithm. We assume a synchronous model in which a unique id is assigned to every node and that any node knows the id's of its outgoing neighbors (i.e., the nodes it can reach).

Similarly to the centralized algorithm, the localized algorithm of every node u has a main loop and in each iteration of the main loop the pivots of the current level are chosen by a simple adaption of the standard distributed algorithm for finding a maximal independent set (cf. Peleg [18]; chapter 8). More specifically, let $N_i(u)$ be the set of nodes at distance at most βM_{i+1} from u whose transmission radius is at least M_{i+1}, where u is the node currently running the algorithm.

```
Algorithm local-disk-spanner (code for node $u$)

for $i \leftarrow 0$ to $\lfloor \log_{1+\alpha} M \rfloor$
    $v \leftarrow extract\text{-}min(N_i(u))$
    if $id(v) > id(u)$
        $E_{\mathrm{SP}}^{\mathrm{DIR}} \leftarrow \phi$
        obtain $E^v(M_{i+1}, M_i)$ from every $v \in N_i(u)$
        let $\hat{E} \leftarrow (\cup_{v \in N_i(u)} E^v(M_{i+1}, M_i)) \cup E^u(M_{i+1}, M_i)$
        for every $(x, y) \in \hat{E}$ do
            if $\nexists (x', y) \in E_{\mathrm{SP}}^{\mathrm{DIR}}$ s.t. $x' \in N_i(u)$
              $E_{\mathrm{SP}}^{\mathrm{DIR}} \leftarrow E_{\mathrm{SP}}^{\mathrm{DIR}} \cup \{(x, y)\}$
              send $(x, y)$ to $x$
```

Fig. 2. A localized spanner construction algorithm for *general* disk graphs

If the graph was undirected then this set could be obtained easily. However, in the directed case u may have neighbors whose transmission range it too small, and thus should not be in $N_i(u)$. By a simple procedure we can overcome this problem without adding any additional assumption to our model. Notice that every node in $N_i(u)$ can transmit to u, thus, u can broadcast a message within its transmission range and every neighbor that gets the message returns an acknowledgment to u if u is within its transmission range. By this procedure, u can find its neighbors that can transmit to it and this is the only information that is needed in order to form the set $N_i(u)$.

The pivot selection is done as follows. The node v with minimal id in $N_i(u)$ is extracted from $N_i(u)$ and if u's id is smaller than v's then u marks itself as a pivot in level i. If u is not a pivot then nothing further is done. However, if u is a pivot then it performs a centralized computation of the spanner edges emanating from nodes of $N_i(u)$, and informs these nodes. For a node v, let $E^v(M_{i+1}, M_i)$ denotes the set of edges of $E(M_{i+1}, M_i)$ emanating from v. The edges that emanate from u and from the nodes of $N_i(u)$ are scanned in a non-decreasing order of length and an edge (x, y) is added to the spanner if and only if it is the first edge from a vertex of $\{u\} \cup N_i(u)$ to y. The algorithm is formally given in Figure 2. Next, we show that the message complexity is linear in the number of edges of the disk graph.

Lemma 4. *The message complexity of the localized algorithm is $O(m + n/\epsilon^d)$.*

Proof. In the i-th iteration every node v sends its edge set $E^v(M_{i+1}, M_i)$ to every pivot u where $v \in N_i(u)$. From packing arguments it follows that there is a constant number of pivots that have v in their close neighbor set. Thus, every edge of $E^v(M_{i+1}, M_i)$ is passed to a constant number of pivots and in total v generates $O(deg(v))$ messages. When a pivot computes the spanner edges it sends messages only to points that have to maintain a link that corresponds to a spanner edge. The total number of such messages is simply the number of spanner edges which is $O(n/\epsilon^d)$.

2.5 Topology updates

A fundamental question in topology control is what will happen when the underlined communication graph is being changed. For example, points are removed from the network or new points are added.

In our case it is easy to see that a deletion or an insertion of one point may remove or add many links which are essential to the connectivity of the network and thus must be in the spanner without considering the distances. As a result of that at the worse-case it may take $\Omega(n)$ time to update the spanner. Deleting and inserting the point u causes to update cost which is proportional to the number of points with small transmission range that are within the transmission range of u.

2.6 Simulations

We have implemented our spanner construction algorithm and tested it on randomly generated disk graphs. The graphs are generated by picking random points in a region of predefined size. Each point is also assigned a random transmission range from a predefined interval. A disk graph is then created by adding an edge from a point p to q if q is within the transmission radius of p. We have constructed spanners with required stretch factors of 2 and 3. Given a region size and a maximal radius, 100 different graphs were generated and the results were averaged over all these graphs. The results are summarized in Table 1 and Table 2. A careful look at the spanner construction reveals that the average degree of a node in the spanner is at most $25 \log M$. As the results indicate (and as one may expect), when the random disk graph becomes denser, then the spanner obtains a better compression rate. The average degree of a node in the random disk graph is reduced by half or more in most of the cases and the resulted spanner has an average degree which is less than $25 \log M$. It implies that there are many natural instances on which better bounds then the worse case bound can be obtained by our spanner construction algorithm.

3 A $(1 + \epsilon)$-spanner for unit disk graphs

In this section we show how to compute a $(1 + \epsilon)$-spanner for a unit disk graph. More specifically, we show that given a set of points S any $(1 + \epsilon)$ geometric spanner of S can be turned into $(1 + \epsilon')$-spanner of the unit disk graph of S.

Let $H(S, E_{\text{SP}})$ be a geometric $(1+\epsilon)$-spanner of S and let $I(S, E)$ be the unit disk graph of S. The following lemma shows that the distances induced by the graph $I(S, E)$ are approximated with a stretch factor of $1 + \epsilon$ in $H(S, E_{\text{SP}})$.

Table 1. Stretch 2

Region	Max Radius	Points	Edges	Removed Edges	Required Stretch	Savings
10 × 10	16	50	1565	343	2	0.22
15 × 15	20	100	5769	1878	2	0.33
25 × 25	25	200	18145	6999	2	0.39
30 × 30	35	500	133752	81916	2	0.61

Table 2. Stretch 3

Region	Max Radius	Points	Edges	Removed Edges	Required Stretch	Savings
10 × 10	16	50	1553	697	3	0.45
15 × 15	20	100	5813	3336	3	0.57
25 × 25	25	200	18304	11725	3	0.64
30 × 30	35	500	134203	108331	3	0.81

Lemma 5. *Let S be a set of points and let $H(S, E_{SP})$ be any $(1+\epsilon)$-spanner of S. If $I(S, E)$ is the unit disk graph of S then $\delta_H(p, q) \leq (1+\epsilon)\delta_I(p, q)$ for every pair of points $p, q \in S$.*

Proof. Let $p, q \in S$ and let $p = x_1, x_2, \ldots, x_\ell = q$ be the vertices on a shortest path between p and q in $I(S, E)$. By the definition of H, $\delta_H(x_i, x_{i+1}) \leq (1+\epsilon)|x_i x_{i+1}|$. Thus, $\delta_H(p, q) \leq (1+\epsilon)\sum_{i=1}^{\ell-1} |x_i x_{i+1}| = (1+\epsilon)\delta_I(p, q)$.

The above lemma states that for any pair of points there exists a path in H that approximates the shortest path between them in the unit disk graph I. However, H is not necessarily a *spanner* of I, as it might have edges that are not included in I, while a spanner must be a subgraph of the original graph. At first glance it might seem that a possible solution to this problem is to remove every edge whose length is strictly greater than 1 from H. Indeed, by doing so we ensure that the resulting graph is a subgraph of I. However, it might no longer be a $(1+\epsilon)$-spanner for I. In particular, consider two points p and q such that $|pq| = 1$. It might so happen that the path σ that approximates this distance in G is composed of two edges, (p, r) and (r, q), where $|pr| = 1+\epsilon/2$ and $|rq| = \epsilon/2$. In such a situation, if all edges whose weight is greater than 1 are removed from H, then the path σ is disconnected.

Our solution to this problem is as follows. Starting from a $(1+\epsilon)$ geometric spanner $H(S, E_{\text{SP}})$ of S, every edge whose length is in the range $(1, 1+\epsilon]$ is removed from E_{SP}. In compensation, for any removed edge we add, if possible,

```
Algorithm unit-disk-spanner $(I(S,E),\epsilon)$

$H(S,E_{\mathrm{SP}}) \leftarrow$ geom-spanner$(S,\epsilon)$
$E_{\mathrm{SP}}^{\mathrm{UDG}} \leftarrow E_{\mathrm{SP}}$
for every $(x,y) \in E_{\mathrm{SP}}^{\mathrm{UDG}}$ do
    if $|xy| > 1$ then
        $E_{\mathrm{SP}}^{\mathrm{UDG}} \leftarrow E_{\mathrm{SP}}^{\mathrm{UDG}} \setminus \{(x,y)\}$
    if $|xy| \in (1, 1+\epsilon]$ then
        if $\exists (u,v) \in E$ s.t. $|xu| \leq \epsilon \wedge |vy| \leq \epsilon$
            $E_{\mathrm{SP}}^{\mathrm{UDG}} \leftarrow E_{\mathrm{SP}}^{\mathrm{UDG}} \cup \{(x,u),(u,v),(v,y)\}$
return $H^{\mathrm{UDG}}(S, E_{\mathrm{SP}}^{\mathrm{UDG}})$
```

Fig. 3. A high level implementation of the spanner construction algorithm for *unit* disk graphs

at most three replacement edges. Each of these three new edges belongs to the unit disk graph and their total length is at most $1 + 2\epsilon$.

Specifically, let (x, y) be an edge whose weight is in the range $(1, 1 + \epsilon]$. We look for a pair of points u and v such that $|xu| \leq \epsilon$, $|vy| \leq \epsilon$ and $(u, v) \in E$. If such a pair of points exists, we add the edges (x, u), (u, v) and (v, y) to the spanner instead of the edge (x, y). Such a situation is depicted in Figure 4. Notice that it might be that $u = x$ or $v = y$. If no such pair of points u and v is found, then nothing is done. Denote the resulting spanner by $H^{\mathrm{UDG}}(S, E_{\mathrm{SP}}^{\mathrm{UDG}})$. The algorithm is given in Figure 3.

3.1 The properties of the spanner

In this section we show that the unit disk graph spanner constructed by our algorithm has the same properties as a regular geometric spanner.

Lemma 6. *The graph $H^{UDG}(S, E_{SP}^{UDG})$ constructed by* **unit-disk-spanner** *Algorithm is a $(1 + \epsilon)$-spanner of $I(S, E)$ with $O(n/\epsilon^d)$ edges.*

Proof. It is easy to see that $E_{\mathrm{SP}}^{\mathrm{UDG}} \subseteq E$, as every edge of $E_{\mathrm{SP}}^{\mathrm{UDG}}$ is of weight at most 1. From Lemma 5 it follows that every edge of $I(S, E)$ is approximated by the geometric spanner. The removal of an edge whose weight is strictly greater than $1 + \epsilon$ has no effect on the approximation of edges of the unit disk graph, since these edges are of weight 1 or less, so edges of weight greater than $1 + \epsilon$ do not participate in approximating them. When an edge whose weight is in the range $(1, 1+\epsilon]$ is replaced with a path of length at most $1+2\epsilon$, only the approximation factor is affected, increasing from $1 + \epsilon$ to at most $(1 + \epsilon)(1 + 2\epsilon) \leq 1 + 5\epsilon$. It remains to show that if a removed edge whose weight is from the range $(1, 1+\epsilon]$ has no replacement path, then its removal is harmless, i.e., there is no edge in the unit disk graph whose approximation is affected. Consider such a removed

Fig. 4. A geometric spanner edge and its possible replacement path

edge (x, y), and assume that there is no edge $(u, v) \in E$ such that $|xu| \leq \epsilon$ and $|vy| \leq \epsilon$. It follows that for every edge in the unit disk graph, at least one of its endpoints is at distance strictly greater than ϵ from both x and y. Thus, the edge (x, y) cannot be used in the approximation of any edge of the unit disk graph and it can be removed without effecting the approximation factor.

The size of $E_{\text{SP}}^{\text{UDG}}$ remains $O(n/\epsilon^d)$, as at most three edges are added for any removed edge.

3.2 The construction time

In this section we explain how to efficiently implement the algorithm when the set of points is in the plane. For every edge of the geometric spanner whose weight is in the range $(1, 1+\epsilon]$, we need to check whether a replacement path exists. For every $p \in S$, we create a nearest neighbor data structure for the points within a radius of ϵ around p (including the point itself). The cost for that is at most $O(\deg(p) \log \deg(p))$, where $\deg(p)$ is the degree of p in $I(S, E)$ (See, [8, 5]). Queries can be answered in $O(\log \deg(p))$ time. Given an edge $(x, y) \in E_{\text{SP}}^{\text{UDG}}$ whose weight is in the range $(1, 1+\epsilon]$, we scan all the edges of length at most ϵ that touch x in $I(S, E)$. For each such edge (x, u), the algorithm queries the data structure of y to find the closest point to u among the points within distance ϵ from y. If the closest point is at distance 1 or less, then we have found the replacement path. If not, then we proceed to the next edge of x. The cost of this search is $O(\deg(x) \log \deg(y))$. This is done for every geometric spanner edge whose weight is in the range $(1, 1+\epsilon]$. A problem may arise if a point with large degree in $I(S, E)$ also has many spanner edges. To avoid that, we use a geometric spanner of bounded degree [4, 7], that is, one where every point has $O(\epsilon^{-d})$ spanner edges. Hence every point will take part in $O(\epsilon^{-d})$ tests, each of cost proportional to its degree. The total running time is thus $O(m \log n)$.

The next theorem summarizes the above arguments.

Theorem 1. *Let S be a set of n points in the plane. Let $I(S, E)$ be the unit disk graph that corresponds to S, where $|E| = m$. There exists a $(1+\epsilon)$ spanner of $I(S, E)$ with $O(n/\epsilon)$ edges that can be constructed in $O(m \log n)$ time.*

4 Concluding remarks

We have presented in this paper two constructions. The first and most important is the first construction ever of spanners for disk graphs. This result raises many other questions, both practical and theoretical. From the perspective of routing it is interesting to use this construction as a topology for greedy based routing algorithms in ad-hoc networks. Our spanner construction allows routing in ad-hoc networks with variable transmission radii. It is also interesting to consider the question of whether efficient compact routing schemes exhibiting a tradeoff between the space usage of each node and the stretch of the paths exist for the model of disk graphs. From a theoretical perspective it is interesting to explore which other natural classes of directed graphs have good spanners.

References

1. Prosenjit Bose and Pat Morin. Online routing in triangulations. In *ISAAC '99: Proceedings of the 10th International Symposium on Algorithms and Computation*, pages 113–122, London, UK, 1999. Springer-Verlag.
2. Prosenjit Bose, Pat Morin, Ivan Stojmenovic, and Jorge Urrutia. Routing with guaranteed delivery in ad hoc wireless networks. *Wireless Networks*, 7(6):609–616, 2001.
3. P. B. Callahan and S. R.Kosaraju. A decomposition of multidimensional point sets with applications to k-nearest-neighbors and n-body potential fields. *J. ACM*, 42:67–90, 1995.
4. T-H. Chan, A. Gupta, B.M. Maggs, and S. Zhou. On hierarchical routing in doubling metrics. In *Proc. 16th ACM-SIAM Symp. on Discrete Algorithms*, pages 762–771, 2005.
5. Timothy Chan and Mihai Patrascu. Point location in sublogarithmic time and other transdichotomous results in computational geometry. In *Proc. 47th IEEE Symp. on Foundations of Computer Science*, pages 325–332, 2006.
6. R. Cole and L. Gottlieb. Searching dynamic point sets in spaces with bounded doubling dimension. In *Proc. 38th ACM Symp. on Theory of Computing*, 2006.
7. G. Das, G. Naraimhan, and J. Salowe. A new way to weigh malnourished Euclidean graphs. In *Proc. 6th ACM-SIAM Symp. on Discrete Algorithms*, pages 215–222, 1995.
8. David P. Dobkin and Richard J. Lipton. Multidimensional searching problems. *SIAM J. Comput.*, 5(2):181–186, 1976.
9. J. Gao, L. Guibas, and A. Nguyen. Deformable spanners and applications. In *Proc. 20th ACM Symp. on Computational Geometry*, pages 179–199, 2004.
10. Jie Gao, Leonidas J. Guibas, John Hershberger, Li Zhang, and An Zhu. Geometric spanners for routing in mobile networks. *IEEE J. on Selected Areas in Communications*, 23(1):174–185, 2005.
11. Jie Gao, Leonidas J. Guibas, and An Nguyen. Distributed proximity maintenance in ad hoc mobile networks. In *Proc. IEEE Conf. on Distributed Computing in Sensor Systems*, pages 4–19, June 2005.
12. Brad Karp and H. T. Kung. GPSR: greedy perimeter stateless routing for wireless networks. In *Proc. 6th Conf. on Mobile computing and networking*, pages 243–254, 2000.
13. Fabian Kuhn and Aaron Zollinger. Ad-hoc networks beyond unit disk graphs. In *DIALM-POMC '03: Proceedings of the 2003 joint workshop on Foundations of mobile computing*, pages 69–78, New York, NY, USA, 2003. ACM.
14. Xiang-Yang Li, Gruia Calinescu, Peng-Jun Wan, and Yu Wang. Localized delaunay triangulation with application in ad hoc wireless networks. *IEEE Trans. on Parallel and Distributed Systems*, 14(10):1035–1047, 2003.

15. Xiang-Yang Li, Wen-Zhan Song, and Yu Wang. Localized topology control for heterogeneous wireless sensor networks. *ACM Transactions on Sensor Networks*, 2(1):129–153, February 2006.
16. Giri Narasimhan and Michiel Smid. *Geometric Spanner Networks.* Cambridge University Press, 2007.
17. Melih Onus and Andrea Richa. Efficient broadcasting and gathering in wireless ad-hoc networks. In *ISPAN '2005*, 2005.
18. David Peleg. *Distributed computing: a locality-sensitive approach.* Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2000.
19. David Peleg and Alejandro A. Schäffer. Graph spanners. *J. Graph Theory*, 13:99–116, 1989.
20. Rajmohan Rajaraman. Topology control and routing in ad hoc networks: a survey. *SIGACT News*, 33(2):60–73, 2002.
21. J. S. Salowe. Constructing multidimensional spanner graphs. *Int. J. Comput. Geometry Appl*, 1(2):99–107, 1991.
22. Ivan Stojmenovic and Xu Lin. Loop-free hybrid single-path/flooding routing algorithms with guaranteed delivery for wireless networks. *IEEE Trans. Parallel Distrib. Syst.*, 12(10):1023–1032, 2001.
23. P. M. Vaidya. A sparse graph almost as good as the complete graph on points in K dimensions. *Discrete & Computational Geometry*, 6:369–381, 1991.
24. Yu Wang and Xiang-Yang Li. Efficient delaunay-based localized routing for wireless sensor networks. *Int. J. of Communication Systems*, 20(7):767–789, 2006.
25. Yu Wang and Xiang-Yang Li. Localized construction of bounded degree and planar spanner for wireless ad hoc networks. *MONET*, 11(2):161–175, 2006.
26. Andrew Chi-Chih Yao. On constructing minimum spanning trees in k-dimensional spaces and related problems. *SIAM J. Comput.*, 11(4):721–736, 1982.