Local Approximation Schemes for Topology Control*

Mirela Damian Saurav Pandit Sriram Pemmaraju

Abstract

This paper presents a distributed algorithm on wireless ad-hoc networks that runs in polylogarithmic number of rounds in the size of the network and constructs a linear size, lightweight, $(1+\varepsilon)$-spanner for any given $\varepsilon > 0$. A wireless network is modeled by a d-dimensional α-quasi unit ball graph (α-qUBG), which is a higher dimensional generalization of the standard unit disk graph (UDG) model. The d-dimensional α-qUBG model goes beyond the unrealistic "flat world" assumption of UDGs and also takes into account transmission errors, fading signal strength, and physical obstructions. The main result in the paper is this: for any fixed $\varepsilon > 0$, $0 < \alpha \leq 1$, and $d \geq 2$ there is a distributed algorithm running in $O(\log n \cdot \log^* n)$ communication rounds on an n-node, d-dimensional α-qUBG G that computes a $(1+\varepsilon)$-spanner G' of G with maximum degree $\Delta(G') = O(1)$ and total weight $w(G') = O(w(MST(G))$. This result is motivated by the topology control problem in wireless ad-hoc networks and improves on existing topology control algorithms along several dimensions. The technical contributions of the paper include a new, sequential, greedy algorithm with relaxed edge ordering and lazy updating, and clustering techniques for filtering out unnecessary edges.

Keywords: Spanners, topology control, wireless ad-hoc networks, unit ball graphs.

Number of pages: 20 in the main body and 7 in the appendix.

*The first author is at the Department of Computer Science, Villanova University, Villanova, PA 19085. E-mail: `mirela.damian@villanova.edu`. The other two authors are at the Department of Computer Science, The University of Iowa, Iowa City, IA 52242-1419. E-mail: [`spandit`, `sriram`]`@cs.uiowa.edu`. The contact author is Sriram Pemmaraju. His phone number is 319 353 2956 and fax number is 319 335 3624.

1 Introduction

Let $G = (V, E)$ be a graph with edge weights $w : E \to \mathcal{R}^+$. For $t \geq 1$, a t-spanner of G is a spanning subgraph G' of G such that for all pairs of vertices $u, v \in V$, the length of a shortest uv-path in G' is at most t times the length of a shortest uv-path in G. The problem of constructing a sparse t-spanner, for small t, of a given graph G has been extensively studied by researchers in distributed computing and computational geometry and more recently by researchers in ad-hoc wireless networks. In this paper we present a fast distributed algorithm for constructing a linear size, lightweight t-spanner of bounded degree for any given $t > 1$, on wireless networks. Below, we describe our result more precisely.

Network model. In this paper we model wireless networks using d-dimensional quasi unit ball graphs. For any fixed α, $0 < \alpha \leq 1$ and integer $d \geq 2$, a *d-dimensional α-quasi unit ball graph* (*α-qUBG*, in short) is a graph $G = (V, E)$ whose vertex set V can be placed in one-one correspondence with a set of points in the d-dimensional Euclidean space and whose edge set E satisfies the constraint: if $|uv| \leq \alpha$ then $\{u, v\} \in E$ and if $|uv| > 1$ then $\{u, v\} \notin E$. Here we use $|uv|$ to denote the Euclidean distance between the points corresponding to vertices u and v. The α-qUBG model does not prescribe whether a pair of vertices whose distance is in the range $(\alpha, 1]$ are to be connected by an edge or not. This is an attempt to take into account transmission errors, fading signal strength, and physical obstructions. The α-qUBG model is a higher dimensional generalization of the somewhat simplistic unit disk graph (UDG) model of wireless networks that is popular in literature. Specifically, when $\alpha = 1$ and $d = 2$, a d-dimensional α-qUBG is just a UDG. UDGs are attractive due to their mathematical simplicity, but have been deservedly criticized for being unrealistic models of wireless networks [10]. In our view, d-dimensional α-qUBGs are a significant step towards a more realistic model of wireless networks. Two-dimensional α-qUBGs were proposed in [1] as a model of wireless ad-hoc networks with unstable transmission ranges and the difficulty of doing geometric routing in such networks was shown.

Our result. For any edge weighted graph J, we use $w(J)$ to denote the sum of the weights of all the edges in J and $MST(J)$ to denote a minimum weight spanning tree of J. For any fixed $\varepsilon > 0$, $0 < \alpha \leq 1$, and $d \geq 2$ our algorithm runs in $O(\log n \cdot \log^* n)$ communication rounds on an n-node, d-dimensional α-qUBG and computes a $(1+\varepsilon)$-spanner G' of G whose maximum degree $\Delta(G') = O(1)$ and whose total weight $w(G') = O(w(MST(G))$. Since any spanner of G has weight bounded below by $w(MST(G))$, the weight of the output produced by the algorithm is within a constant times the optimal weight. Our algorithm does not need to know the locations of nodes of the α-qUBG in d-dimensional Euclidean space; just the pairwise Euclidean distances. As far as we know, our result significantly improves all known results of a similar kind along several dimensions. More on this further below.

Topology control. Our result is motivated by the *topology control* problem in wireless ad-hoc networks. For an overview of topology control, see the survey by Rajaraman [17]. Since an ad-hoc network does not come with fixed infrastructure, there is no topology to start with and informally speaking, the topology control problem is one of selecting neighbors for each node so that the resulting topology has a number of useful properties. More precisely, let V be a set of nodes that can communicate via wireless radios and for each $v \in V$, let $N(v)$ denote the set of all nodes that v can reach when transmitting at maximum power. The induced digraph $G = (V, E)$, where $E = \{\{u, v\} \mid v \in N(u)\}$, represents the network in which every node has chosen to transmit at maximum power and has designated every node it can reach as its neighbor. The topology control problem is the problem of devising an efficient and local protocol P for selecting a set of neighbors $N_P(v) \subseteq N(v)$ for each node $v \in V$. The induced digraph $G_P = (V, E_P)$, where $E_P = \{\{u, v\} \mid v \in N_P(u)\}$ is typically required the satisfy properties such as symmetry (if $v \in N_P(u)$ then $u \in N_P(v)$), sparseness ($|E_P| = O(|V|)$) or bounded degree ($|N_P(v)| \leq c$ for all nodes v and some constant c), and the spanner property. Sometimes stronger versions of connectivity such as k-vertex connectivity or k-edge connectivity (for $k > 1$) are desired, both for providing fault-tolerance and for improving throughput [6, 7]. If the input graph consists of nodes in the plane, it is quite common to require that the output graph be planar [13, 14, 15, 18, 19]. This requirement is motivated by the existence of simple, memory-less, geometric routing algorithms that guarantee message delivery only when the underlying graph is planar [9].

Though the topology control problem is recent, there is already an extensive body of literature on the problem to which the above sample of citations do not do justice. However, many of the topology control protocols that provide worst case guarantees on the quality of the topology, assume that the network is modeled by a UDG. A recent example [15] presents a distributed algorithm that requires a linear number of communication rounds in the worst case to compute a planar t-spanner of a given UDG with $t \approx 6.2$ and in which each node has degree at most 25. These two constants can be slightly tuned – t can be brought down to about 3.8 with a significant increase in the degree bound. We improve on the result in [15] along several dimensions. As is generally known among practitioners in ad-hoc wireless networks, the "flat world" assumption and the identical transmission range assumption of UDGs is unrealistic [10]. By using an α-qUBG we significantly generalize our model of wireless networks, hopefully moving much closer to reality. For any $\varepsilon > 0$, our algorithm returns a $(1 + \varepsilon)$-spanner; as far as we know, this is the first distributed algorithm that produces an *arbitrarily good* spanner for *any* model of wireless networks. We also guarantee that the total weight of the output is within constant times optimal – a guarantee that is not provided in [15]. Finally, using algorithmic techniques and distributed data structures that might be of independent interest, we ensure that our protocol runs in $O(\log n \cdot \log^* n)$ communication rounds. We are not aware of any topology control algorithm that runs in poly-logarithmic number of rounds and provides anywhere close to the guarantees provided by our algorithm.

Spanners in computational geometry. Starting in the early 1990's, researchers in computational geometry have attempted to find sparse, lightweight spanners for complete Euclidean graphs. Given a set P of n points in $\mathcal{R}^d$, the tuple (P, E), where E is the set of line segments $\{\{p, p'\} \mid p, p' \in P\}$, is called the *complete Euclidean graph* on P. For any subset $E' \subseteq E$, (P, E') is called a *Euclidean graph* on P. The specific problem that researchers in computational geometry have considered, is this. Given a set P of n points in $\mathcal{R}^d$ and $t > 1$, compute a Euclidean graph on P that is a t-spanner of the complete Euclidean graph on P, whose maximum degree is bounded by $O(1)$ and whose weight is bounded by the weight of a minimum spanning tree on P. For an early example, see [12] in which the authors show that there are "planar graphs almost as good as the complete graphs and almost as cheap as minimum spanning trees." This was followed by a series of improvements [2, 3, 4, 5], with the most recent paper [2] presenting algorithms for constructing Euclidean subgraphs that provide the additional property of k-fault tolerance. Most of the papers mentioned above start with the following simple, greedy algorithm.

Algorithm `SEQUENTIAL-GREEDY` $(G = (V, E), t)$

---

1. Order the edges in E in non-decreasing order of length.
2. $E' \leftarrow \phi$, $G' \leftarrow (V, E')$
3. For each edge $e = \{u, v\} \in E$ if there is no uv-path in G' of length at most $t \cdot |uv|$
   (a) $E' \leftarrow E' \cup \{e\}$
   (b) $G' \leftarrow (V, E')$

Output G'.

It is well-known [4] that if the input graph $G = (V, E)$ is the complete Euclidean graph, then the output graph $G' = (V, E')$ produced by `SEQUENTIAL-GREEDY` has the following useful properties: (i) G' is a t-spanner of G, (ii) $\Delta(G') = O(1)$, and (iii) $w(G') = O(w(MST(G)))$. A naive implementation of `SEQUENTIAL-GREEDY` takes $O(n^3 \log n)$ time because a quadratic number of shortest path queries need to be answered on a dynamic graph with $O(n)$ edges. Consequentially, papers in this area [4, 5] focus on trying to implement `SEQUENTIAL-GREEDY` efficiently. For example, Das and Narasimhan [4] show how to use certain kind of graph clustering to answer shortest path queries efficiently, thereby reducing the running time of `SEQUENTIAL-GREEDY` to $O(n \log^2 n)$. One of the contributions of this paper is to show how a variant of the Das-Narasimhan clustering scheme can be implemented and maintained efficiently, in a distributed setting.

Summary of our contributions. In obtaining the main result, our paper makes the following contributions.

1. We first show that sparse, lightweight, t-spanners for arbitrarily small $t > 1$, not only ex-

ist for d-dimensional α-qUBGs, but such spanners can be computed using `SEQUENTIAL-GREEDY`. Note that sparse t-spanners for arbitrarily small values of $t \geq 1$ do not exist for general graphs. For example, there is a classical graph-theoretic result that shows that for any $t \geq 1$, there exist (infinitely many) unweighted n-vertex graphs for which every t-spanner needs $\Omega(n^{1+1/(t+2)})$ edges (see Page 179 in [16]).

2. We then consider a version of `SEQUENTIAL-GREEDY` in which the requirement that edges be considered in increasing order of length is relaxed. More precisely, the edges are distributed into $O(\log n)$ bins $B_0, B_1, B_2, \ldots$ such that edges in B_i are all shorter than edges in B_{i+1}. It is then shown that *any* ordering of the edges in which edges in B_0 come first, followed by edges in B_1, followed by the edges in B_2, etc., is good enough for the correctness of `SEQUENTIAL-GREEDY`, even for d-dimensional α-qUBGs. More importantly, we show that the update step in `SEQUENTIAL-GREEDY` (Step 3(a)) need not be performed after each edge is queried. Instead, a more lazy update may be performed, after each bin is completely processed. Being able to perform a lazy update is critical for a distributed implementation; roughly speaking, we want the nodes to query all edges in a bin in parallel and not to have to rely on answers to queries on other edges in a bin.

3. We also use a novel clustering technique as a way to reduce the number of edges to be queried per node. Reducing the number of query edges per node, is critical to being able to guarantee that the output of our distributed version of `SEQUENTIAL-GREEDY` does not have too many edges incident on a node.

4. We then show that this relaxed version of `SEQUENTIAL-GREEDY` can be implemented in a distributed setting in $O(\log n)$ phases — one phase corresponding to each bin — such that each phase requires $O(\log^* n)$ communication rounds. Each phase requires the computation of maximal independent sets (MIS) on some derived graphs. We show that the derived graphs are unit ball graphs of *constant doubling dimension* [11] and use the $O(\log^* n)$-round MIS algorithm of Kuhn, Moscibroda, and Wattenhofer [11].

1.1 Extensions to our Main Result

Here we briefly report on extensions to our main result that we have obtained. They do not appear in this paper due to lack of space.

1. Let $G = (V, E)$ be an edge-weighted graph. For any $t > 1$ and positive integer k, a *k-vertex fault-tolerant t-spanner* of G is a spanning subgraph G' if for each subset S of vertices of size at most k, $G' - S$ is a t-spanner of $G[V - S]$. A k-edge fault-tolerant t-spanner is defined in a similar manner. Using ideas from [2] we can extend our algorithm to produce a k-vertex (or a k-edge) fault-tolerant t-spanner in polylogarithmic number of communication rounds.

2. In this paper, we use Euclidean distances as edge weights for the edges of the input graph G. However, if the metric $c \cdot |uv|^\gamma$, for positive constant c and $\gamma \geq 1$, is used in place of Euclidean distances $|uv|$, we can show that our algorithm still produces a spanner with all three desired properties. Relative Euclidean distances, such as the function mentioned above, may be used to produce *energy spanners*.

3. Let $G = (V, E)$ be an edge-weighted graph. The *power cost* of a vertex $u \in V$, denoted $power(u)$, is $\max\{w(u, v) \mid v \text{ is a neighbor of } u\}$. In other words, the power cost of a vertex u is proportional to the cost of u transmitting to a farthest neighbor. The *power cost* of G is $\sum_{u \in V} power(u)$ [8]. We can show that the output of our algorithm is not only lightweight with respect to the usual weight measure (sum of the weights of all edges) but also with respect to the power cost measure.

2 Sequential Relaxed Greedy Algorithm

In this section, we show that a "relaxed version" of `SEQUENTIAL-GREEDY` produces an output G' with all three desired properties, even when the input is not a complete Euclidean graph, but is a d-dimensional, α-qUBG for fixed d and α. Relaxing the requirement in `SEQUENTIAL-GREEDY` that the edges be totally ordered by length and allowing for the output to be updated lazily are critical to obtaining a distributed algorithm that runs in polylogarithmic number of rounds.

Let $r > 1$ be a constant to be fixed later and let $W_i = r^i \alpha / n$ for each $i = 0, 1, 2, \ldots$. Let $I_0 = (0, \alpha/n]$ and for each $i = 1, 2, \ldots$ let $I_i = (W_{i-1}, W_i]$. Let $m = \lceil \log_r \frac{n}{\alpha} \rceil$. Then, since there is no edge of length greater than 1, the length of any edge in E lies in one of the intervals $I_0, I_1, \ldots, I_m$. Let $E_i = \{\{u, v\} \in E : |uv| \in I_i\}$.

We now eliminate the restriction that edges within a set E_i be processed in increasing order by length. We run `SEQUENTIAL-GREEDY` in $m + 1$ phases: in phase i, the algorithm processes edges in E_i in arbitrary order and adds a subset of edges in E_i to the spanner. For $0 \leq i \leq m + 1$, we use G_i to denote the spanning subgraph of G consisting of edges $E_0 \cup E_1 \cup \cdots \cup E_i$. Thus G_i is the portion of the input graph that the algorithm has processed in phase i and earlier. We use G'_i to denote the output of the algorithm at the end of phase i. In other words, G'_i is the spanning subgraph of G consisting of edges of G that the algorithm has decided to retain in phases $0, 1, \ldots, i$. The final output of the algorithm is $G' = G_{m+1}$.

The way E_0 is processed is different from the way E_i, $i > 0$ is processed. We now separately describe these two parts.

2.1 Processing Edges in E_0

We start by stating a property of G_0 that follows easily from the fact that all edges in G_0 are small.

Lemma 1 *Every connected component of G_0 induces a clique in G.*

The algorithm `PROCESS-SHORT-EDGES` for processing edges in E_0 consists of three steps (i) determine the connected components of G_0, (ii) use `SEQUENTIAL-GREEDY` to compute a t-spanner for each connected component (that is, a clique), and (iii) let G'_0 be the union of the t-spanners computed in Step (2) and output G'_0. The following theorem states the correctness of the `PROCESS-SHORT-EDGES` algorithm. Its proof follows easily from the correctness of `SEQUENTIAL-GREEDY`.

Theorem 2 *G'_0 satisfies the following properties. (i) For every edge $\{u, v\} \in E_0$, G'_0 contains a uv-path of length at most $t \cdot |uv|$, (ii) $\Delta(G'_0) = O(1)$, and (iii) $w(G'_0) = O(w(MST(G)))$.*

2.2 Processing Long Edges

We now describe how edges in E_i are processed, for $i > 0$. The algorithm `PROCESS-LONG-EDGES` has five steps: (i) computing a cluster cover for G'_{i-1}, (ii) selecting query edges in E_i, (iii) computing a cluster graph H_{i-1} for G'_{i-1}, (iv) answering shortest path queries for the query edges selected in Step (ii), and (v) removing redundant edges. These steps are described in the next five subsections.

For any graph J, let $V(J)$ denote the vertex set for J. For any pair of vertices $u, v \in V(J)$ let $\mathtt{sp}_J(u, v)$ denote the length of a shortest uv-path in J. Define a *cluster* of J with *center* $u \in V(J)$ and *radius* r to be a set of vertices $C_u \subseteq V(J)$ such that, for each $v \in C_u$, $\mathtt{sp}_J(u, v) \leq r$. A set of clusters $\{C_{u_1}, C_{u_2}, \ldots\}$ of J is a *cluster cover of J of radius r* if every cluster in the set has radius r, every vertex in $V(J)$ belongs to at least one cluster, and for any pair of cluster centers u_i and u_j, $\mathtt{sp}_J(u_i, u_j) > r$.

2.2.1 Computing a Cluster Cover for G'_{i-1}

At the beginning of phase i we compute a cluster cover of radius δW_{i-1}, where $\delta < 1$ is a constant that will be fixed later. We start with an arbitrary vertex $u \in V$ and run Dijkstra's shortest path algorithm with source u on G'_{i-1}, in order to identify nodes $v \in V$ with the property that $\mathtt{sp}_{G'_{i-1}}(u, v) \leq \delta W_{i-1}$; each such node v gets included in the cluster C_u. Once C_u has been identified, recurse on $V \setminus C_u$ until all nodes belong to some cluster and we have a cluster cover of G'_{i-1} of radius δW_{i-1}.

2.2.2 Selecting Query Edges in E_i

As defined earlier, edges in E_i have weights in the interval $I_i = (W_{i-1}, W_i]$, while the cluster cover for G_{i-1} has radius δW_{i-1}. This implies that each edge in E_i has endpoints in different clusters. Our goal is to select a unique query edge per pair of clusters. This will guarantee that there are a constant number of query edges incident on any node (see Lemma 4) and this fact will be critically used by the distributed version of our algorithm to guarantee the degree bound on the spanner that is constructed.

Let θ be a quantity that satisfies $0 < \theta < \frac{\pi}{4}$ and $t \geq \frac{1}{\cos\theta - \sin\theta}$. Define an edge $e =$

$\{x, y\} \in E_i$ to be a *covered edge* if there is a $z \in V$ such that (i) $\{x, z\} \in G'_{i-1}$, $|yz| \leq \alpha$ and $\angle yxz \leq \theta$ or (ii) $\{y, z\} \in G'_{i-1}$, $|xz| \leq \alpha$ and $\angle xyz \leq \theta$. Any edge in E_i that is not covered is a *candidate* query edge. The motivation for these definitions is the following geometric lemma, due to Czumaj and Zhao [2]. Using this lemma, it is easy to see that for any covered edge $\{x, y\} \in E_i$, G'_{i-1} already contains an xy-path of length at most $t \cdot |xy|$ and hence covered edges need not be queried. Therefore, we take the complement of the set of covered edges and start with these as candidate query edges.

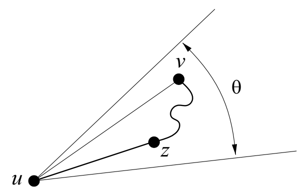

Figure 1: (a) Edge $\{u, v\}$ is *covered*: $\{u, z\}$ followed by a t-spanner zv-path is a t-spanner uv-path.

Lemma 3 (Czumaj and Zhao [2]) *Let $0 < \theta < \frac{\pi}{4}$ and $t \geq \frac{1}{\cos\theta - \sin\theta}$. Let u, v, z be three points in $\mathcal{R}^d$ with $\angle vuz \leq \theta$. Suppose further that $|uz| \leq |uv|$. Then the edge $\{u, z\}$ followed by a t-spanner path from z to v is a t-spanner path from u to v (see Figure 1).*

For each pair of clusters C_a and C_b, let $E_i[C_a, C_b]$ denote the subset of candidate query edges in E_i with one endpoint in C_a and the other endpoint in C_b. Our algorithm selects a unique query edge $\{x, y\}$ from each nonempty subset $E_i[C_a, C_b]$. Assuming that $x \in C_a$ and $y \in C_b$, the edge $\{x, y\}$ is selected so as to minimize

$$t \cdot |xy| - \mathtt{sp}_{G'_{i-1}}(a, x) - \mathtt{sp}_{G'_{i-1}}(b, y) \tag{1}$$

In Section 2.3 we show that it suffices to answer shortest path queries on the selected query edges to ensure that G'_i is a t-spanner of G_i at the end of phase i.

The following lemma shows that selecting query edges as described above filters all but a constant number of edges per cluster. The proof follows from two observations: (i) if a pair of cluster centers are connected by an edge, then the clusters are not too far from each other in Euclidean space and (ii) the Euclidean distance between any pair of cluster centers is bounded from below, because they would otherwise be part of the same cluster.

Lemma 4 *The number of query edges in E_i that are incident on any cluster is $O(1)$.*

2.2.3 Computing a Cluster Graph

For each selected query edge $\{x, y\} \in E_i$, we need to know if G'_{i-1} contains an xy-path of length at most $t \cdot |xy|$. In general, the number of hops in a shortest xy-path in G'_{i-1} can be quite large and having to traverse such a path would mean that the shortest path query corresponding to edge $\{x, y\}$ could not be answered quickly enough. To get around this problem, we use an idea from [4] in which the authors construct an approximation to G'_{i-1}, called a *cluster graph*, and show that for any edge $\{x, y\} \in E_i$, the shortest path query for $\{x, y\}$ can be answered approximately on H_{i-1} in a constant number of steps. The goal of Das and Narasimhan [4] was to improve the running time of `SEQUENTIAL-GREEDY` on complete Euclidean graphs, but we show that the Das-Narasimhan data structure can be constructed and maintained in a distributed fashion for efficiently answering shortest path queries for edges belonging to a α-qUBG. In the following, we describe a sequential algorithm that starts with a cluster cover of G'_{i-1} of radius δW_{i-1}, and builds a *cluster graph* H_{i-1} of G'_{i-1}. This algorithm is identical to the one in Das and Narasimhan [4] and is included mainly for completeness.

The vertex set of H_{i-1} is V and the edge set of H_{i-1} contains two types of edges: *intra-cluster* edges and *inter-cluster* edges. An edge $\{a, x\}$ is an intra-cluster edge if a is a cluster center and x is node in C_a. Inter-cluster edges are between cluster centers. An edge $\{a, b\}$ is an inter-cluster edge if a and b are cluster centers, and at least one of the following two conditions holds: (i) $\mathtt{sp}_{G'_{i-1}}(a, b) \leq W_{i-1}$, or (ii) there is an edge in G'_{i-1} with one endpoint in C_a and the other endpoint in C_b. See Figure 2.

Regardless of the type of a cluster edge $e = \{a, b\}$ (inter- or intra-), the weight of e is the value of $sp_{G'_{i-1}}(a, b)$. The following lemma follows easily from the definition of inter-cluster edges.

Lemma 5 *For any inter-cluster edge $\{a, b\}$ in H_{i-1}, $\mathtt{sp}_{G'_{i-1}}(a, b) \leq (2\delta + 1)W_{i-1}$.*

The above upper bound also implies that $|ab| \leq (2\delta + 1)W_{i-1}$. Using this and arguments

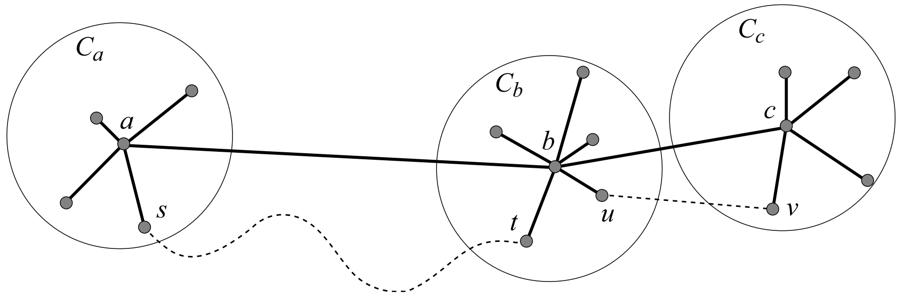

Figure 2: Edges interior to disks are *intra-cluster* edges. Edge $\{a, b\}$ is an *inter-cluster* edge because $\texttt{sp}_{G'_{i-1}}(a, b) \leq W_{i-1}$, and $\{b, c\}$ is an *inter-cluster* edge because $\{u, v\}$ is in G'_{i-1}. An st-path in G'_{i-1}, shown by the dashed curve may be approximated by the path s, a, b, t in H_{i-1}.

similar to those used for Lemma 4, we obtain the following lemma.

Lemma 6 *Each cluster center is incident on a constant number of inter-cluster edges in H_{i-1}.*

The main reason for constructing the cluster graph H_{i-1} is that lengths of paths in H_{i-1} are close to lengths of corresponding paths in G'_{i-1} and shortest path queries for edges in E_i can be answered quickly in H_{i-1}. The following lemma (whose proof appears in Das and Narasimhan [4]) shows that we can construct H_{i-1} such that path lengths in H_{i-1} approximate path lengths in G'_{i-1} to any desired extent, depending on the choice of δ.

Lemma 7 *For any edge $\{x, y\} \in E_i$, if there is a path between x and y in G'_{i-1} of length L_1, then there is a path between x and y in H_{i-1} of length L_2 such that $L_1 \leq L_2 \leq \frac{1+6\delta}{1-2\delta} L_1$.*

2.2.4 Answering Shortest Path Queries

For query edges $\{x, y\} \in E_i$, we are interested in knowing whether G'_{i-1} has an xy-path of length at most $t \cdot |xy|$. We ask this question on the cluster graph H_{i-1}. If H_{i-1} contains an xy-path of length at most $t \cdot |xy|$, we do not add $\{x, y\}$ to G'_i; otherwise we do. If H_{i-1} contains an xy-path of length at most $t \cdot |xy|$, then so does G'_{i-1} (by Lemma 7, since $L_1 \leq L_2$). Therefore, not adding $\{x, y\}$ to the spanner is not a dangerous choice. On the other hand, even if H_{i-1} does not contain an xy-path of length at most $t \cdot |xy|$, G'_{i-1} might contain such a path and in this case adding edge $\{x, y\}$ is unnecessary. Adding extra edges is of course not problematic for the t-spanner property. It will turn out that this is not a problem even for the requirement that the spanner should have bounded degree and small weight, given that paths in H_{i-1} can approximate paths in G'_{i-1} to an arbitrary degree.

Given the structure of the cluster graph, all but at most 2 edges in any simple xy-path are inter-cluster edges. Since the radius of each cluster is δW_{i-1}, each inter-cluster edge has weight greater than δW_{i-1}. We are looking for a path of length at most $t \cdot |xy|$. Since $|xy| \in (W_{i-1}, W_i]$, we are looking for a path of length at most $t \cdot W_i = t \cdot r \cdot W_{i-1}$. Any simple path in H_{i-1} of length at most $t \cdot r \cdot W_{i-1}$ has at most $2 + tr/\delta$ hops, which is a constant. This yields the following lemma.

Lemma 8 *For any edge $\{x, y\} \in E_i$, if $\mathtt{sp}_{H_{i-1}}(x, y) \leq t \cdot |xy|$, then H_{i-1} contains a shortest xy-path with a constant number of hops.*

One issue we need to deal with, especially when attempting to construct and answer queries in H_{i-1} in a distributed setting, is that edges in H_{i-1} need not be present in the underlying network G. Specifically, for an intra-cluster edge $\{u, a\}$, where C_a is a cluster and $u \in C_a$, it may be the case that $|ua| > \alpha$ and $\{u, a\}$ may be absent from G. Similarly, an inter-cluster edge $\{a, b\}$ in H_{i-1} may be absent in G. However, for any edge $\{x, y\}$ in H_{i-1} (intra- or inter-cluster edge), we have the bound $\mathtt{sp}_{G'_{i-1}}(x, y) \leq (2\delta + 1)W_{i-1}$. This follows from Lemma 5 and the fact that the radius of each cluster is δW_{i-1}. Thus a shortest xy-path in G'_{i-1} lies entirely in a ball of radius $(2\delta + 1)W_{i-1}$ centered at x. Since G'_{i-1} is a spanning subgraph of G, this implies that there is a shortest xy-path P in G that lies entirely in the d-dimensional ball of radius $(2\delta + 1)W_{i-1}$ centered at x. Since any two vertices in P that are two hops away from each other are at least α apart (in the d-dimensional Euclidean space), P contains at most $2(2\delta + 1)W_{i-1}/\alpha < 2(2\delta + 1)/\alpha$ hops. This argument along with Lemma 8 yields the following theorem.

Theorem 9 *For any edge $\{x, y\} \in E_i$, if $\mathtt{sp}_{H_{i-1}}(x, y) \leq t \cdot |xy|$, then G contains a shortest xy-path with a constant number of hops.*

This theorem implies that brute force search initiated from one of the endpoints, say x, will be able to answer the shortest path query on edge $\{x, y\}$ in $O(1)$ rounds in a distributed setting.

2.2.5 Removing Redundant Edges

Recall that shortest path queries for edges in E_i are answered on H_{i-1}, and so updates to G'_i in phase i do not influence subsequent shortest path queries in phase i. Thus it is possible that in phase i two edges $\{u, v\}$ and $\{u', v'\}$ get added to G_i, yet both of the following hold:

(i) $\mathtt{sp}_{H_{i-1}}(v, u') + |u'v'| + \mathtt{sp}_{H_{i-1}}(v', u) \leq t \cdot |uv|$

(ii) $\mathtt{sp}_{H_{i-1}}(v', u) + |uv| + \mathtt{sp}_{H_{i-1}}(v, u') \leq t \cdot |u'v'|$

Note that, since $\mathtt{sp}_{G'_{i-1}}(x, y) \leq \mathtt{sp}_{H_{i-1}}(x, y)$ holds for any pair of nodes x and y, conditions (i) and (ii) above imply that G'_i contains t-spanner paths from u to v and from u' to v'. We call two edges $\{u, v\}$ and $\{u', v'\}$ satisfying conditions (i) and (ii) above *mutually redundant*: one of them could potentially be eliminated from G_i, without compromising the t-spanner property of G_i. In fact, such mutually redundant pairs of edges need to be eliminated from G'_i because our proof that G' has small weight (Theorem 12) depends on the absence of such pairs of edges.

To do this, we build a graph J that has a node for each edge in a mutually redundant pair and an edge between every pair of nodes that correspond to a mutually redundant pair of edges in G'_i. We construct an MIS I of J and eliminate from G'_i all edges associated with nodes in J that do not appear in I.

2.3 The Three Desired Properties

Let $G' = G'_{m+1}$ be the spanner at the end of phase $m + 1$. We now prove that G' satisfies the three properties that the output of `SEQUENTIAL-GREEDY` was guaranteed to have. The proofs of these theorems form the technical core of the paper. Unfortunately, due to space restrictions, the proofs do not appear in the main body of the paper. However, they do appear in full detail in the Appendix.

Theorem 10 *For any $0 < \delta \leq \frac{t-1}{4}$, the output G' is a t-spanner.*

Theorem 11 *G' has $O(1)$ degree.*

Theorem 12 *Let* $0 < \delta < (t-1)/(6+2t)$. *Let* t_δ *denote* $t \cdot (1-2\delta)/(1+6\delta)$. *Let* $1 < r < (t_\delta + 1)/2$. *When the relaxed greedy algorithm is run with these values of* δ *and* r*, the output* G' *satisfies* $w(G') = O(wt(MST(G)))$.

3 Distributed Relaxed Greedy Algorithm

We now describe a distributed version of the relaxed greedy algorithm from Section 2. Like the sequential relaxed greedy algorithm, this algorithm also runs in $O(\log n)$ phases — with edges in E_i being processed in phase i. We will show that edges in E_0 can be processed in $O(1)$ rounds. Recall that each subsequent phase consists of the following five steps: (i) computing a cluster cover of G'_{i-1}, (ii) selecting query edges in E_i, (iii) computing a cluster graph H_{i-1} of G'_{i-1}, (iv) answering shortest path queries for selected query edges, and (v) deleting some redundant edges. We will show that Steps (ii), (iii), and (iv) can be completed in $O(1)$ rounds and Steps (i) and (v) take $O(\log^* n)$ rounds. Step (i) and Step (v) will each involve computing an MIS in a certain derived graph and in both cases, we will show that the derived graph is a UBG that resides in a metric space of constant doubling dimension. Putting this all together, we will show that the algorithm runs in $O(\log n \cdot \log^* n)$ communication rounds.

3.1 Distributed Processing of Short Edges

Lemma 1 implies that vertices in the same component of $G_0 = G[E_0]$ induce a clique and therefore can communicate in one hop with each other. In the distributed version of the algorithm, each vertex u obtains the topology of its closed neighborhood along with pairwise distances between neighbors in one hop. Using this information, u determines the connected component C of G_0 that it belongs to. Then u simply runs `SEQUENTIAL-GREEDY` on C and computes a t-spanner of C. Finally, u identifies the edges of the t-spanner incident on itself and informs all its neighbors of this.

Theorem 13 *The edges in* E_0 *can be processed in* $O(1)$ *rounds of communication.*

3.2 Distributed Processing of Long Edges

In this section, we show how long edges, that is, edges in E_i, $i > 0$, can be processed in a distributed setting. The first step of this process is the computation of a cluster cover for the spanner at the end of the previous phase, G'_{i-1}.

3.2.1 Distributed Cluster Cover for G'_{i-1}

Recall that in this step our goal is to compute a cluster cover $\{C_{u_1}, C_{u_2}, \ldots\}$ of G'_{i-1} of radius δW_{i-1}. To do this, each node u first identifies all nodes v in G satisfying $\mathtt{sp}_{G'_{i-1}}(u,v) \le \delta W_{i-1}$. Using arguments similar to those in Section 2.2.4, it is easy to see that any node v satisfying $\mathtt{sp}_{G'_{i-1}}(u,v) \le \delta W_{i-1}$ must be at most $2\delta W_{i-1}/\alpha$ hops from u. So each node u constructs the subgraph of G'_{i-1} induced by nodes that are at most $2\delta W_{i-1}/\alpha$ hops away from it in G. Node u then runs a (sequential) single source shortest path algorithm with source u on the local view of G'_{i-1} it has obtained and identifies all nodes v satisfying $\mathtt{sp}_{G'_i}(u,v) \le \delta W_{i-1}$.

At the end of the above process, every node u in the network is a cluster center. We now force some nodes to cease being cluster centers, so that all pairs of cluster centers are far enough from each other. Let J be the graph with vertex set V and whose edges $\{x,y\}$ are such that $x \in C_y$ (and by symmetry, $y \in C_x$). If $\{x,y\}$ is an edge in J, it is the case that $\mathtt{sp}_{G'_{i-1}}(x,y) \le \delta W_{i-1}$. Now assign to every pair of nodes $\{x,y\}$ in V a weight $w(x,y) = \mathtt{sp}_{G'_{i-1}}(x,y)$. The weights w form a metric simply because shortest path distances in any graph form a metric. Thus J is a graph whose nodes reside in a metric space and whose edges connect pairs of nodes separated by distance of at most δW_{i-1} (in the metric space). By scaling the quantity δW_{i-1} up to one, we see that J is a UBG in the underlying metric space defined by the weights w. Recall from [11] that the *doubling dimension* of a metric space is the smallest ρ such that every ball can be covered by at most 2^ρ balls of half the radius. To see that the metric space induced by the weights w has constant doubling dimension, start with a ball of radius R centered at an arbitrary vertex u. Every vertex v in this ball satisfies $\mathtt{sp}_J(u,v) \le R$. Now cover the vertices in this ball using balls of radius $R/2$ as follows: repeatedly pick an uncovered vertex v in the radius-R ball and grow a radius $R/2$ ball centered at v. It is easy to see that the number of radius $R/2$ balls is bounded because any pair of centers of these balls are far apart.

Lemma 14 *J is a UBG that resides in a metric space of constant doubling dimension.*

Let I be an MIS of J constructed using the MIS algorithm in [11]. This algorithm runs in $O(\log^* n)$ communication rounds on a UBG that resides in a metric space of constant doubling dimension. Then each node in $V \setminus I$ has one or more neighbors in I. Each node $u \in I$ is declared a cluster center, and each node $v \in V \setminus I$ attaches itself to the neighbor in I with the highest identifier. This gives us the desired cluster cover of radius δW_{i-1}.

Theorem 15 *A cluster cover of G'_{i-1} of radius δW_{i-1} can be computed in $O(log^* n)$ rounds of communication.*

3.2.2 Distributed Query Edge Selection

Only nodes that are cluster heads need to participate in the process of selecting query edges. Each cluster head a seeks to gather information on all edges in E_i between the cluster C_a and any other cluster C_b. Using the argument in Section 2.2.4, we know that every node in C_a is at most $2\delta W_{i-1}/\alpha$ hops away from a in G. Therefore, if there is an edge $\{u, v\} \in E_i$, $u \in C_a$ and $v \in C_b$, then v is at most $1 + 2\delta W_{i-1}/\alpha$ hops away from a. So a gets information from nodes that are at most $1+2\delta W_{i-1}/\alpha$ hops away from it and it identifies all edges in $E_i[C_a, C_b]$. Recall that this is the set of edges in E_i which connect a node in C_a and a node in C_b. Node a then discards all covered edges from $E_i[C_a, C_b]$, leaving only candidate query edges in E_i between C_a and C_b. Finally, from among the candidate query edges, node a selects an edge $\{u, v\}$ that minimizes $t \cdot |uv| - \mathtt{sp}_{G'_{i-1}}(a, u) - \mathtt{sp}_{G'_{i-1}}(b, v)$.

Theorem 16 *Query edges from E_i can be selected in $O(1)$ rounds of communication.*

3.2.3 Distributed Construction of the Cluster Graph

As in the query edge selection step, only the cluster heads need to perform actions to compute the cluster graph. Any member u of a cluster C_a lies at most $2\delta W_{i-1}/\alpha$ hops away from a in G. Thus a can identify intra-cluster edges incident on it by gathering information from at most $2\delta W_{i-1}/\alpha$ hops away. If C_b is a cluster with $\mathtt{sp}_{G'_{i-1}}(a, b) \leq W_{i-1}$, then node a can identify the inter-cluster edge $\{a, b\}$ by gathering information from at most $2W_{i-1}/\alpha$ hops away. If C_b is a cluster such that there is an edge $\{u,v\}$ in G'_{i-1} with $u \in C_a$ and $v \in C_b$, then node a can identify the inter-cluster edge $\{a,b\}$ by gathering information from at most $2(2\delta+1)W_{i-1}/\alpha$ hops away. Note that the information that a gathers contains a local view of G'_{i-1} along with all pairwise distances. Using this information, node a is able to run a single source shortest path algorithm with source a and determine the weights of all inter-cluster and intra-cluster edges incident on a.

Theorem 17 *Computing the cluster graph H_{i-1} of G'_{i-1} takes $O(1)$ communication rounds.*

3.2.4 Answering Shortest Path Queries

Each node u knows all the query edges incident on it. As proved in Section 2.2.4, node u only needs to gather information from nodes that are at most a constant number of hops away, to be able to determine locally, for all incident query edges $\{u,v\} \in E_i$, whether $\mathtt{sp}_{H_{i-1}}(u,v) \leq t \cdot |uv|$. Thus, after constant number of communication rounds, u knows the subset of incident query edges $\{u,v\}$ for which $\mathtt{sp}_{H_{i-1}}(u,v) > t \cdot |uv|$ and u identifies these as the incident edges to be added to G'_i.

Theorem 18 *Answering shortest path queries takes $O(1)$ communication rounds.*

3.2.5 Distributed Removal of Redundant Edges

Two edges $\{u,v\}$ and $\{u',v'\}$ in G'_i are mutually redundant if (i) $\mathtt{sp}_{H_{i-1}}(v,u') + |u'v'| + \mathtt{sp}_{H_{i-1}}(v',u) \leq t \cdot |uv|$ and (ii) $\mathtt{sp}_{H_{i-1}}(v',u) + |uv| + \mathtt{sp}_{H_{i-1}}(v,u') \leq t \cdot |u'v'|$. Each node u takes charge of all edges $\{u,v\}$ added to G_i in phase i and for which the identifier of u is higher than the identifier of v. For each such edge $\{u,v\}$ that u is in charge of, u determines all edges $\{u',v'\}$ such that $\{u,v\}$ and $\{u',v'\}$ form a mutually redundant pair. Note that the nodes u and v' are a constant number of hops away from each other in G. Similarly, nodes v and u'. Node u then contributes to the construction of the graph J by adding to $V(J)$ a vertex for each redundant edge u is in charge of, and to $E(J)$ an edge connecting nodes in $V(J)$ that correspond to mutually redundant edges in G_i. Using an argument similar to the one used in Lemma 14, we can show the following property of J:

Lemma 19 *J is a UBG that resides in a metric space of constant doubling dimension.*

Let I be an MIS of J constructed using the MIS algorithm in [11] that takes $O(\log^* n)$ communication rounds on a UBG that resides in a metric space of constant doubling dimension. Each node u then removes from G_i all incident edges in $V(J) \setminus I$.

Theorem 20 *Removing redundant edges takes $O(\log^* n)$ communication rounds.*

References

[1] L. Barriére, P. Fraigniaud, and L. Narayanan. Robust position-based routing in wireless ad hoc networks with unstable transmission ranges. In *Proceedings of the 5th International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications (DIALM)*, pages 19–27, 2001.

[2] A. Czumaj and H. Zhao. Fault-tolerant geometric spanners. *Discrete & Computational Geometry*, 32(2):207–230, 2004.

[3] G. Das, P. Heffernan, and G. Narasimhan. Optimally sparse spanners in 3-dimensional euclidean space. In *SCG '93: Proceedings of the ninth annual symposium on Computational geometry*, pages 53–62, New York, NY, USA, 1993. ACM Press.

[4] G. Das and G. Narasimhan. A fast algorithm for constructing sparse euclidean spanners. *Int. J. Comput. Geometry Appl.*, 7(4):297–315, 1997.

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

[6] M. Hajiaghayi, N. Immorlica, and V. S. Mirrokni. Fault-tolerant and 3-dimensional distributed topology control algorithms in wireless multi-hop networks. In *Proceedings of the 11th IEEE International Conference on Computer Communications and Networks (IC3N)*, pages 392–398, 2002.

[7] M. Hajiaghayi, N. Immorlica, and V. S. Mirrokni. Power optimization in fault-tolerant topology control algorithms for wireless multi-hop networks. In *Proceedings of the 9th annual international conference on Mobile computing and networking (MobiCom)*, pages 300–312, 2003.

[8] M. Hajiaghayi, G. Kortsarz, V.S. Mirrokni, and Z. Nutov. Power optimization for connectivity problems. In *IPCO*, pages 349–361, 2005.

[9] B. Karp and H. T. Kung. Greedy perimeter stateless routing for wireless networks. In *Proceedings of the Sixth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom)*, pages 243–254, 2000.

[10] David Kotz, Calvin Newport, and Chip Elliot. The mistaken axioms of wireless-network research. Technical Report TR2003-467, Dartmouth College, Department of Computer Science, 2003.

[11] Fabian Kuhn, Thomas Moscibroda, and Roger Wattenhofer. On the locality of bounded growth. In *Proceedings of the 24th ACM Symposium on the Principles of Distributed Computing (PODC)*, pages 60–68, 2005.

[12] C. Levcopoulos and A. Lingas. There are planar graphs almost as good as the complete graphs and almost as cheap as minimum spanning trees. *Algorithmica*, 8:251–256, 1992.

[13] X. Y. Li, G. Calinescu, and P. Wan. Distributed construction of planar spanner and routing for ad hoc wireless networks. In *Proceedings of the 21st Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, 2002.

[14] X. Y. Li, G. Calinescu, P. J. Wan, and Y. Wang. Localized delaunay triangulation with application in ad hoc wireless networks. *IEEE Trans. Parallel Distrib. Syst.*, 14(10):1035–1047, 2003.

[15] Xiang-Yang Li and Yu Wang. Efficient construction of low weighted bounded degree planar spanner. *International Journal of Computational Geometry and Applications*, 14(1–2):69–84, 2004.

[16] David Peleg. *Distributed computing: a locality-sensitive approach.* Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2000.

[17] R. Rajaraman. Topology control and routing in ad hoc networks: A survey. *SIGACT News*, 33:60–73, 2002.

[18] Y. Wang and X. Y. Li. Localized construction of bounded degree and planar spanner for wireless ad hoc networks. In *Proceedinngs of the Joint Workshop on Foundations of Mobile Computing (DIALM-POMC)*, pages 59–68, 2003.

[19] R. Wattenhofer and A. Zollinger. XTC: A practical topology control algorithm for ad-hoc networks. In *4th International Workshop on Algorithms for Wireless, Mobile, Ad Hoc and Sensor Networks (WMAN)*, 2004.

Appendix

Let $G' = G'_{m+1}$ be the spanner at the end of phase $m+1$. We now prove that G' satisfies the three properties that the output of `SEQUENTIAL-GREEDY` was guaranteed to have.

Theorem 21 *For any $0 < \delta \leq \frac{t-1}{4}$, the output G' is a t-spanner.*

Proof: We first prove that the theorem holds for all query edges in E, then we extend the argument to non-query edges as well. Let $\{x, y\}$ be an arbitrary query edge and let i be such that $\{x, y\} \in E_i$. Then either (i) $\{x, y\}$ is added to the spanner in phase i, or (ii) $\mathtt{sp}_{H_{i-1}}(x, y) \leq t \cdot |xy|$. If the former is true and $\{x, y\}$ is not a redundant edge, then the theorem holds. If $\{x, y\}$ is a redundant edge but does not get removed from G_i, then again the theorem holds. If $\{x, y\}$ is a redundant edge that gets removed from G_i, then at least one mutually redundant counterpart edge must remain in G_i (since removed edges form an independent set), ensuring a t-spanner xy-path in G_i. If (ii) is true, then from Lemma 7, $\mathtt{sp}_{G'_{i-1}}(x, y) \leq \mathtt{sp}_{H_{i-1}}(x, y)$ (first part of the inequality) and therefore $\mathtt{sp}_{G'_{i-1}}(x, y) \leq t \cdot |xy|$.

For non-query edges, the proof is by induction on the length of edges in G. The base case corresponds to edges in E_0, for which `SEQUENTIAL-GREEDY` ensures that the theorem holds.

Assume that the theorem is true for any edge in E of length no greater than some value s, and consider a smallest non-query edge $\{x, y\}$ in G of length greater than s. We prove that $\mathtt{sp}_{G'}(x, y) \leq t \cdot |xy|$. Let i be such that $\{x, y\} \in E_i$. We now consider two cases, depending on whether $\{x, y\}$ is a *candidate* query edge in phase i or not. If $\{x, y\}$ is not a candidate query edge, then it is a covered edge. That is, there exists an edge $\{x, z\}$ in G'_{i-1} such that $|yz| \leq \alpha$ and $\angle yxz \leq \theta$ or an edge $\{y, z\}$ in G'_{i-1} such that $|xz| \leq \alpha$ and $\angle xyz \leq \theta$. The two cases are symmetric and so without loss of generality, assume that the former is true. Here θ satisfies the hypothesis of the Czumaj-Zhao lemma (Lemma 3), that is, $0 < \theta < \frac{\pi}{4}$ and $t \geq \frac{1}{\cos\theta - \sin\theta}$. Since $|yz| \leq \alpha$ and G is an α-qUBG, this implies that $\{y, z\}$ is an edge is E. Furthermore, since $0 < \theta < \frac{\pi}{4}$, we have $|yz| < |xy|$. Refer to Figure 3a. If $\{y, z\}$ is a query edge, then by the argument above we have that G' contains a t-spanner yz-path p. Otherwise, if $\{y, z\}$ is not a query edge, since its length is less than the length of $\{x, y\}$, by the inductive hypothesis we get that there is a t-spanner yz-path p. In either case, Lemma 3 tells us that $\{x, z\}$ followed by p is a t-spanner path from x to y, completing this case.

We now consider the case when $\{x, y\}$ is a candidate query edge in phase i, but not a query edge. Let a and b be such that $x \in C_a$ and $y \in C_b$, and let $\{u, v\}$ be the query edge selected in phase i between C_a and C_b, with $u \in C_a$ and $v \in C_b$. Refer to Figure 3b. Due to the criteria for selecting $\{u, v\}$, we have

$$t \cdot |uv| - \mathtt{sp}_{G'_{i-1}}(a, u) - \mathtt{sp}_{G'_{i-1}}(b, v) \leq t \cdot |xy| - \mathtt{sp}_{G'_{i-1}}(a, x) - \mathtt{sp}_{G'_{i-1}}(b, y). \tag{2}$$

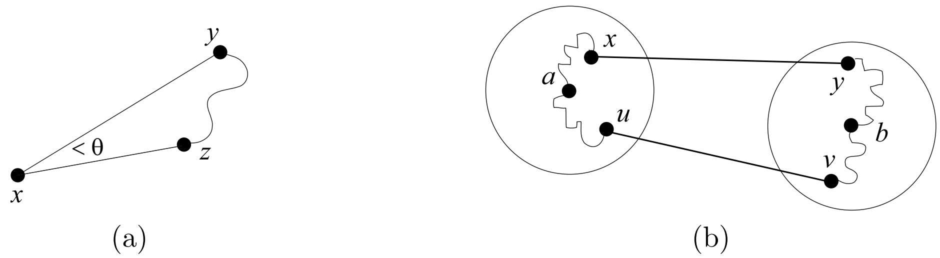

Figure 3: (a) $\{x, y\}$ is a covered edge (b) $\{u, v\}$ is a query edge: if G_i contains a t-spanner uv-path, then G_i contains a t-spanner xy-path.

Recall that G'_i is the partial spanner at the end of phase i. We show that $\mathtt{sp}_{G'_i}(x, y) \leq t \cdot |xy|$. We discuss two cases, depending on whether $\{u, v\}$ was added to G'_i or not.

Assume first that $\{u, v\}$ was not added to G'_i. This means that $\mathtt{sp}_{H_{i-1}}(u, v) \leq t \cdot |uv|$. Note however that

$$\mathtt{sp}_{H_{i-1}}(u, v) = \mathtt{sp}_{G'_{i-1}}(u, a) + \mathtt{sp}_{H_{i-1}}(a, b) + \mathtt{sp}_{G'_{i-1}}(b, v) \leq t \cdot |uv|. \tag{3}$$

We now evaluate

$$\begin{aligned} \mathtt{sp}_{G'_{i-1}}(x, y) &\leq \mathtt{sp}_{G'_{i-1}}(x, a) + \mathtt{sp}_{G'_{i-1}}(a, b) + \mathtt{sp}_{G'_{i-1}}(b, y) \\ &\leq \mathtt{sp}_{G'_{i-1}}(x, a) + \mathtt{sp}_{H_{i-1}}(a, b) + \mathtt{sp}_{G'_{i-1}}(b, y) \\ &\leq t \cdot |xy|. \end{aligned}$$

This latter inequality involves simple substitutions that use inequalities (2) and (3).

Assume now that $\{u, v\}$ was added to G'_i. Since $u \in C_a$ and C_a has radius δW_{i-1}, we have that $\mathtt{sp}_{G'_{i-1}}(a, u) \leq \delta W_{i-1}$. Similarly, $\mathtt{sp}_{G'_{i-1}}(b, v) \leq \delta W_{i-1}$. These together with (2) yield

$$t \cdot |uv| - 2\delta W_{i-1} \leq t \cdot |xy| - \mathtt{sp}_{G'_{i-1}}(a, x) - \mathtt{sp}_{G'_{i-1}}(b, y). \tag{4}$$

The existence of $\{u, v\}$ in G'_i enables us to construct in G'_i a path from a to b of weight

$$\begin{aligned} \mathtt{sp}_{G'_i}(a, b) &\leq \mathtt{sp}_{G'_i}(a, u) + |uv| + \mathtt{sp}_{G'_i}(v, b) \\ &\leq 2\delta W_{i-1} + |uv|, \end{aligned} \tag{5}$$

since $\mathtt{sp}_{G'_i}(a, u) \leq \mathtt{sp}_{G'_{i-1}}(a, u) \leq \delta W_{i-1}$, and same for $\mathtt{sp}_{G'_i}(v, b)$. We can now construct a path in G'_i from x to y of weight

$$
\begin{aligned}
\mathtt{sp}_{G'_i}(x,y) &\leq \mathtt{sp}_{G'_i}(a,x) + \mathtt{sp}_{G'_i}(b,y) + \mathtt{sp}_{G'_i}(a,b) \\
&\leq t\cdot|xy| + 2\delta W_{i-1} - t\cdot|uv| + \mathtt{sp}_{G'_i}(a,b) && \text{(substitute (4))} \\
&\leq t\cdot|xy| + 4\delta W_{i-1} - (t-1)\cdot|uv| && \text{(substitute (5))} \\
&< t\cdot|xy| + 4\delta W_{i-1} - (t-1)W_{i-1} && \text{(since } |uv| > W_{i-1}\text{)}
\end{aligned}
$$

Note that for any $\delta \leq \frac{t-1}{4}$, the quantity $4\delta W_{i-1} - (t-1)\cdot W_{i-1}$ above is negative, yielding $\mathtt{sp}_{G_i}(x,y) < t\cdot|xy|$. This completes the proof. $\square$

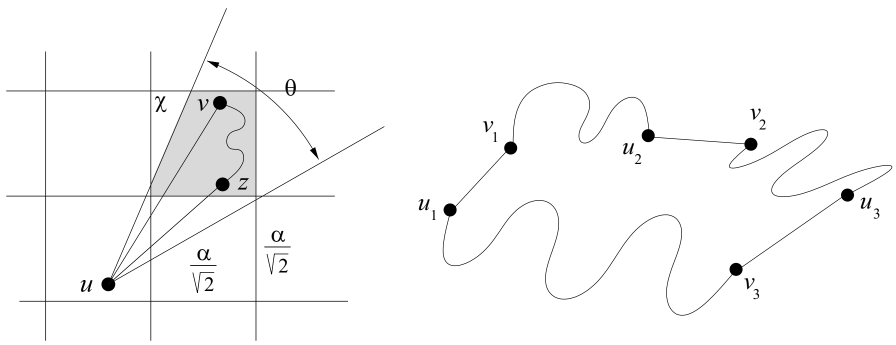

Figure 4: (a) Region χ contains two neighbors v and z of u. (b) Definition of the t-leapfrog property with $S = \{\{u_1,v_1\},\{u_2,v_2\},\{u_3,v_3\}\}$.

Theorem 22 *G' has $O(1)$ degree.*

Proof: Let θ be a quantity satisfying the conditions of Lemma 3. Fix a vertex u and consider the d-dimensional unit radius ball centered at u. For some T that depends only on θ and d, this ball can be partitioned into T cones, each with apex u, such that for any x, y in a cone, $\angle xuy \leq \theta$. Place an infinite axis-parallel grid of d-dimensional cubes, each of dimension $\frac{\alpha}{\sqrt{d}} \times \frac{\alpha}{\sqrt{d}} \times \cdots \times \frac{\alpha}{\sqrt{d}}$, on the plane. See Figure 4(c) for a 2-dimensional version of this picture. There are $O(1/\alpha^d)$ cells that intersect the unit ball centered at u, and therefore there are $O(1/\alpha^d)$ cells that intersect each cone in the cone partition of this unit ball. Thus the cones and the square cells together partition the unit ball centered at u into $O(T/\alpha^d)$ regions. We show that in G', u has a constant number of neighbors in each region.

Let $v_1, v_2, \ldots, v_k$ be neighbors of u in G' that lie in a region χ. Without loss of generality, assume that $|uv_1| \geq |uv_j|$, for $j = 2,\ldots,k$, and let i be such that $\{u,v_1\} \in E_i$. Since $|uv_j| \leq |uv_1|$, we have that for all $j = 2,\ldots,k$, $\{u,v_j\} \in E_\ell$, with $\ell \leq i$.

We now prove that $\{u,v_j\}$ is in fact in E_i for all j. To derive a contradiction, assume that there is a $j > 1$ such that $\{u,v_j\} \in E_\ell$, with $\ell < i$. This means that just before edge $\{u,v_1\}$ is processed, G' contains edge $\{u,v_j\}$. Also note that since v_1 and v_j lie in the same region, $|v_1v_j| \leq \alpha$. But, this means that $\{u, v_1\}$ is a covered edge in phase i and will not be queried. This contradicts the presence of edge $\{u, v_1\}$ in G'.

We have shown that $\{u, v_j\} \in E_i$ for all j. Recall that our algorithm picks a unique query edge per pair of clusters. This along with Lemma 4 proves that k is constant. $\square$

In the next theorem, we show that the spanner produced by the algorithm has small weight. The proof relies on the line segments in the spanner satisfying a property known as the *leapfrog property* [5, 2]. For any $t \geq t' > 1$, a set of line segments, denoted F, has the (t', t)*-leapfrog property* if for every subset $S = \{\{u_1, v_1\}, \{u_2, v_2\}, \ldots, \{u_s, v_s\}\}$ of F

$$t' \cdot |u_1 v_1| < \sum_{i=2}^{s} |u_i v_i| + t \cdot \Big(\sum_{i=1}^{s-1} |v_i u_{i+1}| + |v_s u_1| \Big). \tag{6}$$

Informally, this definition says that if there exists an edge between u_1 and v_1, then any path not including $\{u_1, v_1\}$ must have length greater than $t'|u_1 v_1|$ (see Figure 4(c) for an illustration of this definition). The following implication of the (t', t)-leapfrog property was shown by Das and Narasimhan [4].

Lemma 23 (Das and Narasimhan [4]) *Let $t \geq t' > 1$. If the line segments F in d-dimensional space satisfy the (t', t)-leapfrog property, then $wt(F) = O(wt(MST))$, where MST is a minimum spanning tree connecting the endpoints of line segments in F. The constant in the asymptotic notation depends on t, t' and d.*

Theorem 24 *Let $0 < \delta < (t-1)/(6+2t)$. Let t_δ denote $t \cdot (1-2\delta)/(1+6\delta)$. Let $1 < r < (t_\delta + 1)/2$. When the relaxed greedy algorithm is run with these values of δ and r, the output G' satisfies $w(G') = O(wt(MST(G)))$.*

Proof: Let $\beta > 1$ be a constant picked as follows. When $t\alpha < 1$, pick β satisfying $1 < \beta < \min\{2, 1/(1-t\alpha)\}$. Otherwise, pick β satisfying $1 < \beta < 2$. Partition the edges of G' into subsets $F_0, F_1, \ldots$ such that $F_0 = \{\{u, v\} \in G' \mid |uv| \leq \alpha\}$ and for each $j > 0$, $F_j = \{\{u, v\} \in G' \mid \alpha\beta^{j-1} < |uv| \leq \alpha\beta^j\}$. Let $\ell = \lceil \log_\beta \frac{1}{\alpha} \rceil$. Then every edge in G' is in some subset F_j, $0 \leq j \leq \ell$. We will now show that each F_j satisfies the (t', t)-leapfrog property, for any t' satisfying:

$$1 \leq t' < \min\{\frac{t_\delta + 1}{r} - 1, \frac{2}{r}, \frac{t}{r}, \frac{2}{\beta}, t\alpha + \frac{1}{\beta}\}. \tag{7}$$

It is easy to check that our choice for δ, r, and β guarantee that each quantity inside the min operator is strictly greater than 1. Showing the (t', t)-leapfrog property for F_j would imply that This would imply that $w(F_j) = O(w(MST(G)))$ and since the edges of G' are partitioned into a constant number of subsets F_j, $w(G') = O(w(MST(G)))$. We consider F_0 separately from F_j, $j > 0$.

The F_0 case. Consider an arbitrary subset $S = \{\{u_1, v_1\}, \{u_2, v_2\}, \ldots, \{u_s, v_s\}\} \subseteq F_0$. To prove inequality (6) for S, it suffices to consider the case when $\{u_1, v_1\}$ is a longest edge in S. If for any $1 \leq k < s$, $|v_k u_{k+1}| > |u_1 v_1|$ or $|v_s, u_1| > |u_1 v_1|$, then the leapfrog property holds. So we assume that for all $1 \leq k < s$, $|v_k u_{k+1}| \leq |u_1 v_1|$ and $|v_s u_1| \leq |u_1 v_1|$. Let i be the phase in which $\{u_1, v_1\}$ gets processed, i.e., $\{u_1, v_1\} \in E_i$. Since $|u_1 v_1| \leq \alpha$, it is the case that for all $1 \leq k < s$, $|v_k u_{k+1}| \leq \alpha$ and $|v_s u_1| \leq \alpha$. Hence, $\{\{v_s, u_1\}\} \cup \{\{v_k, u_{k+1}\} \mid 1 \leq k < s\}$ is a subset of edges of G and each edge in this set gets processed in phase i or earlier.

Assume first that at least one edge in $\{\{v_s, u_1\}\} \cup \{\{v_k, u_{k+1}\} \mid 1 \leq k < s\}$ gets processed in phase i. Then the right hand side of inequality (6) is at least tW_{i-1}, since edges in E_i have weights in the interval $I_i = (W_{i-1}, rW_{i-1}]$. Also since $t'|u_1 v_1| \leq t' r W_{i-1}$, and since the inequality $t' r W_{i-1} < tW_{i-1}$ is guaranteed by the values of r and t' in (7), the leapfrog property holds for this case.

Assume now that all edges in $\{\{v_s, u_1\}\} \cup \{\{v_k, u_{k+1}\} \mid 1 \leq k < s\}$ have been processed in phase $i-1$ or earlier, meaning that t-spanner paths between their endpoints exist in G'_{i-1} at the time $\{u_1, v_1\}$ gets processed. For $1 \leq k < s$, let P_k be a shortest $v_k u_{k+1}$-path in G'_{i-1}, and let P_s be a shortest $v_s u_1$-path in G'_{i-1}. Let P be the following $u_1 v_1$-path in G'_i: $P = P_1 \oplus \{u_2, v_2\} \oplus P_2 \oplus \{u_3, v_3\} \oplus \cdots \oplus P_s$. Here, we use $\oplus$ to denote concatenation. We distinguish two cases, depending on the size of the subset $S \cap E_i$.

(i) $|S \cap E_i| > 2$. Then, $w(P) \geq 2W_{i-1}$. We also have that $|u_1 v_1| \leq rW_{i-1}$, since $\{u_1, v_1\} \in E_i$. It follows that $w(P) > t'|u_1, v_1|$ for any $t' < \frac{2}{r}$. Furthermore, $w(P)$ is no greater than the right hand side of the (t', t)-leapfrog inequality (6), so lemma holds for this case as well.

(ii) $|S \cap E_i| = 2$. In addition to $\{u_1, v_1\}$, assume that $\{u_k, v_k\} \in E_i$ for some k, $1 < k \leq s$. It the (t', t)-leapfrog inequality (6) holds, we are done and so let us assume the opposite of that:

$$t' \cdot |u_1 v_1| \geq \sum_{i=2}^{s} |u_i v_i| + t \cdot \Big(\sum_{i=1}^{s-1} |v_i u_{i+1}| + |v_s u_1| \Big). \tag{8}$$

Since all edges $\{u_j, v_j\}$, $1 \leq j \leq s$, except for $\{u_1, v_1\}$ and $\{u_k, v_k\}$ are in G'_{i-1}, and since G'_{i-1} contains t-spanner $v_j u_{j+1}$-paths for all j, $1 \leq j < s$, and a t-spannner $v_s u_1$-path, the above inequality yields

$$t' \cdot |u_1 v_1| \geq \mathtt{sp}_{G'_{i-1}}(v_1, u_k) + |u_k v_k| + \mathtt{sp}_{G'_{i-1}}(v_k, u_1).$$

Multiplying both sides by $(1 + 6\delta)/(1 - 2\delta)$ and using $t' < t_\delta$ (which is implied by our choice of

t') and Lemma 7, we get

$$t \cdot |u_1 v_1| \geq \mathtt{sp}_{H_{i-1}}(v_1, u_k) + |u_k v_k| + \mathtt{sp}_{H_{i-1}}(v_k, u_1). \tag{9}$$

We now observe that the inequality

$$t_\delta \cdot |u_k v_k| < \sum_{i=1}^{k-1} |u_i v_i| + \sum_{i=k+1}^{s} |u_i v_i| + t \cdot \Big(\sum_{i=1}^{s-1} |v_i u_{i+1}| + |v_s u_1| \Big) \tag{10}$$

implies the (t', t)-leapfrog property. To see this use the fact that both $\{u_1, v_1\}$ and $\{u_k, v_k\}$ belong to E_i and therefore $|u_1 v_1| < r \cdot |u_k v_k|$, which substituted in (10) yields:

$$t_\delta \cdot |u_k v_k| - (r-1) \cdot |u_k v_k| < \sum_{i=2}^{s} |u_i v_i| + t \cdot \Big(\sum_{i=1}^{s-1} |v_i u_{i+1}| + |v_s u_1| \Big).$$

We get the lower bound $t' \cdot |u_1, v_1|$ on the lefthand side of the above inequality by using $|u_k v_k| > |u_1 v_1|/r$ again and our choice of $t' < (t_\delta + 1)/r - 1$. This yields the (t', t)-leapfrog property. So we assume that inequality (10) does not hold, that is,

$$t_\delta \cdot |u_k v_k| \geq \sum_{i=1}^{k-1} |u_i v_i| + \sum_{i=k+1}^{s} |u_i v_i| + t \cdot \Big(\sum_{i=1}^{s-1} |v_i u_{i+1}| + |v_s u_1| \Big).$$

Since all edges $\{u_j, v_j\}$, $1 \leq j \leq s$, except for $\{u_1, v_1\}$ and $\{u_k, v_k\}$ are in G'_{i-1}, and since G'_{i-1} contains t-spanner $v_j u_{j+1}$-paths for all j, $1 \leq j < s$, and a t-spannner $v_s u_1$-path, the above inequality yields

$$t_\delta \cdot |u_k v_k| \geq \mathtt{sp}_{G'_{i-1}}(v_1, u_k) + |u_1 v_1| + \mathtt{sp}_{G'_{i-1}}(v_k, u_1).$$

Multiplying both sides by $(1 + 6\delta)/(1 - 2\delta)$ and using Lemma 7, we get

$$t \cdot |u_k v_k| \geq \mathtt{sp}_{H_{i-1}}(v_1, u_k) + |u_1 v_1| + \mathtt{sp}_{H_{i-1}}(v_k, u_1). \tag{11}$$

Inequalities (9) and (11) imply that edges $\{u_1, v_1\}$ and $\{u_2, v_2\}$ are mutually redundant and therefore cannot both exist in the spanner — a contradiction.

Now consider a subset $S = \{\{u_1, v_1\}, \{u_2, v_2\}, \ldots, \{u_s, v_s\}\} \subseteq F_i$, with $\{u_1, v_1\}$ a longest edge in S. Note that $|u_k v_k| > |u_1 v_1|/\beta$ for all $k = 2, 3, \ldots, s$. If $|S| \geq 3$, then the right hand side of the (t', t)-leapfrog inequality (6) is at least $2 \cdot |u_1 v_1|/\beta$ and therefore the (t', t)-leapfrog inequality goes through for any $1 < t' < 2/\beta$. Otherwise, if $|S| = 2$, then we need to show that $t' \cdot |u_1 v_1| < |u_2 v_2| + t \cdot (|u_1 v_2| + |u_2 v_1|)$.

If each of $|u_1v_2|$ and $|u_2v_1|$ is at most α, then using the same argument as in the F_0-case with $|S \cap E_i| = 2$, we can show that $\{u_1, v_1\}$ and $\{u_2, v_2\}$ are mutually redundant and will not both exist in the spanner. Otherwise, if one of $|u_1v_2|$ or $|u_2v_1|$ is greater than α, then the right hand side of the (t', t)-leapfrog inequality (6) is greater than $|u_1v_1|/\beta + t\alpha$. To ensure that the inequality goes through, we require that $t' \cdot |u_1v_1| \leq \frac{|u_1v_1|}{\beta} + t \cdot \alpha$. Since $|u_1v_1|$ is at most 1, the above inequality is satisfied for any $1 < t' \leq t\alpha + \frac{1}{\beta}$, which holds true cf. (7). $\square$