Topology Control and Geographic Routing in Realistic Wireless Networks

Kevin M. Lillis[1,2], Sriram V. Pemmaraju[1], and Imran A. Pirwani[1]

[1] Department of Computer Science,
University of Iowa, Iowa City, IA 52242-1419, U.S.A.
[lillis,sriram,pirwani]@cs.uiowa.edu
[2] Computer and Information Science,
St. Ambrose University, Davenport, IA 52803, U.S.A.
LillisKevinM@sau.edu

Abstract. We present a distributed topology control protocol that runs on a d-QUDG for $d \geq 1/\sqrt{2}$, and computes a sparse, constant-spanner, both in Euclidean distance and in hop distance. QUDGs (short for Quasi Unit Disk Graphs) generalize Unit Disk Graphs and permit more realistic modeling of wireless networks, allowing for imperfect and non-uniform transmission ranges as well as uncertain node location information. Our protocol is local and runs in $O(1)$ rounds. The output topology permits memoryless (geographic) routing with guaranteed delivery. In fact, when our topology control protocol is used as preprocessing step for the geographic routing protocol GOAFR$^+$, we get the routing time guarantee of $O(\ell^2)$ for any source-destination pair that are ℓ units away from each other in the input d-QUDG. The key idea is simple: to obtain planarity, we replace each edge intersection with a *virtual node* and have a real node serve as a proxy for the virtual node. This idea is supported by other parts of our protocol that (i) use clustering to keep the density of edge crossings bounded and (ii) guarantee that an edge between a virtual node and a neighbor is realized by a constant-hop path in the real network. The virtual node idea is simple enough to be useful in many contexts. For example, it can be combined with a scheme recently suggested by Funke and Milosavljević (INFOCOM 2007) to guarantee delivery under uncertain node locations. Similarly, the virtual nodes idea can also be used as a cheap alternative to edge-crossing removal schemes suggested by Kim et al. (DIALM-POMC 2005, SENSYS 2006).

1 Introduction

A wireless ad-hoc network typically consists of battery-powered individual nodes that are able to communicate with nodes in transmission range via radio broadcast and perform local computations. Because there is no centralized control and because each node has only a small amount of memory, *memoryless routing* protocols are highly desirable for wireless ad-hoc networks. In memoryless routing protocols, each node decides whom a message should be forwarded to based solely on the source and destination of the message and on information gathered from nearby nodes. As a result, the per node memory used by a memoryless routing protocol depends on the local density of nodes rather than on the total number of nodes. *Geographic routing protocols* are memoryless routing protocols that use geographic information such as coordinates of source and destination, and coordinates of nodes that are a small number of hops away. Geographic routing protocols take advantage of the fact that nodes in a wireless sensor network typically reside in low dimensional Euclidean space and may know their coordinates, at least approximately, in some agreed-upon global coordinate system. For the rest of the paper, we assume that nodes of the given wireless network reside in $\mathbb{R}^2$, the Euclidean plane.

Well-known methods in geographic routing are *greedy routing*, *face routing*, and *dual-mode routing*. The last method combines greedy and face routing. In greedy routing each node forwards a packet by choosing a neighbor that is closest to the destination. Greedy routing guarantees message delivery only in special circumstances, for example, when the communication graph of the wireless network happens to be a Delaunay triangulation [4]. In general, this scheme cannot guarantee delivery because a routed message may be trapped in a cycle [4]. In face routing it is assumed that the network is embedded in the plane with no edge crossings and using the *right-hand rule*, messages are forwarded along the boundaries of adjacent faces that intersect the line segment between the source node and the destination node. There exists a trade-off between greedy and face routing. Greedy routing is fast but cannot guarantee delivery, whereas face routing guarantees delivery, but is less efficient than the greedy method [18]. These two techniques have been combined in an effort to garner the benefits offered by each. Such dual-mode routing schemes [14, 16] oscillate between face routing and greedy routing. A message is greedily forwarded until a *local minimum* is reached; this is a node that is closer to the destination than any of its neighbors. Then face routing is employed to forward the message until greedy routing can resume. Well known dual-mode routing protocols include GPSR [14] and GOAFR$^+$ [16].

A starting point of much of the algorithmic research on wireless sensor networks is a graph-theoretic model of the pairwise communication between nodes in the network. The *unit disk graph* (UDG) may be the simplest of these models. Though popular because of its simplicity, the UDG model makes unrealistic assumptions such as radio transmission ranges being perfect disks, nodes having uniform transmission ranges, etc. The *Quasi Unit Disk Graph* model [2, 17] alleviates some of the shortcomings of the UDG model. A Quasi Unit Disk Graph is parameterized by d, $0 \leq d \leq 1$ and is usually denoted d-QUDG. This graph consists of vertices in $\mathbb{R}^2$ and an edge set E satisfying the rules: (i) $\{u, v\} \in E$ if $\|u - v\|_2 \leq d$ and (ii) $\{u, v\} \notin E$ if $\|u - v\|_2 > 1$. Note that edges between pairs of vertices u, v with $d < \|u - v\|_2 \leq 1$ are left unspecified and are assumed to be provided by an adversary. By adjusting the parameter d, a d-QUDG can model physical obstructions, imprecise location information, and irregularity in transmission ranges. In this paper, we use the d-QUDG model with $d \geq 1/\sqrt{2}$.

It is easy to see that d-QUDGs and UDGs are locally dense graphs and are therefore far from being planar. On the other hand, all known guaranteed-delivery geographic routing schemes seem to require a planar embedding of the communication network. This has motivated a large body of literature on *topology control protocols*. Informally speaking, topology control usually refers to the process of dropping edges in the communication network so as to construct a sparse spanning subgraph that has desirable properties such as planarity. Using a sparse spanning subgraph for communication also has other benefits such as allowing nodes to transmit at lower power and reducing interference between radio signals. The disadvantage of topology control is that nodes u and v that might have been close to each other in the original communication network may end up being far away from each other following topology control. One precise statement of the topology control problem is this: given a network $G = (V, E)$ (modeled as a d-QUDG or as a UDG) embedded in $\mathbb{R}^2$, find a spanning subgraph $H = (V, E_H)$ such that H has the following properties:

Planarity: No two edges in E_H cross in the embedding of G.
Sparsity: The maximum degree of H is bounded, i.e., $\Delta(H) = O(1)$.
Spanner property: H is a t-spanner of G for $t = O(1)$. Recall that H is a t-spanner of G if for all $u, v \in V$, $d_H(u, v) \leq t \cdot d_G(u, v)$. Here, $d_H(\cdot, \cdot)$ and $d_G(\cdot, \cdot)$ refer to distances in H and G, respectively. These distances may be Euclidean distances or hop distances and when the difference is important we will call H a *Euclidean* t-spanner or a *hop* t-spanner respectively.

Given the many goals of topology control, the above stated problem is just one of many possible variants of the topology control problem. For example, in some topology control research, minimizing the total Euclidean weight of the spanner is a goal [19, 7]; in other work, an important goal is to minimize an explicitly defined measure of signal interference [20].

Topology control protocols have been quite successful when the input graph is a UDG. For example, Wang and Li [22] present a distributed local protocol that takes a UDG as input and, in $O(1)$ communication rounds, computes a bounded degree, planar, Euclidean t-spanner, for constant t. This paper has been preceded by a number of papers on topology control that use versions of geometric proximity structures such as Delaunay triangulations, Gabriel graphs, Relative neighborhood graphs, Yao graphs, etc. The result that is critical to some of this work is a classical result from computational geometry due to Dobkin et al. [8] showing that "Delaunay graphs are almost as good as complete graphs" in the sense that a Delaunay triangulation of any planar point set is a Euclidean t-spanner of the complete graph on that point set. Topology control protocols for d-QUDG have been less common and less successful due to the somewhat fundamental reason that for d-QUDGs it is not always possible to obtain a spanning subgraph that is both planar and connected. See Figure 1(a) for a simple example. The graph shown in this figure represents a family of d-QUDGs for $d < 1$. This figure emphasizes the fact that even though UDGs (which are d-QUDGs with $d = 1$) always have a planar spanner, d-QUDGs may not, even for values of d arbitrarily close to 1. Thus, the standard approach to topology control in the plane - delete edges until a sparse, planar spanner is obtained, does not work. Barriére et al. [2] add *virtual edges* to the given d-QUDG to first obtain a supergraph and then run a standard topology control algorithm on the supergraph. The resulting output is a mix of real edges and virtual edges, that are realized by real paths. We propose adding *virtual nodes*, an approach that seems to be a simpler, more flexible alternative and one for which provable performance guarantees are relatively easy to obtain[3]. Next we describe our results in some detail.

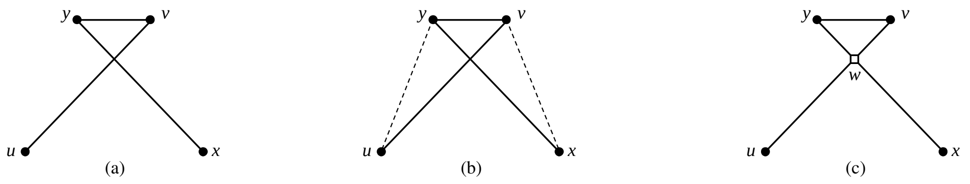

Fig. 1. (a) A d-QUDG G for which there is no connected, planar spanning subgraph. Here, $0 < d < 1$, $\|u - v\|_2 = \|x - y\|_2 = \|u - x\|_2 = 1$ and $\|y - v\|_2 = 1 - d$. By the triangle inequality $\|u - y\|_2$ (and similarly $\|x - v\|_2$) $> d$. In order to make G planar, edge $\{u, v\}$ or $\{x, y\}$ must be removed, thus disconnecting G. (b) The algorithm of Barriére et al. [2] first adds virtual edges $\{u, y\}$ and $\{x, v\}$, the Gabriel Graph construction then drops edges $\{u, v\}$ and $\{x, y\}$. (c) Our algorithm adds virtual node w at the intersection of edges $\{u, v\}$ and $\{x, y\}$. The virtual node w is *controlled* by a real node in $\{u, v, x, y\}$.

Our results. We present a distributed topology control protocol that runs on a d-QUDG for $d \geq 1/\sqrt{2}$, and computes a sparse, hop t-spanner, for constant t. As in other papers [2, 11, 17], the constraint $d \geq 1/\sqrt{2}$ is quite fundamental; otherwise two edges $e_1 = \{u_1, v_1\}$ and $e_2 = \{u_2, v_2\}$ may cross even though the hop-distance between an endpoint of e_1 and an endpoint of e_2 may be arbitrarily large. Our protocol is local and runs in $O(1)$ rounds. The output topology permits memoryless (geographic) routing with guaranteed delivery. In fact, when our topology control protocol is used as preprocessing step for the geographic routing protocol GOAFR$^+$ [16], we get a guarantee of $O(\ell^2)$ hops traveled by any message between a source-destination

[3] An anonymous referee has pointed out to us that this idea has been briefly mentioned by Kuhn et al [17].

pair that are ℓ hops away from each other in the input d-QUDG. The key idea is simple: to obtain planarity, we replace each edge crossing by a *virtual node* and have a real node serve as a proxy for the virtual node (see Figure 1(c) for an example). This idea is supported by other parts of our protocol that (i) use clustering to keep the density of edge crossings bounded and (ii) guarantee that an edge between a virtual node and a neighbor is realized by a constant-hop path in the real network. A more complicated version of our protocol yields, in $O(1)$ rounds, a sparse *Euclidean t*-spanner, for constant t. When used in combination with GOAFR$^+$ [16], we get a routing guarantee of $O(\ell^2)$ Euclidean distance traveled by any message between a source-destination pair that are a Euclidean distance ℓ apart in the input d-QUDG. In a recent paper, Funke and Milosavljević [11] present a scheme, called *Macroscopic Geographic Greedy Routing* (MGGR), that they claim guarantees geographic routing under uncertain node locations. We show that MGGR can yield graphs with many connected components and therefore no message delivery guarantee can be provided; the virtual nodes idea provides a simple fix to this problem. However, a key idea from MGGR when combined with our scheme, adds load balancing properties to our routing protocol. Since most known guaranteed-delivery geographic routing schemes rely on planarity, some recent papers have focused on the problem of removing edge crossings, for example the CLDP protocol [15] and the LCR protocol [13]. The virtual nodes idea provides a simple, cheap alternative to these schemes. To simplify our presentation, we assume the $\mathcal{LOCAL}$ model of distributed computation [21] which assumes that nodes have unique IDs, nodes run synchronously, and there is no bound on message sizes. However, we do not misuse the unboundedness of message sizes; all messages exchanged in our protocols contain a constant number of node IDs, a constant number of node locations, and a constant amount of control bits. We further assume that the amount of memory available to each node is a constant times the amount of memory required to store a node's ID and its Euclidean coordinates.

1.1 Related Work

Most papers on topology control and geographic routing assume the UDG model for wireless networks. Here we review three papers that have considered topology control and memoryless routing on d-QUDGs and are therefore most relevant to our work.

Barriére et al. [2] solve the same problem as we do. The wireless network is modeled as a d-QUDG G with $d \geq 1/\sqrt{2}$ and the authors note that computing a Gabriel graph of G (which is a standard technique for topology control on UDGs, first introduced in [5]) may produce a disconnected spanning subgraph. To overcome this problem *virtual edges* are added to G and a Gabriel graph of the resulting supergraph H is computed. The output is a mixture of real and virtual edges and is connected and planar (see Figure 1(b)). The shortcomings of this approach are (i) virtual edges may be realized by arbitrarily long real paths, (ii) the Gabriel graph is not a constant-spanner and in fact there are families of planar point sets for which the Gabriel graph is an $\Omega(\sqrt{n})$ spanner [3, 9], and (iii) the protocol is not local in the sense that information may have to travel between nodes that are more than constant hops from each other.

Kuhn et al. [17] model the wireless network as an arbitrary d-QUDG G, with $0 \leq d \leq 1$. Using a standard technique (as described by Alzoubi et al. [1]), a *connected dominating set (CDS)* is first extracted from G. Clustering is then used to reduce the number of edges of the CDS, resulting in a subgraph H of G. The authors show that using the *Echo Flooding Algorithm* [6, 21] on H, yields asymptotically optimal routing on G. The authors then go on to consider d-QUDGs with $d \geq 1/\sqrt{2}$. For such d-QUDGs, it is shown that if the algorithm of Barriére et al. [2] is executed on H, rather than on the original graph G, then the resulting topology allows for geographic routing with good worst case guarantees. This result (at least as described by Kuhn et al. [17]) is specific to the hop metric and does not seem to immediately extend to the Euclidean metric. The paper also briefly mentions the idea of using virtual nodes for topology control on d-QUDGs.

Funke and Milosavljević [11] also attempt to solve the problem of topology control and memoryless routing on a d-QUDG input graph G, but do so via an interesting idea called *macroscopic routing*. Their starting point is the selection of *landmark nodes*. For an integer parameter $k > 0$, the landmarks form a *k-independent set* of $V(G)$. Each node in G is then "associated" with the landmark that is closest to it, in hops, effectively partitioning $V(G)$ into *Voronoi tiles*. From this combinatorial Voronoi tiling, a *Combinatorial Delaunay Map* (CDM) is constructed. The vertex set of the CDM is the set of landmarks, $\{L_i\}$. Edge $\{L_a, L_b\}$ is then added if the following local rule is satisfied: (i) there is a path in G from L_a to L_b consisting of a sequence of nodes associated with L_a followed by a sequence of nodes associated with L_b, and (ii) the one-hop neighborhood of this path contains only nodes associated with L_a and L_b. As can be seen in Figure 2, even though this local rule yields a planar graph there are instances of G for which the CDM has multiple connected components.

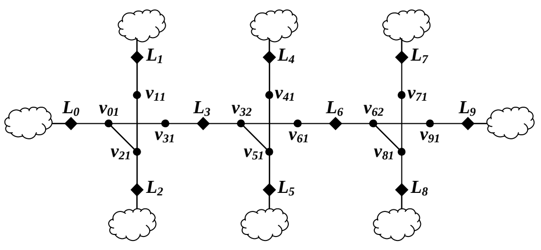

Fig. 2. Illustration of how the CDM constructed by Funke and Milosavljević [11] may have connected components. Consider the d-QUDG shown above; let parameter $k = 1$ and let the set of landmarks contain $L_i, 0 \leq i \leq 9$. The Voronoi tile of each landmark L_i contains nodes v_{ij}. None of the edges $\{L_i, L_j\}$, $0 \leq i \neq j \leq 9$ are added to the CDM because part (ii) of the local rule is violated and we get a CDM with at least 8 connected components. This construction can be easily extended to produce a CDM with arbitrarily many connected components. Similar examples can be constructed on UDGs and with any value of $k \geq 0$.

Once the CDM is constructed, a routing algorithm such as $GPSR$ [14] can be used to determine in which direction to send a message. The Funke-Milosavljević routing protocol only uses the CDM as a *macroscopic* guide for routing; a node u uses locations of nearby landmarks to determine which neighboring Voronoi tile to forward the message to and then the *microscopic* routing takes over and the message is forwarded to that neighboring tile using the gradient descent method similar to that used in the GLIDER protocol [10].

2 The Concept of Virtual Nodes

Rather than attempt to eliminate edge crossings to obtain planarity, we simply treat each edge crossing as a node — a *virtual node*. For the rest of the paper, we assume that $G = (V, E)$ is the input d-QUDG with $d \geq 1/\sqrt{2}$. Furthermore, we assume that G comes with an embedding in $\mathbb{R}^2$; each vertex $v \in V$ is at a point $p_v \in \mathbb{R}^2$ and each edge $\{u, v\}$ is a straight line segment connecting points p_u and p_v. We will think of vertices and edges as combinatorial objects as well as geometric objects depending on the context. Now consider a pair of edges $\{u, v\}$ and $\{x, y\}$ that cross. We deal with this edge crossing by modifying G as follows:

1. Add a virtual node w to G corresponding to the edge crossing. Associated with node w is a point $p_w \in \mathbb{R}^2$, namely the point of intersection of edges $\{u, v\}$ and $\{x, y\}$. Also associated with w is a *controller* node, denoted $w.controller$, belonging to the set $\{u, v, x, y\}$, which serves as the real proxy for w.
2. Delete edges $\{u, v\}$ and $\{x, y\}$ from G and add the four new edges $\{w, u\}$, $\{w, v\}$, $\{w, x\}$, and $\{w, y\}$. This transformation can be seen as going from the non-planar embedding of the graph in Figure 1(a) to the planar embedding in Figure 1(c).

We can eliminate all edge crossings by repeatedly applying the above two steps. The resulting *plane graph* (i.e., a planar graph along with a planar embedding) is denoted $VN(G)$. It is this plane graph on which we attempt to perform memoryless routing. First, we describe 3 properties that we seek for $VN(G)$. If $VN(G)$ were to satisfy all 3 properties, then we could perform fast memoryless routing on it.

Property 1: For every pair of adjacent nodes u and v in $VN(G)$, there is a constant-length path in G from $u.controller$ to $v.controller$. A routing step on $VN(G)$ that forwards a message from node u to its neighbor v is realized on the actual network G by a message going from $u.controller$ to $v.controller$. We want to show that the hop-distance in G between $u.controller$ and $v.controller$ is bounded by a constant. Otherwise, even though a pair of nodes u and v may be close together in $VN(G)$, there may be no cheap way of routing between u and v.

Property 2: Every real node in $VN(G)$ is the controller for a constant number of virtual nodes. For each node v in $VN(G)$ a routing table containing information on the neighbors of v in $VN(G)$ must be maintained at $v.controller$. This means the a real node u must store its own routing table as well as the routing table of each virtual node for which it is the controller. Hence we require that each real node in $VN(G)$ be the controller for only a constant number of virtual nodes.

Property 3: The size of the routing table for each node in $VN(G)$ is constant. For our routing to be memoryless, we must guarantee not only that each real node maintain a small number of routing tables, but also that the size of each routing table is small. One way to ensure this is to bound the degrees of nodes in $VN(G)$; a constant bound on the degrees would be ideal. This would result in each node's routing table consisting of the IDs and Euclidean coordinates of a constant number of nodes.

Now we prove the first of the three desired properties mentioned above for d-QUDGs with $d \geq 1/\sqrt{2}$.

Lemma 1. *Let $G = (V, E)$ be a d-QUDG with $d \geq 1/\sqrt{2}$. Then $VN(G)$ is a plane graph such that for adjacent nodes u and v in $VN(G)$ the hop distance between $u.controller$ and $v.controller$ in G is at most 5.*

Proof. We start with an observation that due to Kuhn et al. (see Lemma 8.1 in [17]): for intersecting edges $\{u, v\}, \{x, y\} \in E(G)$ and any $s, t \in \{u, v, x, y\}$ there is an st-path, that has hop-length at most 3. The lemma follows easily because (i) if both u and v are real nodes, then $u = u.controller$ and $v = v.controller$ are one hop apart, (ii) if one of u and v is a real node then there is a path from $u.controller$ to $v.controller$ whose length is at most 3 hops, and (iii) if both u and v are virtual nodes, then $u.controller$ and $v.controller$ are at most 5 hops apart.

It is easily checked that the upper bound of Lemma 1 is tight.

Corollary 1. *Let $G = (V, E)$ be a d-QUDG with $d \geq 1/\sqrt{2}$ and let H be a hop t-spanner of G. Then $VN(H)$ is a plane graph such that for any pair of adjacent nodes u and v in $VN(H)$, the hop distance between u.controller to v.controller in G is at most $5 \cdot t$.*

Property (2) above can be satisfied by ensuring that each edge in crossed at most a constant number of times, whereas Property (3) can be satisfied by guaranteeing a constant upper bound on the maximum vertex degree. This motivates the question of whether every d-QUDG has an $O(1)$-spanner with (i) constant degree and (ii) constant number of edges crossing every edge. An $O(1)$-*hop* spanner with constant degree is impossible, even for cliques since with a constant degree bound it takes $\Omega(\log n)$ hops to reach every vertex from any given vertex. Unfortunately, as shown below, the constant edge crossing requirement is incompatible with having an $O(1)$-*Euclidean* spanner. This example should be viewed as a generalization of the example in Figure 1(a). There, 0 crossings were allowed and we could not guarantee connectedness; here a constant number of crossings are allowed, but this still does not allow for the Euclidean t-spanner property, for any constant t.

Lemma 2. *Let G be a d-QUDG as shown in Figure 3 with $d < 1$, integer $\alpha > 0$, and $t \geq 1$. If H is an arbitrary spanning subgraph of G such that every edge in H is intersected by at most α other edges, then H is not a Euclidean t-spanner of G.*

Proof. Let $\ell = \|x_i - x_{i+1}\|_2 = \|y_i - y_{i+1}\|_2 = \frac{1-d}{\alpha}$, for $1 \leq i \leq \alpha$. Since G is connected and $N_G(u) = \{v\}$, edge $\{u, v\} \in E(H)$. Since $\{u, v\}$ is crossed by at most α other edges in H, there exists a vertex x_j that does not have an incident edge in H that intersects $\{u, v\}$. Specifically, edge $\{x_j, y_j\} \notin E(H)$. Hence $d_H(x_j, y_j) > \ell$. Combined with the fact that $d_G(x_j, y_j) = \varepsilon$, we get the ratio $\frac{d_H(x_j,y_j)}{d_G(x_j,y_j)} > \frac{\ell}{\varepsilon} = \frac{(1-d)/\alpha}{\varepsilon} > \frac{(1-d)/\alpha}{(1-d)/(t\cdot\alpha)} = t$. Therefore H is not a Euclidean t-spanner of G. □

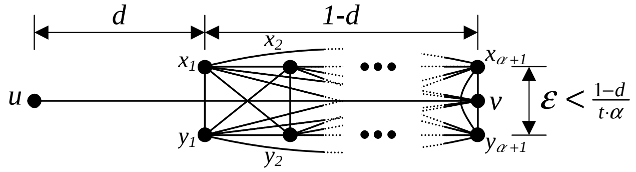

Fig. 3. This figure illustrates the proof of Lemma 2. In this graph G, $N_G(u) = \{v\}$ and the set of vertices $V(G) \setminus \{u\}$ form a clique. Note that each edge intersecting $\{u, v\}$ has the form $\{x_i, y_j\}$ and that the length of each edge $\{x_i, y_i\} = \varepsilon$.

3 A Hop-Spanner with Virtual Nodes

Going from G to $VN(G)$ gives us planarity and Lemma 1 tells us that adjacent nodes in $VN(G)$ are nearby in G (Property 1 holds for $VN(G)$). However, $VN(G)$ does not support *memoryless* routing because it does not guarantee Properties 2 or 3.

To get around this problem, we start by clustering G, sparsifying it and only then introducing virtual nodes (see Figure 4). Network clustering is a standard technique in routing and our approach is similar to that of Gao et al. [12]. We start by constructing a backbone graph G_B whose vertex set is a small subset of $V(G)$. A routing graph G_R is constructed by adding virtual nodes to G_B and then attaching non-backbone nodes to G_B. Algorithm `BuildBackbone` describes the backbone construction and Algorithm `Route1` describes how routing is performed on G_R.

Algorithm `BuildBackbone`
Input: $G = d$-QUDG with $d \geq 1/\sqrt{2}$
Output: Backbone Graph G_B

1. Place an infinite grid of size $d/\sqrt{2} \times d/\sqrt{2}$ on the plane. Since nodes know their locations in a global coordinate system, each node knows the identity of the grid cell to which it belongs. The nodes in each grid cell form a *cluster* and the grid induces a partition of $V(G)$ into clusters. Let $\widetilde{G}$ be the *cluster graph* of G in which the vertex set is the set of clusters and there is an edge from vertex C to C' iff there is an edge in G with one end node in cluster C and the other in cluster C'. Note that since the diagonal of each grid cell has length d, the nodes within any given grid cell form a clique.
2. For each cluster C_i the nodes belonging to the cluster select a representative *cluster head* h_i. Let $\mathcal{L}$ be the set of all cluster heads.
3. Each cluster head h_i (the head of cluster C_i) selects exactly one edge $\{u, v\}$ in G corresponding to each neighbor C_j of C_i in $\widetilde{G}$ such that u is in cluster C_i and v is in cluster C_j. The number of edges selected by h_i is therefore equal to the degree of C_i in $\widetilde{G}$. Let $\mathcal{S}$ be the set of end nodes of all such selected edges in G.
4. G_B is then defined as the subgraph of G induced by $\mathcal{L} \cup \mathcal{S}$.

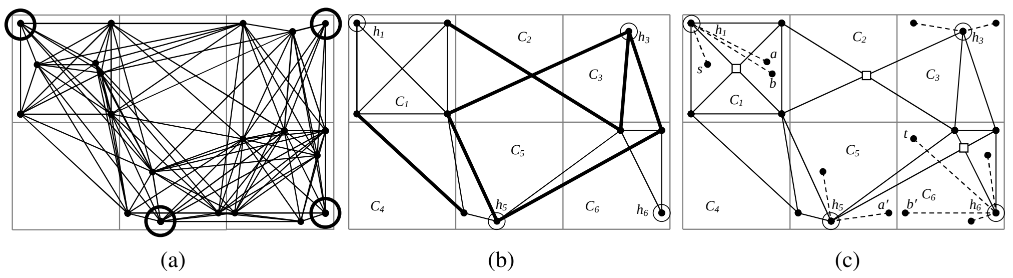

Fig. 4. (a) A d-QUDG G with $d = 1/\sqrt{2}$ along with a $1/2 \times 1/2$ grid and cluster heads for nonempty grid cells. (b) Backbone graph G_B. Edges to neighboring clusters selected by the cluster heads are bold. Other induced edges are non-bold. (c) Routing graph G_R, comprised of (i) $VN(G_B)$, i.e. G_B with virtual nodes (squares) at edge intersections and (ii) remaining nodes in G attached to cluster heads (dashed edges). To route a message M from s in cluster C_1 to t in cluster C_6, s forwards M to cluster head h_1, M is then routed from h_1 to h_6 using graph $VN(G_B)$ and a routing protocol such as GPSR, and h_6 broadcasts M to all of its neighbors, one of which is t.

Lemma 3. *G_B has the following properties: (i) $\Delta(G_B)$ is $O(1)$, (ii) Each edge in G_B is crossed by $O(1)$ other edges, (iii) $V(G_B)$ is a connected dominating set of G.*

Proof. Let us begin with a definition and two facts. **Definition:** For each grid cell C let the *surrounding* cells of C be the set of cells $\{C' \mid \exists x, y \in \mathbb{R}^2 : x \in C', y \in C, \|x - y\|_2 \leq 1\}$. **Fact 1:** For any grid cell C the number of surrounding cells of C is bounded above by a constant. **Fact 2:** A constant number of nodes from each grid cell C_i are included in G_B. These include cluster head h_i along with at most two nodes associated with each surrounding cell of C_i.

(i) Consider a node u in G_B that lies in grid cell C. Since G_B is an induced subgraph of G, the degree of u in G_B is bounded by the number of nodes in $V(G_B)$ that are at most distance 1 from u. Since there are $O(1)$ cells surrounding C (Fact 1) and since each such cell contributes $O(1)$ nodes to $V(G_B)$ (Fact 2), there are $O(1)$ nodes whose distance from u is at most 1. Hence degree of u in G_B is $O(1)$.

(ii) Consider an edge $\{u, v\}$ in G_B. Since the grid size is $d/\sqrt{2} \times d/\sqrt{2}$ and the maximum edge length is 1, edge $\{u, v\}$ passes through at most five grid cells; call them C_1 through C_5. Hence for any edge $\{x, y\}$ that intersects $\{u, v\}$, x and y both must lie in the surrounding cells of C_1 through C_5. From Facts 1 and 2 there are a constant number of such surrounding cells, each with a constant number of nodes in G_B. Therefore there are $O(1)$ edges that intersect $\{u, v\}$.

(iii) Since $V(G)$ is partitioned into clusters, each of which forms a clique, and since $V(G_B)$ includes at least one node from each cluster, $V(G_B)$ is a dominating set of G. We now show that $V(G_B)$ is connected. Consider nodes s and t in G_B. Since G is connected there is a path in G from s to t. Let this path be $s = x_0, x_1, \dots, x_p = t$. This st-path corresponds to a sequence of adjacent clusters $C_0, C_1, \dots, C_q$, $q \leq p$, with s being in cluster C_0 and t being in cluster C_q. To show s and t are connected in G_B we show the following pairs of nodes are connected in G_B: (i) s and h_0, (ii) t and h_q, (iii) h_i and $h_{i+1}, 0 \leq i \leq q-1$. Since s and h_0 are in the same cluster there is an edge between them in G which means there is also an edge between them in G_B. Likewise, there is an edge between t and h_q in G_B. Since C_i and C_{i+1} are adjacent in $\widetilde{G}$ there is an edge $\{u, v\}$ in G_B where u is in cluster C_i and v is in cluster C_{i+1}. Also since u and h_i are in the same cluster there is an edge between them in G_B. Likewise there is an edge between v and h_{i+1}. Therefore h_i and h_{i+1} are connected in G_B. $\square$

Once G_B is constructed we add virtual nodes at each edge intersection and obtain $VN(G_B)$. The routing graph G_R is then defined as $VN(G_B)$ along with all nodes $V(G) \setminus V(G_B)$ with edges to their respective cluster heads. The following algorithm can then be used to route messages using G_R (see Figure 4(c)).

```
Algorithm Route1
Input: $s, t \in V(G)$, message $M$, geographic routing protocol $\mathcal{A}$

  1. Node $s$ forwards the message to its cluster head $h$.
  2. The message is routed from cluster head $h$ to cluster head $h'$ of $t$ using geographic protocol $\mathcal{A}$.
  3. Cluster head $h'$ broadcasts the message to its neighbors, one of which is $t$.
```

`Route1` is memoryless because each real backbone node maintains constant-sized routing tables for itself and at most constant number of virtual nodes; each non-backbone node just maintains the ID and position of its cluster head. Now we claim (without proof to conserve space) a worst case upper bound on the number of hops it takes for a message to be delivered by Algorithm `Route1`. The lemma not only claims a relatively short st-path, but also that such a path uses only the backbone graph, with the possible exception of the two end nodes, s and t.

Lemma 4. *For any vertices s and t in G there is a path $L = (s = z_0, z_1, \dots, z_\ell = t)$ in G_R such that $\ell \leq \alpha \cdot d_G(s, t)$, for some constant α. Furthermore, $z_1, z_2, \dots, z_{\ell-1}$ are nodes in $VN(G_B)$.*

To obtain worst case guarantees on the number of hops it takes to route using Algorithm `Route1`, we use GOAFR$^+$ [16] as the geographic routing protocol $\mathcal{A}$ in Step 2 of Algorithm

`Route1`. Like GPSR, GOAFR$^+$ is a dual-mode routing protocol, but it uses a clever way of controlling the face routing mode by using a set of bounding circles with geometrically varying area. Kuhn et al. [16] prove the following theorem about the performance of GOAFR$^+$.

Theorem 1. (Kuhn et al. [16]) *Let p^* be an optimal path from s to t. On a bounded degree UDG, GOAFR$^+$ reaches t with cost $O(c^2(p^*))$.*

Here $c : (0, 1] \to \mathbb{R}^+$ is an arbitrary *cost function* that associates a cost $c(\|u - v\|_2)$ to every edge $\{u, v\}$ in the graph. The only restriction on c is that it is non-decreasing, i.e., $c(d') \geq c(d)$ if $d' > d$. For any path P in the graph, $c(P)$ is simply the sum of the costs of the edges in P. Note that $c(d) = 1$ for all $d \in (0, 1]$ models hop distances, whereas $c(d) = d$ for all $d \in (0, 1]$ models Euclidean distances. Theorem 1 is proved for bounded degree UDGs, however it easily extends to arbitrary subgraphs (not necessarily induced) of bounded degree UDGs. Since $VN(G_B)$ is clearly such a graph, we can use GOAFR$^+$ in Step 2 of Algorithm `Route1` on $VN(G_B)$, even though $VN(()\,G_B)$ may not be a UDG. As a result, we get the following theorem providing worst case guarantees on the performance of Algorithm `Route1`.

Theorem 2. *Let s and t be nodes in G and let ℓ be the hop-length of a shortest st-path in G. If GOAFR$^+$ is used as the geographic routing algorithm $\mathcal{A}$ in `Route1`, then the hop-distance traveled by a message routed from s to t using `Route1` is $O(\ell^2)$.*

4 A Euclidean Spanner with Virtual Nodes

The routing graph G_R constructed in Section 3 may not be a Euclidean t-spanner of G (see nodes a and b in cell C_1 of Figure 4(c) for an example). One possible solution is to compute a bounded degree, planar, Euclidean t-spanner of the clique induced by the vertices in each cell C rather than connecting all non-backbone nodes in C directly to C's cluster head. In order to account for nodes which may be arbitrarily close together yet lie in different grid cells (see nodes a' and b' of Figure 4(c)) we utilize three vertex partitions of G induced by three different grids obtained by "shifting" the grid of Section 3. This ensures every pair of vertices that are close to each other will be contained completely in a grid cell of at least one of the three grids. This construction is described in Algorithm `BuildRoutingGraph` and illustrated in Figure 5.

A key ingredient to this solution is the construction of a bounded degree, planar, Euclidean t-spanner of the cliques induced by the grid cells of the three grids. Wang and Li [22] have proposed a local algorithm for computing just such a spanner. For concreteness, we fix the Wang-Li algorithm and for any UDG H, let the *Wang-Li spanner* of H, denoted $WL(H)$, be the bounded degree, planar, Euclidean t-spanner of H computed by the algorithm of Wang and Li [22]. Let $G[C]$ be the subgraph of G induced by the vertices in cell C.

```
Algorithm BuildRoutingGraph
Input: $G = d$-QUDG with $d \geq 1/\sqrt{2}$
Output: Routing Graph $G_R$

1. Place a blue grid of $\frac{d}{\sqrt{2}} \times \frac{d}{\sqrt{2}}$ cells passing through $(0,0)$, a red grid of $\frac{d}{\sqrt{2}} \times \frac{d}{\sqrt{2}}$ cells passing
   through $(\frac{d}{3\sqrt{2}}, \frac{d}{3\sqrt{2}})$, and a green grid of $\frac{d}{\sqrt{2}} \times \frac{d}{\sqrt{2}}$ cells passing through $(\frac{2d}{3\sqrt{2}}, \frac{2d}{3\sqrt{2}})$.
2. For each edge $e$ in $G$ initialize $color(e)$ to the empty set.
3. For each non-empty grid cell $C$ in each grid, construct the Wang-Li spanner $WL(G[C])$. If
   $C$ belongs to a grid of color $x \in \{\text{blue}, \text{red}, \text{green}\}$ then add $x$ to $color(e)$ for each edge $e$ in
   $WL(G[C])$.
4. Construct $G_B$ using the blue grid and algorithm BuildBackbone from Section 3.
5. $G_R$ is the union of $VN(G_B)$ and each $WL(G[C])$ from step 3.
```

To route a message from s to t, node s first checks to see if it shares a cell with t in any of the three grids. If so, the message is routed from s to t using only edges of $WL(G[C])$. If not, then the backbone graph is used for routing. The details of this protocol are given below in Algorithm `Route2`.

```
Algorithm Route2
Input: $s, t \in G$, message $M$, geographic routing algorithm $\mathcal{A}$.

1. If $s$ and $t$ share a cell $C$ in a grid of color $x$ then route $M$ on edges colored $x$, using algorithm $\mathcal{A}$.
   STOP.
2. Otherwise, let $C$ be the blue cell containing $s$. Route $M$ from $s$ to the cluster head $s'$ of $C$ on
   edges colored blue using $\mathcal{A}$.
3. Let $t'$ be the cluster head in the blue cell containing $t$. Route from $s'$ to $t'$ using $\mathcal{A}$ on $VN(G_B)$
   constructed in Step 4 of BuildRoutingGraph.
4. Route from $t'$ to $t$ on edges colored blue using algorithm $\mathcal{A}$. STOP.
```

As in the case of `Route1`, it is easy to see that `Route2` is memoryless. The following lemma states that if s and t are nearby then one of the Wang-Li spanners will provide a short st-path.

Lemma 5. *For any pair of vertices u and v in G with $\|u-v\|_2 \leq \frac{d}{3\sqrt{2}}$, there is a cell of some color that contains both u and v.*

In the following, let $d'_H(s,t)$ denote the length of an st-path in H with smallest Euclidean length.

Lemma 6. *For any pair of vertices s and t in G, there exists a path $L = (s = z_0, z_1, z_2, \ldots, z_p = t)$ whose Euclidean length is at most $\beta \cdot d'_G(s,t)$ for some constant β. Furthermore, if $\|s-t\|_2 > \frac{d}{3\sqrt{2}}$ then there are vertices z_i and z_j in L, $i < j$, such that (i) z_i and z_j are cluster heads of the blue cells containing s and t respectively, (ii) the subpath $L_1 = (s = z_0, z_1, \ldots, z_i)$ lies in the blue cell that s belongs to, (ii) the subpath $L_2 = (z_i, z_{i+1}, \ldots, z_{j-1}, z_j)$ belongs entirely to $VN(G_B)$, and (iii) the subpath $L_3 = (z_j, z_{j+1}, \ldots, z_p = t)$ lies entirely in the blue cell containing t.*

Proof. Consider an st pair. If s and t share a grid cell of color x, then according to "Step 1" of `Route2`, there is a path of length at most $c_1 \cdot d'_G(s,t)$, for some constant c_1, using only edges of color x.

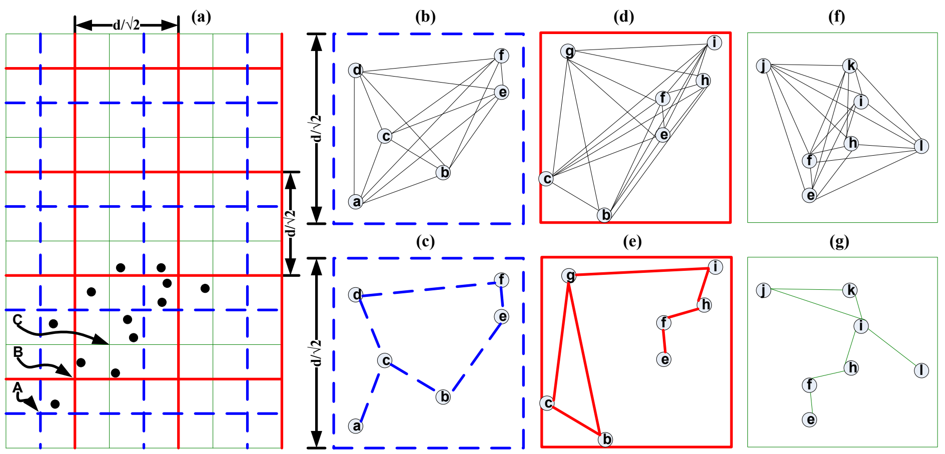

Fig. 5. (a) The three shifted grids with $\frac{d}{\sqrt{2}} \times \frac{d}{\sqrt{2}}$ dimensional cells. The grids are colored blue, red, and green and are shown with dashed, heavy, and light lines, respectively. (b) A clique induced by the blue cell C_b with lower left corner A. (c) A planar, constant-degree Euclidean t-spanner of $G[C_b]$. (d) A clique induced by the red cell C_r with lower left corner B. Note that vertices b, c, e, and f are common to C_b and C_r. (e) A planar, constant-degree Euclidean t-spanner of $G[C_r]$. (f) A clique induced by the green cell C_g with lower left corner C. Note that vertices e, f, h, and i are common to C_r and C_g. (g) A planar, constant-degree Euclidean t-spanner of $G[C_g]$. Note that vertices e and f are common to C_b, C_r, and C_g. Also note that edge $\{e, f\}$ happens to be common to the spanners in (c), (e), and (g) and hence has three colors in G_R.

So, let s and t not share any grid cell. By Lemma 5, $d'_G(s,t) > \frac{d}{3\sqrt{2}}$. Now, consider the blue grid and the partition induced by it. Let G_B be the backbone constructed in "Step 4" of `BuildRoutingGraph`. Let P be a shortest Euclidean length path in G having fewest vertices. Partition P into contiguous subsequences of vertices that belong to the same cell. Write these sequences as $B_0, B_1, \ldots, B_q$, each B_i belongs to C_i, and B_i and B_{i+1} belong to distinct cells. Now note that any vertex in B_i and any vertex in B_{i+2} are more than d units apart. Hence, $\frac{q \cdot d}{2} \leq d'_G(s,t)$. Let h_i denote the clusterhead of C_i. Consider $Q = (s \rightsquigarrow h_0, h_1, \ldots, h_q \rightsquigarrow t)$, where $s \rightsquigarrow h_0$ and $h_q \rightsquigarrow t$ are paths in the blue Wang-Li spanner between s and its clusterhead h_0, and t and its clusterhead h_q, respectively. Now, consider the backbone graph along with all the blue Wang-Li spanners; call it H. Note that H has no virtual nodes. Q can be realized as a path $R = (s \rightsquigarrow h_0, x_1, x_2 \ldots, x_{r-1}, h_r \rightsquigarrow t)$ in H with $r \leq 3q$. This is because G_B contains a path of at most 3 hops between h_i and h_{i+1}, $0 \leq i < q$. Furthermore, note that all the x_i's belong to G_B. G_R is obtained from H by the introduction of virtual nodes into G_B. The introduction of virtual nodes expands each edge in G_B by at most $(c+1)$ hops where c is the constant upper bound on the number of edges of G_B that can cross any edge in G_B. Thus the total Euclidean distance of the "expanded" path is $2c_1 + 3(c+1)q$, where c_1 is the upper bound on the Euclidean stretch guaranteed by the Wang-Li spanner. Using the upper bound of $q \leq 2d'_G(s,t)/d$, we get the result. □

We can combine the spanning property of the routing graph, proved in the above lemma, with worst case performance guarantees for GOAFR$^+$ to obtain the following result.

Theorem 3. *Let s and t be nodes in G and let ℓ be the length of a shortest Euclidean st-path in G. If GOAFR$^+$ is used as the geographic routing algorithm $\mathcal{A}$ in* `Route2`*, then the Euclidean distance traveled by a message routed from s to t using* `Route2` *is $O(\ell^2)$.*

Proof. If s and t are such that $\|s - t\|_2 \leq \frac{d}{3\sqrt{2}}$ then Lemma 5 tells us that s and t are in a cell C. In this case, Algorithm `Route2` uses $WL(G[C])$ to route the message from s to t. By construction, $WL(G[C])$ is guaranteed to be planar and contain a path of Euclidean length at most $c \cdot \ell$ for some constant c. Therefore, if GOAFR$^+$ is used to route the message from s to t in $WL(G[C])$, the message travels $O((c\ell)^2) = O(\ell^2)$ Euclidean distance.

If $\|s-t\|_2 > \frac{d}{3\sqrt{2}}$, then Lemma 6 tells us that there exists a path $L = (s = z_0, z_1, z_2, \dots, z_p = t)$ whose Euclidean length is at most $\beta \cdot \ell$ for some constant β. Now consider the subpaths L_1, L_2, and L_3 of L, and let ℓ_1, ℓ_2, and ℓ_3 respectively be the Euclidean lengths of these subpaths. Let C be the blue cell containing s. Since L_1 is a path from $s = z_0$ to z_i in $WL(G[C])$ and since Algorithm `Route2` routes from $s = z_0$ to z_i using GOAFR$^+$ on the graph $WL(G[C])$, the message travels a distance of $O(\ell_1^2)$ from s to z_i. Similarly, the message travels a distance of $O(\ell_2^2)$ from z_i to z_j and $O(\ell_3^2)$ from z_j to t. Thus the message travels a total distance of $O(\ell_1^2+\ell_2^2+\ell_3^2)$ in G_R. Using the fact that $\ell_1 + \ell_2 + \ell_3 =\leq \beta \cdot \ell$ and the fact that $\ell_1^2 + \ell_2^2 + \ell_3^2 \leq (\ell_1 + \ell_2 + \ell_3)^2$, we get that $O(\ell^2)$ is the total distance traveled by the message. □

5 Suggestions for Load Balancing

The topology control and routing schemes proposed in the previous two sections tend to overload the backbone nodes with the responsibility of forwarding most messages, with only a tiny fraction of the network nodes bearing most of the responsibility for forwarding messages. These schemes can be modified so the routing responsibility is more evenly distributed. Such modifications could include the standard technique of rotating the cluster head designation among all nodes in a cluster. Here we mention two other approaches. In the construction in the previous section, we used 3 shifted grids to ensure that nodes that are close enough share a grid cell in at least one of the three grids. This idea can also be used to construct 3 backbone graphs instead of one and routing could “rotate” among the different backbone graphs. An obvious extension would allow $k \geq 3$ distinct grids, leading to k different backbone graphs; this comes at the price of larger routing tables whose size would grow linearly in k. Another approach to load balancing may be achieved by combining the virtual nodes idea with a key idea from the MGGR protocol [11] mentioned in Section 1. An important idea in MGGR is to use the backbone graph simply as a “guide” rather than for actual routing. Suppose a node s has a message to send to node t. Rather than forwarding the message to its cluster head h, s determines *for itself* to which backbone node h', node h would have forwarded the message. Node s then attempts to send the message to some node in the cluster of h'. Thus the message moves from cluster to cluster without necessarily using the backbone, thereby reducing the load on backbone nodes.

Acknowledgment. We thank an anonymous referee for useful comments and pointers to the literature, especially for pointing out that the “virtual nodes idea” has been mentioned before.

References

1. K. M. Alzoubi, P.-J. Wan, and O. Frieder. Message-optimal connected dominating sets in mobile ad hoc networks. In *Proceedings of the 3rd ACM international symposium on Mobile ad hoc networking & computing (MobiHoc'02)*, pages 157–164, New York, NY, USA, 2002. ACM Press.
2. 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'01)*, pages 19–27, 2001.

3. P. Bose, L. Devroye, W. S. Evans, and D. G. Kirkpatrick. On the spanning ratio of gabriel graphs and beta-skeletons. In *Proceedings of the 5th Latin American Symposium on Theoretical Informatics (LATIN'02)*, pages 479–493, 2002.
4. P. Bose and P. Morin. Online routing in triangulations. In *Proceedings of the 10th International Symposium on Algorithms and Computation (ISAAC'99)*, pages 113–122, London, UK, 1999. Springer-Verlag.
5. P. Bose, P. Morin, I. Stojmenović, and J. Urrutia. Routing with guaranteed delivery in ad hoc wireless networks. *Wireless Networks*, 7(6):609–616, 2001.
6. E. Chang. Echo algorithms: Depth parallel operations on general graphs. *IEEE Transactions on Software Engineering*, 8(4):391–401, July 1982.
7. M. Damian, S. Pandit, and S. Pemmaraju. Local approximation schemes for topology control. In *Proceedings of the twenty-fifth annual ACM symposium on Principles of distributed computing (PODC'06)*, pages 208–217, New York, NY, USA, 2006. ACM Press.
8. D. P. Dobkin, S. J. Friedman, and K. J. Supowit. Delaunay graphs are almost as good as complete graphs. *Discrete and Computational Geometry*, 5(4):399–407, 1990.
9. D. Eppstein. Spanning trees and spanners. In Jörg-Rudiger Sack and Jorge Urrutia, editors, *Handbook of Computational Geometry*, chapter 9, pages 425–461. Elsevier, 2000.
10. Q. Fang, J. Gao, L. Guibas, V. Silva, and L. Zhang. GLIDER: Gradient landmark-based distributed routing for sensor networks. In *Proceedings of the 24th Conference of the IEEE Communication Society (INFOCOM'05)*, volume 1, pages 339–350, March 2005.
11. S. Funke and N. Milosavljević. Guaranteed-delivery geographic routing under uncertain node locations. In *Proceedings of the 26th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM'07)*, 2007.
12. J. Gao, L. J. Guibas, J. Hershberger, L. Zhang, and A. Zhu. Geometric spanner for routing in mobile networks. In *Proceedings of the 2nd ACM international symposium on Mobile ad hoc networking & computing (MobiHoc'01)*, pages 45–55, 2001.
13. Y.-J. Kim R. Govindan, B. Karp, and S. Shenker. Lazy cross-link removal for geographic routing. In *Proceedings of the 4th international conference on Embedded networked sensor systems (SenSys'06)*, pages 112–124, New York, NY, USA, 2006. ACM Press.
14. B. Karp and H. T. Kung. GPSR: Greedy perimeter stateless routing for wireless networks. In *Proceedings of the Sixth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom'00)*, pages 243–254, 2000.
15. Y.-J. Kim, R. Govindan, B. Karp, and S. Shenker. On the pitfalls of geographic face routing. In *Proceedings of the 2005 joint workshop on Foundations of mobile computing (DIALM-POMC'05)*, pages 34–43, New York, NY, USA, 2005. ACM Press.
16. F. Kuhn, R. Wattenhofer, Y. Zhang, and A. Zollinger. Geometric ad-hoc routing: Of theory and practice. In *Proceedings of the twenty-second annual symposium on Principles of distributed computing (PODC'03)*, pages 63–72, New York, NY, USA, 2003. ACM Press.
17. F. Kuhn, R. Wattenhofer, and A. Zollinger. Ad-hoc networks beyond unit disk graphs. In *Proceedings of the 2003 joint workshop on Foundations of mobile computing (DIALM-POMC'03)*, pages 69–78, New York, NY, USA, 2003. ACM Press.
18. F. Kuhn, R. Wattenhofer, and A. Zollinger. Worst-case optimal and average-case efficient geometric ad-hoc routing. In *Proceedings of the 4th ACM international symposium on Mobile ad hoc networking & computing (MobiHoc'03)*, pages 267–278, New York, NY, USA, 2003. ACM Press.
19. X.-Y. Li, W.-Z. Song, and W. Wang. A unified energy-efficient topology for unicast and broadcast. In *Proceedings of the 11th annual international conference on Mobile computing and networking (MobiCom'05)*, pages 1–15, New York, NY, USA, 2005. ACM Press.
20. T. Moscibroda and R. Wattenhofer. Minimizing interference in ad hoc and sensor networks. In *Proceedings of the 2005 joint workshop on Foundations of mobile computing (DIALM-POMC'05)*, pages 24–33, New York, NY, USA, 2005. ACM Press.
21. D. Peleg. *Distributed computing: a locality-sensitive approach*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2000.
22. Y. Wang and X.-Y. Li. Localized construction of bounded degree and planar spanner for wireless ad hoc networks. In *Proceedings of the Joint Workshop on Foundations of Mobile Computing (DIALM-POMC'03)*, pages 59–68, 2003.