Ad-Hoc Networks Beyond Unit Disk Graphs*

Fabian Kuhn, Roger Wattenhofer, Aaron Zollinger
Department of Computer Science
ETH Zurich
8092 Zurich, Switzerland
{kuhn, wattenhofer, zollinger}@inf.ethz.ch

ABSTRACT

In this paper we study a model for ad-hoc networks close enough to reality as to represent existing networks, being at the same time concise enough to promote strong theoretical results. The Quasi Unit Disk Graph model contains all edges shorter than a parameter d between 0 and 1 and no edges longer than 1. We show that—in comparison to the cost known on Unit Disk Graphs—the complexity results in this model contain the additional factor $1/d^2$. We prove that in Quasi Unit Disk Graphs flooding is an asymptotically message-optimal routing technique, provide a geometric routing algorithm being more efficient above all in dense networks, and show that classic geometric routing is possible with the same performance guarantees as for Unit Disk Graphs if $d \geq 1/\sqrt{2}$.

Categories and Subject Descriptors

F.2.2 [**Analysis of Algorithms and Problem Complexity**]: Nonnumerical Algorithms and Problems—*geometrical problems and computations, routing and layout*;
G.2.2 [**Discrete Mathematics**]: Graph Theory—*network problems*;
C.2.2 [**Computer-Communication Networks**]: Network Protocols—*routing protocols*

General Terms

Algorithms, Performance, Theory

Keywords

Ad-Hoc Networks, Face Routing, Flooding, Geometric Routing, Mobile Computing, Modeling, Unit Disk Graphs, Wireless Communication.

*The work presented in this paper was supported (in part) by the National Competence Center in Research on Mobile Information and Communication Systems (NCCR-MICS), a center supported by the Swiss National Science Foundation under grant number 5005-67322.

1. INTRODUCTION

Ad-hoc networks are formed by mobile devices communicating via radio. If two ad-hoc nodes are too far apart to communicate directly, intermediate nodes can relay their messages.

A widely employed model for the study of ad-hoc networks is the so-called *Unit Disk Graph* model: Nodes are located in the Euclidean plane and are assumed to have identical (unit) transmission radii. Consequently an edge between two nodes u and v—representing that u and v are in mutual transmission range—exists iff $|uv|$, their Euclidean distance, is not greater than one. On the one hand, clearly, this is a glaring simplification of reality, since, even if all network nodes are homogeneous, this model does not account for the presence of obstacles, such as walls, buildings, mountains—or also weather conditions—, which might obstruct signal propagation. On the other hand, Unit Disk Graphs are simple enough to promote strong theoretical results (such as the cost for routing [20, 21] or topology control [1, 11, 30]).

In this paper we study a graph model which is considerably closer to reality [4]. We maintain the assumption that all mobile nodes are placed in the plane (that is, they have coordinates in $\mathbb{R}^2$). In a *Quasi Unit Disk Graph*, two nodes are connected by an edge if their distance is less than or equal to d, d being a parameter between 0 and 1. Furthermore, if the distance between two nodes is greater than 1, there is no edge between them. In the range between d and 1 the existence of an edge is not specified.

In this paper we first establish a constructive lower bound for Quasi Unit Disk Graphs showing that basically any algorithm without routing tables requires sending of $\Omega\big((\frac{c}{d})^2\big)$ messages to route from a source s to a destination t, where c is the length of the shortest path between s and t. We show that, with the aid of a topology control graph structure, a restricted flooding algorithm is guaranteed not to perform worse and that this technique is consequently asymptotically message-optimal.

If we attribute the network nodes with information about their own and their neighbors' positions and assume that the message source knows the position of the destination—the basic assumptions of *geometric routing*—, a more subtle approach than flooding of the network is possible. We present a combination of greedy routing and restricted flooding. This yields a routing algorithm that is still asymptotically opti-

mal in the worst case, but also efficient in the average case, as previous work on average-case efficiency of geometric ad-hoc routing algorithms suggests [6, 22]. Finally, if we assume d to be at least $1/\sqrt{2}$, we show that it is possible to locally introduce virtual edges and perform the classic variations of geometric routing while preserving performance guarantees known from Unit Disk Graphs.

After discussing related work in the following section, we state the model and provide definitions in Section 3. In Section 4 we establish a lower bound for the message complexity of so-called volatile memory routing algorithms. Section 5 contains the description of the topology control structure forming the basis for the subsequent algorithms. Section 6 provides the analysis of flooding algorithms with respect to message and time complexity. Section 7 discusses the combination of flooding with a greedy approach for geometric routing, whereas Section 8 shows that for large enough d, classic geometric routing can be employed. Section 9 draws the conclusions of the paper.

2. RELATED WORK

So far, the most popular network structure to model mobile ad-hoc networks has been the Unit Disk Graph. The underlying assumption of this model is that the nodes are placed in the plane, all of them having the same—normalized to one—transmission range. A more general model is provided by disk graphs where in contrast to Unit Disk Graphs, nodes can have different transmission ranges. Disk graphs have also been widely used, but, while for Unit Disk Graphs a number of theoretical results have been achieved, most of the knowledge on disk graphs bases on simulations. Disk graphs provide a simple method to analyze unidirectional links, however, it is not possible to model any kind of obstacles. Barrière et al. [4] describe a model which is—up to scaling—identical to our Quasi Unit Disk Graph model. Similarly to the technique stated in Section 8, [4] exploits local construction of virtual links to allow for correct (i.e. arrival of message guaranteed) geometric routing on Quasi Unit Disk Graphs for $d \geq 1/\sqrt{2}$. In Section 8, we extend this result towards efficiency.

In this paper, we give different complexity results concerning the Quasi-UDG model. We show how to construct a subgraph of the network graph G which enables cost-optimal flooding and we show how the flooding overhead can be reduced (in practice) by using geometric routing for $d \geq 1/\sqrt{2}$ and a combination of geometric routing and flooding for arbitrary d. Constructing a sub-structure G' of G such that G' features some desirable properties is often termed topology control. Topology control is used to reduce the number of nodes and the number of edges involved in protocols such as routing. An important issue of such a precomputation is the reduction of interference effects, message complexity, or energy consumption. Sometimes, algorithms need network graphs with special properties; for example all face-routing based geometric routing algorithms [6, 16, 18, 20, 21, 22] need a planar graph to operate correctly. For Unit Disk Graphs, a number of different ideas in order to reduce the complexity of the network topology have been proposed. Many of them are based on dominating sets [1, 10, 11, 13, 14, 19, 29] or angle of arrival [30]. For finding planar subgraphs, various constructions of different quality and complexity have been conceived [9, 11, 28]. A recent survey on topology control algorithms for ad-hoc networks can be found in [26].

Flooding is an essential ingredient of many ad-hoc routing algorithms [15, 25]. It is therefore crucial to reduce the number of messages sent. One way to reduce the cost of flooding is to lower the complexity of the network by using appropriate topology control mechanisms. Apart from this there are other approaches which try to optimize flooding performance by using geometric information about the destination [5, 17]. These algorithms differ from our greedy routing/flooding approach in that they only try to flood into the right direction but they do not apply real geometric routing whenever possible.

Geometric routing (also known as location-based, position-based or geographic routing) has also mainly been studied on Unit Disk Graphs. Greedy routing algorithms have been studied in [8, 11, 12, 18, 27]. Greedy routing behaves well in practice, but no guarantee can be given about the arrival of messages for all of them. The first algorithms with guaranteed delivery were [6, 18] followed by a slightly changed approach in [16]. The first geometric routing algorithm whose cost is bounded by a function of the cost c of an optimal path was given in [21]. The cost $O(c^2)$ was shown to be optimal and the routing scheme of [21] was later extended to also achieve practical efficiency [20, 22].

3. MODEL

This section provides definitions of the model employed in this paper. We first give a formal definition of our ad-hoc network model:

DEFINITION 3.1. **(Quasi Unit Disk Graph)** *Let $V \subset \mathbb{R}^2$ be a set of points in the 2-dimensional plane and $d \in [0, 1]$ be a parameter. The symmetric Euclidean graph (V, E), such that for any pair of points $u, v \in V$*

- *$(u, v) \in E$ if $|uv| \leq d$ and*
- *$(u, v) \notin E$ if $|uv| > 1$,*

is called a Quasi Unit Disk Graph (Quasi-UDG) with parameter d.

In the subsequent section we establish a lower bound for the message complexity of so-called volatile memory routing algorithms. With this model nodes are attributed with a short-term memory in which for each message a constant number of bits may be stored temporarily.

DEFINITION 3.2. **(Volatile Memory Routing Algorithm)** *The task of a volatile memory routing algorithm is to transmit a message from a source s to a destination t on a graph, where each node of the graph holds a memory in which $O(\log n)$ bits[1] may be stored as long as the message is en route.*

[1]We need a logarithmic number of bits to enable all nodes to concurrently flood the network.

In particular this model allows to store message identifiers required for flooding (cf. Section 6).

The second important algorithm model discussed in this paper is *geometric* routing [21]:

DEFINITION 3.3. **(Geometric Routing Algorithm)** *The task of a geometric routing algorithm is to transmit a message from a source s to a destination t on a graph while observing the following rules:*

- *Every node is informed about its own and all of its neighbors' positions.*
- *The source of a message knows the position of the message destination.*
- *A message may contain control information about at most* O(1) *nodes.*
- *A node is only allowed to temporarily store a message before retransmission; no other memory is available.*

As stated above, in original geometric routing a node is allowed to store messages only temporarily before relaying them. In order to enable an algorithm to employ flooding, this assumption can be relaxed:

DEFINITION 3.4. **(Geometric Volatile Memory Routing Algorithm)** *A geometric volatile memory routing algorithm is a volatile memory routing algorithm additionally observing the first three rules of the definition of geometric routing algorithms.*

In the following we provide a concise overview of basic concepts of distributed computing vital for the understanding of this paper. More detailed descriptions can be found in textbooks, such as in [23].

At certain points of the paper we have to distinguish between the *synchronous* and the *asynchronous* model of distributed computation. In the *synchronous* model, communication delays are assumed to be bounded. As a consequence it can be assumed that all processes running on different network nodes perform their message sending and receiving operations in simultaneous and globally clocked rounds. In the *asynchronous* model, message delays are unbounded. No assumptions can be made on the duration of single process operations.

Two fundamental measures in distributed computing are *message* and *time complexity*. The *message complexity* of a distributed algorithm is the total number of messages sent during its execution. The definition of *time complexity* depends on the synchrony model: In the synchronous model, time complexity is the total number of rounds elapsed between algorithm start and algorithm termination. In the asynchronous model such a simple time model cannot naturally be obtained, since the transmission delay of a message is unbounded. The common solution to this is the assumption that the message delay is at most one time unit.

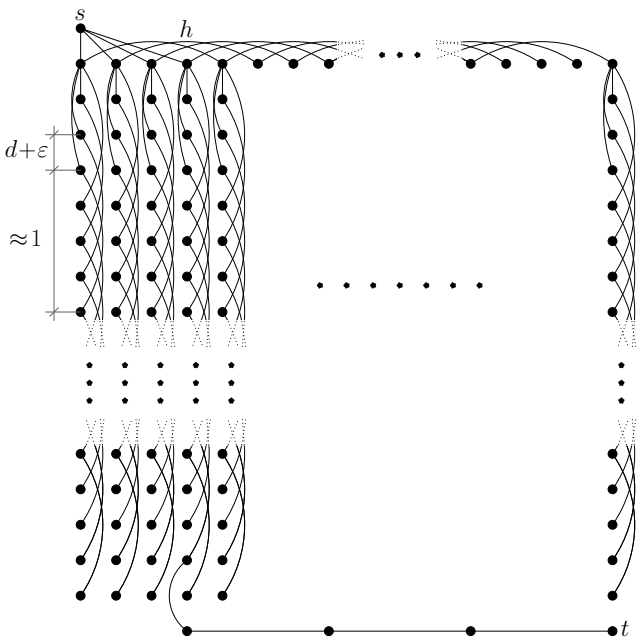

Figure 1: Message Complexity Lower Bound for Volatile Memory Routing Algorithms on Quasi-UDGs

Finally, since we consider message complexities in this paper, we define the *cost of a path* according to the link distance metric, that is, the cost of a path is the number of edges on the path. Similarly, we consider *spanner* graphs with respect to the link distance metric: A graph $G' = (V, E')$ is a spanner of a graph $G = (V, E)$ with stretch factor k iff for any pair of nodes (u, v) the cost of the shortest path on G' is at most k times the cost of the shortest path on G.

4. LOWER BOUND

In this section, we present a lower bound on the message complexity of any volatile memory routing algorithm.

THEOREM 4.1. *Let c be the cost of a shortest path from s to t. There exist graphs on which any (randomized) volatile memory routing algorithm has (expected) message complexity* $\Omega((\frac{c}{d})^2)$.

PROOF. We provide a constructive proof by describing a class of graphs for which the theorem holds.

The basic element used for the construction of these graphs is formed by k nodes (k to be determined later) equidistantly placed on a line, such that the distance between two adjacent nodes is $d + \varepsilon$ for a small $\varepsilon > 0$ (cf. vertical chains in Figure 1). There exists an edge between every pair of nodes (u, v), such that $(\lceil \frac{1}{d} \rceil - 1)\, d < |uv| \leq 1$, that is, the nodes are connected by all the edges with maximum Euclidean length not greater than 1. In addition there is a head node having an edge to each one of the first $\lceil \frac{1}{d} \rceil - 1$ nodes on the line (the head node has to be located such that all additional

edges have length at most 1). As shown in Figure 1, k such vertical chains are placed side by side with distance $d + \varepsilon$ such that the nodes form a matrix. The head nodes of these chains now are interconnected in a way that they among themselves have the same chain structure (uppermost row in Figure 1) with their head node (of second order) denoted by s. The node t—located near the bottom right corner of the node matrix—is connected to one of the end nodes of exactly one of the vertical chains by a simple chain of nodes. Note that the constructed graph is a Quasi Unit Disk Graph.

The main property posing a problem for a routing algorithm is that a matrix column consists of $\lceil \frac{1}{d} \rceil - 1$ interleaved chains which are only connected via the head node. (The same also holds for the first matrix row.) Consequently only one of s's neighbors leads to h, the head node of the column connected to t, and only one of h's neighbors leads to the bottom node connected to t. Since a volatile memory routing algorithm has no a priori information about the graph structure, a deterministic algorithm has to explore every matrix node before finding the path to t. (For a randomized algorithm t can be connected to the matrix such that the algorithm has to explore roughly half of the matrix nodes in expectation.) A volatile memory routing algorithm therefore has to send $\Omega(n)$ messages, where n is the total number of nodes. The optimal path on the other hand—almost exclusively using edges of length nearly 1—has cost about $2\,k \cdot d$, which—together with $k \approx \sqrt{n}$—establishes the theorem. □

5. TOPOLOGY CONTROL

In the previous section we introduced a lower bound graph class, on which any volatile memory routing algorithm cannot find the destination with message complexity less than $\Omega((\frac{c}{d})^2)$. In this section we now describe how to obtain a subgraph of a given Quasi Unit Disk Graph which forms the basis for our algorithms matching the lower bound. This *Backbone Graph* features two important properties exploited for routing: (1) It contains in a given area A at most $\mathrm{O}\big(\frac{A}{d^2}\big)$ nodes and (2) it is a $\mathrm{O}\big(\log(\frac{1}{d})\big)$-spanner.

Given a Quasi Unit Disk Graph G, the Backbone Graph is constructed in three steps. Steps 1 and 2 can be performed by a standard distributed algorithm (as described in [1]) by having the nodes send `dominator` and `connector` messages. We do not discuss the details of this algorithm for space reasons.

1. The first step consists of a clustering process. We construct a Maximal Independent Set *MIS* of nodes on G. Note that since *MIS* is an independent set on G, any two nodes in *MIS* have distance greater than d and consequently a given area A contains at most $\mathrm{O}\big(\frac{A}{d^2}\big)$ nodes in *MIS*. For the purpose of routing, the nodes in *MIS* will later become cluster heads: Since the nodes in *MIS* also form a Dominating Set, any node in G will have at least one node from *MIS* within its neighborhood and will choose one of these as its cluster head.

2. In a second step the cluster heads are linked together by connector nodes, resulting in the Complete Backbone Graph G_{CBG}. Note that since *MIS* is a Dominating Set, the cluster heads can be connected by bridges consisting of at most two nodes. Also note that G_{CBG} is a constant-stretch spanner of G.

3. G_{CBG} can contain $\Omega(\frac{A}{d^4})$ nodes in a given area A, which is too many by a factor of $\frac{1}{d^2}$ compared to the lower bound. The size of *MIS* matching the lower bound, the third step now reduces the number of connecting bridges between cluster heads. Let $G^{(v)}_{CBG}$ denote the graph with node set *MIS* and (virtual) edges between all nodes connected by bridges in G_{CBG}. Our objective is now to construct a subgraph $G^{(v)}_{BG}$ of $G^{(v)}_{CBG}$ with $\mathrm{O}\big(\frac{A}{d^2}\big)$ (virtual) edges within the area A. It eventually follows that the final Backbone Graph G_{BG}—where the (virtual) edges in $G^{(v)}_{BG}$ have again been replaced by connector nodes and their adjacent edges—contains at most $\mathrm{O}\big(\frac{A}{d^2}\big)$ nodes within the area A.

   In order to obtain a graph $G^{(v)}_{BG}$ with the desired property, the plane is divided by a grid into square cells of side length 6. In each cell z all nodes and edges completely contained within z temporarily form a local network. (Note that we assume for this operation that the nodes are informed about their positions.) The number of nodes contained within z is at most $\mathrm{O}\big(\frac{1}{d^2}\big)$. We now apply an algorithm constructing a sparse spanner [2, 23, 24] to reduce the number of edges contained in z to $\mathrm{O}\big(\frac{1}{d^2}\big)$.[2] This procedure is repeated three times on grids with their origin shifted by $(3,0)$, $(0,3)$, and $(3,3)$ relative to the origin of the first grid (cf. Figure 2). (Note that these are local operations, since the subgraphs are of bounded size.) The edge set of graph $G^{(v)}_{BG}$ is finally formed by the union of all edges resulting from the edge reduction steps on all four grids.

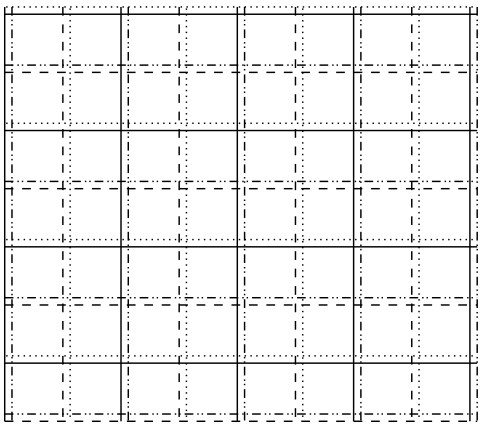

Figure 2: Construction of a Sparse Spanner: Grid Structure

LEMMA 5.1. *In a given area A (with constant extension in each direction) the number of nodes and the number of edges in the Backbone Graph are both bounded by* $\mathrm{O}\big(\frac{A}{d^2}\big)$.

[2]The mentioned algorithm constructs for a constant $\kappa \geq 1$ an $\mathrm{O}(\kappa)$-spanner with at most $n^{1+1/\kappa}$ edges. Setting $\kappa = \log n$ and since $n = \frac{1}{d^2}$, we obtain a graph with the required properties.

PROOF. The grids employed for the edge reduction steps are chosen to have two properties: (1) Every edge in $G^{(v)}_{CBG}$ is completely contained in at least one cell and (2) any region (with constant extension in each direction) is intersected by at most a constant number of grid cells (for instance a square of side length 3 can be intersected in total by at most 9 grid cells). Property (1) guarantees that every edge is considered at least with one of the four grids: Together with the fact that the edge reduction step does not alter the number of components in a cell subgraph, it follows that the number of components in the complete graph is not altered either. Since each resulting subgraph contains at most $O\big(\frac{1}{d^2}\big)$ edges and together with Property (2), it follows that also the union of all remaining edges—that is the number of edges in $G^{(v)}_{BG}$ is not greater than $O\big(\frac{1}{d^2}\big)$ for a constant region. The fact that each edge in $G^{(v)}_{BG}$ corresponds to at most two nodes and three edges in G_{BG} and $G^{(v)}_{BG}$ having at most $O\big(\frac{A}{d^2}\big)$ nodes for an area A (the nodes in *MIS*) finally leads to the lemma. □

LEMMA 5.2. *The Backbone Graph G_{BG} is a spanner of G_{CBG} with stretch factor $O\big(\log(\frac{1}{d})\big)$.*

PROOF. Every edge in $G^{(v)}_{CBG}$ is contained in at least one grid cell and consequently also considered in at least one of the according subgraphs. Since the edges retained in each subgraph form a $O\big(\log(\frac{1}{d})\big)$-spanner (on the subgraph), this property also holds for the union of all subgraphs, $G^{(v)}_{BG}$. Finally, each edge in $G^{(v)}_{BG}$ resulting in at most three edges in G_{BG}, the lemma follows. □

In distributed computing a distinction can be made between the *one-hop broadcast* model and the *point-to-point communication* model: In the one-hop broadcast model a node can simultaneously send a message to all its neighbors, whereas in the point-to-point communication model a message is sent over an edge to one distinct neighbor. The algorithms described in the remaining sections are assumed to execute on G_{BG}. Since on this graph the number of nodes and the number of edges are asymptotically equal in a given area, the two models can be employed interchangeably, depending on whether we argue over the number of nodes or edges in the graph.

When routing a message m from a source s' to a destination t', the nodes s' and t' will in general not be cluster heads. The complete process of routing therefore consists of

1. s' sending m to its associated cluster head s,
2. routing m from s to t, the cluster head associated to t', and
3. t sending m to t'.

Since steps 1 and 3 incur only constant cost with respect to both message and time complexity, we exclusively consider step 2 in the remaining part of the paper. Whenever mentioning a source s or a destination t we therefore assume that s and t are cluster heads.

6. MESSAGE-OPTIMAL FLOODING

In this section we discuss the message and time complexities of the Echo algorithm on Quasi Unit Disk Graphs. Due to space reasons we only give a short outline of the algorithm execution; more detailed information can be found in [7, 23]. The Echo algorithm consist of a flooding phase and an echo phase.

- The flooding phase is initiated by the source s by sending a flooding message—containing a Time To Live (TTL) counter τ—to all its neighbors. Each node receiving the flooding message for the first time decrements the TTL counter by one and retransmits the message to all its neighbors (with the exception of the neighbor it received the message from). In the synchronous model this flooding phase constructs a Breadth First Search (BFS) tree.
- From the leaves of this tree—the nodes where the τ counter reaches 0—echo messages are sent back to to the source along the BFS tree constructed during the flooding phase. An inner node in the BFS tree can decide locally when to send an echo message to its parent in the tree by awaiting the receipt of an echo message from all of its children.

By initiating the first flooding phase with τ set to 1 and relaunching a flooding phase with doubled τ whenever the echo messages indicate that the destination has not yet been reached, both time and message complexities can be bounded:

THEOREM 6.1. *Employed on G_{BG} in the synchronous model, the Echo algorithm reaches the destination with message complexity $O\big((\frac{c}{d})^2\big)$ and time complexity $O\big(c \cdot \log(\frac{1}{d})\big)$, where c is the cost of a shortest path between s and t. This is asymptotically optimal with respect to message complexity.*

PROOF. The Echo algorithm floods the complete network with message complexity $O(m)$ and time complexity $O(D)$, where m is the number of edges in the network and D is the diameter of the network. Since no edge in G_{BG} is longer than 1, all nodes reached with a certain τ lie within the circle centered at s with radius τ. The number of edges within this circle is bounded by $O\big((\frac{\tau}{d})^2\big)$. Note that the destination is reached at the latest for $\tau = 2 \cdot c$. Since τ is doubled after each failure, the total number of visited edges is dominated by the number of edges in the circle with maximum τ, from which the message complexity follows. Asymptotic optimality follows from the lower bound established in Section 4.

The time complexity follows from the fact that the BFS tree constructed during the flooding phase contains a shortest path from s to t. Since G_{BG} is a $\log(\frac{1}{d})$-spanner of G, the shortest path on G_{BG}, on which the algorithm is executed, is $c \cdot \log(\frac{1}{d})$. The time complexity of a single flooding-echo round being proportional to τ and again the total time complexity asymptotically being dominated by the maximum τ used, the time complexity follows. □

In the asynchronous model the synchronizer construction introduced in [3] can be employed.

THEOREM 6.2. *When employed on G_{BG} in the asynchronous model, the Echo algorithm reaches the destination with message complexity $O\big((\frac{c}{d})^2 \cdot \log^3(\frac{c}{d})\big)$ and time complexity $O\big(c \cdot \log(\frac{1}{d}) \cdot \log^3(\frac{c}{d})\big)$, where c is the cost of the shortest path between s and t.*

PROOF. The synchronizer construction introduced in [3] incurs a cost factor of $O\big(\log^3(\frac{1}{d})\big)$ with respect to both message and time complexity. Plugging in the above Echo algorithm for the synchronous model yields the lemma. □

For geometric routing as discussed in the following section, a variant of Echo can be defined by replacing the Time To Live counter by a geometric argument: The flooding message is retransmitted only by nodes located within a circle centered at s with a certain radius r.

LEMMA 6.3. *The geometric Echo algorithm reaches t with message and time complexity $O\big((\frac{c}{d})^2\big)$, where c is the link cost of the shortest path. This holds for both the synchronous and the asynchronous model and is asymptotically optimal with respect to message complexity.*

PROOF. In contrast to the above Echo algorithm using TTL, all nodes located within the restricting circle centered at s with radius r participate in the execution of the geometric algorithm. This circle containing at most $O\big((\frac{r}{d})^2\big)$ nodes, the message complexity follows, where the remaining reasoning is analogous to the one in the proof of Theorem 6.1. The time complexity follows from the fact that time complexity cannot be greater than message complexity. □

7. GREEDY ECHO ROUTING

Although asymptotically message-optimal, a flooding-based algorithm is prohibitively expensive in most networks for practical purposes. Previous work showed that this problem can often be tackled by combining a correct routing algorithm (that is guaranteed to find the destination) with a greedy routing scheme [6, 22]. In this section we therefore describe a geometric volatile memory routing algorithm that tries to leverage the advantages of a greedy routing approach with respect to both conceptual simplicity and message-efficiency: In order to route a message, a node simply forwards it to its neighbor closest to the destination. Greedy routing can however run into a local minimum with respect to the distance to the destination, that is a node without any neighbors closer to t. In our case such a local minimum is circumvented by employment of restricted flooding, in particular by the aid of the geometric Echo algorithm as described in the previous section. In this section we therefore refer by Echo to the geometric Echo algorithm. We denote with Echo_r the subalgorithm of geometric Echo consisting of the flooding and the corresponding echo phase for the radius r.

Our algorithm GEcho combines both greedy routing and flooding in two modes: Generally the message is forwarded in greedy mode as long as possible. Whenever running into a local minimum, the algorithm switches to echo mode. In order to keep the cost of flooding-based echo low, the algorithm tries to fall back to greedy mode as early as possible. The fallback criterion is chosen such that the combined routing algorithm is asymptotically optimal with respect to message complexity. In particular, the Echo algorithm does not terminate only when finding t, but already when finding a node v which is significantly closer to t than the local minimum, as described in Step 2 of the GEcho algorithm:

GEcho

The value q is a constant parameter chosen prior to algorithm execution such that $0 < q \leq 1$.

0. Start at s.
1. **(Greedy Mode)** Forward the message to the neighbor in G closest to t. If t is reached, terminate. If a local minimum is reached, continue with step 2, otherwise repeat step 1 at the next node.
2. **(Echo Mode)** Execute algorithm Echo starting at the local minimum u until either reaching t—in which case the algorithm terminates—or finding a node v, such that $|ut| - |vt| \geq q \cdot r$, where r is the currently chosen radius in Echo's subalgorithm Echo_r. Proceed to v and continue with step 1.

In the following we obtain a statement on the asymptotic complexity of the algorithm. We first show that the number of messages sent in Greedy mode is bounded:

LEMMA 7.1. *The number of messages sent in Greedy mode is bounded by $O\big((\frac{c}{d})^2\big)$.*

PROOF. Let us exclusively consider the sequence U of nodes sending messages in Greedy mode or receiving messages sent in greedy mode during the execution of the algorithm. Note that the distance to t is strictly decreasing within U. Since the algorithm stays in greedy mode until reaching a local minimum, U is partitioned into subsequences $U_1, U_2, \ldots, U_k, k \geq 1$ of nodes by the occurrence of local minima: A local minimum only receives a Greedy message without being able to send it to a subsequent node in Greedy mode. Within a subsequence $U_i = u_1, u_2, \ldots, u_{\ell_i}$, $\ell_i \geq 2$ any two nodes $u_j, u_{j+2}, 1 \leq j \leq \ell_i - 2$ have distance greater than d (otherwise u_j would have sent the Greedy message directly to u_{j+2}). On the other hand also the distance between a local minimum u_{ℓ_i} and the first node in the following subsequence U_{i+1} have distance greater than d (otherwise u_{ℓ_i} would be no local minimum). Together with the fact that all nodes in U are located within the circle C centered at t with radius $|st|$, the number of nodes in the total sequence U is therefore bounded by two times the maximum number of nodes with relative distance greater than d—or likewise the maximum number of nonintersecting disks of radius $d/2$—that can be placed within C. With $|st| \leq c$, the lemma follows. □

We now confine ourselves to the number of messages sent in Echo mode. Note that after each *round*, defined to be one execution of Step 1 or Step 2, the algorithm is strictly closer to t than before that round.

LEMMA 7.2. *For a given r the subalgorithm* Echo_r *is executed at most* $\lceil \frac{|st|}{q\,r} - 1 \rceil$ *times.*

PROOF. According to the criterion described in Step 2, an Echo round initiated at node u terminates—unless arriving at t—only if it finds a node v, such that $|ut| - |vt| \geq q \cdot r$. For any r (also if at a particular node Echo_r fails and r is doubled) such a progress can be made at most $\lceil \frac{|st|}{q\,r} - 1 \rceil$ times, since after each round the algorithm is strictly closer to t than before. □

With this property we can obtain the total number of messages sent in Echo mode during algorithm execution.

LEMMA 7.3. *The total number of messages sent in Echo mode is at most* $\mathrm{O}\big((\frac{c}{d})^2\big)$.

PROOF. We obtain the total number of messages sent in Echo mode by summing up over all nodes ever contained in a circle bounding Echo_r. Since the number of nodes contained in a given circular area is asymptotically proportional to the size of the area, it is sufficient to compute the total area covered by all Echo_r bounding circles. Let $r_i = 2^i$, $i = 1, 2, 3, \ldots$ denote the radii of the Echo bounding circles. The maximum r_i can be found by the observation that (1) all Echo restricting circles have their centers at a node closer to t than s and (2) the circle centered at any node closer to t than s having radius $2\,c$ completely contains the shortest path. Since the value of r in Echo_r is obtained by doubling, the maximum r_i used over all is less than $4\,c$; the maximum i reached is consequently $\lceil \log(4\,c) \rceil$. With n_i being the total number of bounding circles used with radius r_i, we obtain

$$A = \sum_{i=0}^{\lceil \log(4\,c) \rceil} n_i \cdot \pi r_i^2$$

for the total covered area A. Using Lemma 7.2 we obtain

$$\begin{aligned} A &\leq \pi \cdot \sum_{i=0}^{\lceil \log(4\,c) \rceil} \lceil \frac{|st|}{q\,r_i} - 1 \rceil \cdot r_i^2 \\ &< \pi \cdot \sum_{i=0}^{\lceil \log(4\,c) \rceil} \frac{|st|}{q} \cdot r_i \underset{(|st| \leq c)}{\leq} \frac{\pi\,c}{q} \cdot \sum_{i=0}^{\lceil \log(4\,c) \rceil} 2^i \\ &= \frac{\pi\,c}{q} \cdot \frac{2^{\lceil \log(4\,c) \rceil + 1} - 1}{3} \in \mathrm{O}\big(c^2\big) . \end{aligned}$$

The Area A containing at most $\mathrm{O}\big(\frac{A}{d^2}\big)$ nodes (cf. Section 5), the lemma follows. □

LEMMA 7.4. *The algorithm GEcho finds the destination with both message and time complexity* $\mathrm{O}\big((\frac{c}{d})^2\big)$, *where c is the link cost of the shortest path.*

PROOF. The message complexity bound follows directly from the previous two lemmas. The time complexity bound follows from the fact that time complexity cannot be greater than message complexity. □

THEOREM 7.5. *The algorithm GEcho is asymptotically optimal with respect to message complexity.*

PROOF. Follows from Lemma 7.4 and Section 4. □

8. LARGE D-VALUES

This section treats the special case where the parameter d of the Quasi Unit Disk Graph G is $d \geq 1/\sqrt{2}$. This case has already been considered by Barrière et al. in [4]. There, it is shown that for $d \geq 1/\sqrt{2}$ standard geometric routing is possible. Here, we extend their results and present a geometric routing algorithm which is asymptotically optimal, i.e. whose cost is quadratic in the cost of an optimal path (cf. [21]).

The structural difference between Quasi-UDGs for $d < 1/\sqrt{2}$ and Quasi-UDGs for $d \geq 1/\sqrt{2}$ lies in the local environment of intersecting edges. If $d \geq 1/\sqrt{2}$, all intersections can be detected locally. This is shown by the following two lemmas.

LEMMA 8.1. *Let $e = (u, v)$ be an edge and w be a node which is in the disk with diameter (u, v). Either u and w or v and w are connected by an edge.*

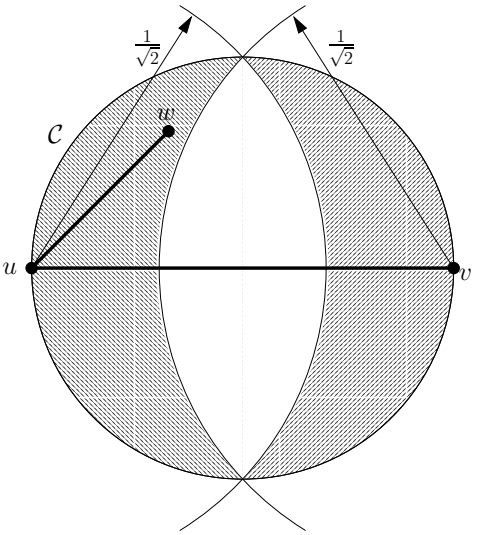

Figure 3: w is either connected to u or to v

PROOF. The following proof is illustrated by Figure 3. Because $|uv| \leq 1$, the regions of the points whose distances to u and v are greater than $1/\sqrt{2}$ do not intersect inside $\mathcal{C}$ (hatched areas in Figure 1). Thus either $|uw| \leq 1/\sqrt{2}$ or $|vw| \leq 1/\sqrt{2}$. In the picture, this holds for u and w and therefore there is an edge between the two nodes in G. □

LEMMA 8.2. *Let $e_1 = (u_1, v_1)$ and $e_2 = (u_2, v_2)$ be two intersecting edges in a Quasi-UDG G with parameter $d \geq 1/\sqrt{2}$. Then at least one of the edges (u_1, u_2), (u_1, v_2), (v_1, u_2), and (v_1, v_2) exists in G.*

PROOF. We have to show that one of the four sides of the quadrangle (u_1, u_2, v_1, v_2) is shorter than $1/\sqrt{2}$. Because the sum of the interior angles of the quadrangle is 2π, at least one of the angles has to be greater or equal to $\pi/2$. W.l.o.g. let it be the angle at node u_2. In this case u_2 lies in the disk with diameter (u_1, v_1) and the lemma follows from Lemma 8.1. □

We will now give an overview of the results of [4]. The algorithm consists of three steps. In a first step, the Quasi-UDG G is extended by adding virtual edges. Whenever there is an edge (u, v) and a node w which is inside the circle with diameter (u, v), for at least one of the nodes u and v—w.l.o.g. let it be u—the distance to w is smaller than or equal to $1/\sqrt{2}$ (Lemma 8.1) and therefore u has a connection to w. If there is no edge between v and w, a virtual edge is added. Sending a message over this virtual edge is done by sending the message via node u. This process is done recursively, i.e. also if (u, v) is a virtual edge. The graph obtained by adding the virtual edges to G is called super-graph $S(G)$. Barrière et al. prove that on $S(G)$ the Gabriel Graph $\mathbf{GG}(S(G))$ can be constructed yielding a planar subgraph of $S(G)$. In the Gabriel Graph construction, an edge (u, v) is removed if and only if there is a node w in the disk with diameter (u, v) [9]. On $\mathbf{GG}(S(G))$, any correct geometric routing algorithm is applied ([6, 18]).

In order to obtain an optimal geometric routing algorithm, we have to change the algorithm of [4] in two ways: i) the planar graph which we need for geometric routing should be a spanner and the number of nodes in a given area A should not be larger than $\mathrm{O}(A)$, and ii) we have to replace the geometric routing algorithm by a more elaborate variant such as AFR [21] or one of its successors GOAFR [22] and GOAFR+ [20]. One of the bounding factors for the spanning property is given by the recursive depth of the virtual edge construction, i.e. the length of paths corresponding to virtual edges. From [4] we have the following result.

LEMMA 8.3. *Let λ be the minimum Euclidean distance between any two nodes. If $d \geq 1/\sqrt{2}$, the length of the route in G corresponding to a virtual edge in $S(G)$ is at most $1 + \frac{1}{2\lambda^2}$.*

PROOF. The lemma directly follows from Property 1 in Section 5 of [4]. □

We cannot assume that there is a minimum Euclidean distance λ between any two nodes. However, by using the Backbone Graph G_{BG} (cf. Section 5) we obtain a Quasi-UDG with bounded degree, a property which we prove to be equivalent to the minimum distance assumption.

Precisely, we start by constructing G_{BG}. This gives us a set of dominator nodes $D = MIS$ and a set of connector nodes C. We transform G_{BG} into a Quasi-UDG $G'_{BG} = (V', E')$ by setting $V' = D \cup C$ and by including all possible edges of E in E' (all edges between nodes of V').

LEMMA 8.4. *The degree of each node in the Quasi-UDG G'_{BG} is bounded by a constant.*

PROOF. Because the dominator nodes D have distance at least $1/\sqrt{2}$ from each other, the number of dominators which are within three hops from a node $v \in V'$ is bounded by a constant. Only those nodes can add connector nodes which are neighbors of v. Each of them can only add a constant number of connector nodes and therefore, the degree of node v has to be constant. A more detailed proof can be done analogously to the proof for the same lemma for Unit Disk Graphs [1, 29]. □

G'_{BG} is now used for the Gabriel Graph construction. First, virtual edges are added as in the algorithm of [4] resulting in a super-graph $S(G'_{BG})$. Then $\mathbf{GG}(S(G'_{BG}))$ is constructed. Analogously to Lemma 8.3 we are able to get a bound on the maximum route length for any virtual edge.

LEMMA 8.5. *Let $G = (V, E)$ be a Quasi-UDG with maximum node degree Δ. If $d \geq 1/\sqrt{2}$, the length of the route in G corresponding to a virtual edge in $S(G)$ is at most $\mathrm{O}(\Delta^2)$.*

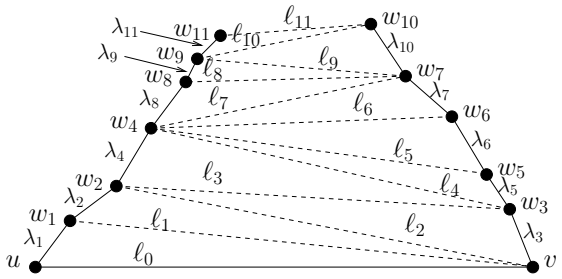

Figure 4: Recursive Depth of Virtual Edges

PROOF. Let $(u, v) \in E$ be an edge of G. Further let $w_1, \ldots, w_k$ be a sequence of nodes which recursively force the creation of new virtual edges e_i for which the corresponding route contains (u, v). Let $\ell_0 := |uv|$ and ℓ_i be the Euclidean length of the virtual edge e_i (see Figure 4 as an explanation). λ_i is the length of the edge which together with e_{i-1} provides the route for e_i ($\lambda_i \leq d$). For the length ℓ_i of the i^{th} virtual edge e_i we get

$$\begin{aligned} \ell_i &\leq \sqrt{\ell_{i-1}^2 - \lambda_i^2} = \sqrt{1 - \frac{\lambda_i^2}{\ell_{i-1}^2}} \cdot \ell_{i-1} \\ &\leq \sqrt{1 - \frac{\lambda_i^2}{\ell_0^2}} \cdot \ell_{i-1} \leq \sqrt{1 - \lambda_i^2} \cdot \ell_{i-1}. \end{aligned}$$

The second inequality follows from $\ell_0 \geq \ell_{i-1}$, the last inequality follows from $\ell_0 \leq 1$. We therefore get

$$\ell_k \leq \prod_{i=1}^{k} \sqrt{1 - \lambda_i^2} \cdot \ell_0 \leq \prod_{i=1}^{k} \sqrt{1 - \lambda_i^2}. \qquad (1)$$

We define $\lambda := 1/k \sum_{i=1}^{k} \lambda_i$ to be the average length of the edges corresponding to the λ_i. It can be shown that the expression of Equation (1) can be upper-bounded by replacing each λ_i by λ:

$$\ell_k \leq \left(\sqrt{1 - \lambda^2}\right)^k = \left(1 - \lambda^2\right)^{k/2}. \qquad (2)$$

In a Quasi-UDG all nodes in a disk with radius $d/2$ are neighbors of each other. Therefore, when starting at a node u, after at most $\Delta + 1$ hops, one has to leave the disk with radius $d/2$ around u. Thus, the sum of the lengths of $\Delta + 1$ successive edges on a cycle-free path has to be greater than

$d/2$, i.e. for $d \geq 1/\sqrt{2}$ this is a constant. The average edge length of any cycle-free path is thus $O(1/\Delta)$. In Figure 4, we see that the λ_i form two paths. Therefore, the average λ_i has to be in the order of $\lambda = O(1/\Delta)$. Because $(1-1/n)^n \leq 1/e$, we set $k = 2/\lambda^2 = O(\Delta^2)$ in (2) and get

$$\ell_k \leq (1-\lambda^2)^{1/\lambda^2} \leq 1/e \leq 1/\sqrt{2}.$$

Because the length of a virtual edge e_k has to be $\ell_k \geq 1/\sqrt{2}$, this concludes the proof. □

LEMMA 8.6. *The Gabriel Graph* $\mathbf{GG}(S(G'_{BG}))$ *is a spanner for the Quasi-UDG* G.

PROOF. By Lemma 8.4 and Lemma 8.5, we see that the virtual edges only impose a constant factor on the cost of a path. We can therefore proceed as if all virtual edges were normal edges of G. Further, it is well known that the Gabriel Graph construction retains an energy-optimal path (edge cost = square of the Euclidean edge length) (see e.g. [21]). This is because whenever an edge is removed, there is an alternative (two-hop) path with lower or equal energy cost. Because the average edge length of $S(G'_{BG})$ is a constant (cf. proof of Lemma 8.5), the number of hops and the energy cost of a path only differ by a constant factor and therefore, the minimum energy path is only by a constant factor longer than the shortest path connecting two nodes. For further details, we refer to the analysis for Unit Disk Graphs [20]. □

THEOREM 8.7. *Let* G *be a Quasi Unit Disk Graph with* $d \geq 1/\sqrt{2}$. *Applying AFR [21], GOAFR [22], or GOAFR+ [20] on* $\mathbf{GG}(S(G'_{BG}))$ *yields a geometric routing algorithm whose cost is* $O(c^2)$, *where* c *is the cost of an optimal path. This is asymptotically optimal.*

PROOF. Because G can be the Unit Disk Graph, the lower bound follows from the lower bound for Unit Disk Graphs in [21]. The number of nodes as well as the number of edges of $\mathbf{GG}(S(G'_{BG}))$ in a given area A is proportional to A and therefore, the $O(c^2)$ cost also directly follows from the respective analyses in [20, 21, 22]. □

8.1 Alternative Construction

We conclude the section on Quasi-UDG for $d \geq 1/\sqrt{2}$ with the description of an alternative construction of a planar graph which can be used to perform geometric routing. By Lemma 8.2, all edge intersections of a Quasi-UDG with $d \geq 1/\sqrt{2}$ can be detected locally (one communication round). Instead of the virtual edges/Gabriel Graph construction, we can define virtual nodes at all intersections of two edges. These virtual nodes are managed by the end-points of the intersecting edges; sending a message from or to a virtual node means sending a message from or to a neighbor (not virtual) of the virtual node. If this is applied on G'_{BG}, we obtain a planar (by definition!) graph with only $O(A)$ nodes in any given area A. Because this planar graph is a spanner, we get a geometric routing algorithm with cost $O(c^2)$ by applying AFR, GOAFR, or GOAFR+ [20, 21, 22].

9. CONCLUSION

What is the benefit of oversimplified models about which interesting properties can be proven, that have however barely anything in common with reality? But what if we adjust our model to imitate reality to the least detail and obtain nothing but a system far too complex for stringent reasoning? These are the two extremes for which we study a potential way out in the field of ad-hoc network modeling: A model capturing the essence of ad-hoc networks, yet concise enough to permit stringent theoretical results. For the Quasi Unit Disk Graph model—having edges between all nodes with distance at most d, d lying between 0 and 1, and no edge of length greater than 1—we constructed a message complexity lower bound for any volatile memory routing algorithm. We showed that a flooding algorithm matches this lower bound and is consequently asymptotically optimal with respect to message complexity. We described a geometric routing algorithm combining greedy routing and geometric flooding, resulting in a message-optimal algorithm in the worst case and message-efficient algorithm in the average case. We finally showed that classic geometric routing algorithms can be employed with the same performance guarantees as for Unit Disk Graphs if d is at least $1/\sqrt{2}$.

10. REFERENCES

[1] K. Alzoubi, P.-J. Wan, and O. Frieder. Message-Optimal Connected Dominating Sets in Mobile Ad Hoc Networks. In *Proc. of the 3rd ACM Int. Symposium on Mobile ad hoc networking & computing (MOBIHOC)*, 2002.

[2] B. Awerbuch. Complexity of Network Synchronization. *Journal of the ACM (JACM)*, 32(4):804–823, 1985.

[3] B. Awerbuch and D. Peleg. Network synchronization with polylogarithmic overhead. In *Proc. of the 31st IEEE Symp. on Fouccations of Computer Science (FOCS)*, pages 514–522, 1990.

[4] L. Barrière, P. Fraigniaud, and L. Narayanan. Robust Position-Based Routing in Wireless Ad Hoc Networks with Unstable Transmission Ranges. In *Proc. of the 5th International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications (DIAL-M)*, pages 19–27. ACM Press, 2001.

[5] S. Basagni, I. Chlamtac, V.R. Syrotiuk, and B.A. Woodward. A Distance Routing Effect Algorithm for Mobility (DREAM). In *Proc. of the 4th Annual International Conference on Mobile Computing and Networking (MOBICOM)*, pages 76–84. ACM Press, 1998.

[6] P. Bose, P. Morin, I. Stojmenovic, and J. Urrutia. Routing with Guaranteed Delivery in ad hoc Wireless Networks. In *Proc. of the 3rd International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications (DIAL-M)*, pages 48–55, 1999.

[7] E. Chang. Echo Algorithms: Depth Parallel Operations on General Graphs. *IEEE Transactions on Software Engineering*, pages 391–401, 1982.

[8] G.G. Finn. Routing and addressing problems in large metropolitan-scale internetworks. Technical Report ISI/RR-87-180, USC/ISI, March 1987.

[9] K.R. Gabriel and R.R. Sokal. A New Statistical Approach to Geographic Variation Analysis. *Systematic Zoology*, 18:259–278, 1969.

[10] J. Gao, L. Guibas, J. Hershberger, L. Zhang, and A. Zhu. Discrete Mobile Centers. In *Proc. 17^{th} Annual Symposium on Computational Geometry (SCG)*, pages 188–196. ACM Press, 2001.

[11] J. Gao, L. Guibas, J. Hershberger, L. Zhang, and A. Zhu. Geometric Spanner for Routing in Mobile Networks. In *Proc. of the 2^{nd} ACM International Symposium on Mobile Ad Hoc Networking & Computing*, pages 45–55. ACM Press, 2001.

[12] T.C. Hou and V.O.K. Li. Transmission range control in multihop packet radio networks. *IEEE Transactions on Communications*, 34(1):38–44, 1986.

[13] H. Huang, A. W. Richa, and M. Segal. Approximation Algorithms for the Mobile Piercing Set Problem with Applications to Clustering in Ad-hoc Networks. In *Proc. of the 6^{th} International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications (DIAL-M)*, pages 52–61, 2002.

[14] L. Jia, R. Rajaraman, and R. Suel. An Efficient Distributed Algorithm for Constructing Small Dominating Sets. In *Proc. of the 20^{th} ACM Symposium on Principles of Distributed Computing (PODC)*, pages 33–42, 2001.

[15] David B Johnson and David A Maltz. Dynamic source routing in ad hoc wireless networks. In Imielinski and Korth, editors, *Mobile Computing*, volume 353. Kluwer Academic Publishers, 1996.

[16] B. Karp and H.T. Kung. GPSR: Greedy Perimeter Stateless Routing for Wireless Networks. In *Proc. of the 6^{th} Annual International Conference on Mobile Computing and Networking (MOBICOM)*, pages 243–254, 2000.

[17] Y.-B. Ko and N.H. Vaidya. Location-Aided Routing (LAR) in Mobile Ad Hoc Networks. In *Proc. of the 4^{th} Annual International Conference on Mobile Computing and Networking (MOBICOM)*, pages 66–75, 1998.

[18] E. Kranakis, H. Singh, and J. Urrutia. Compass Routing on Geometric Networks. In *Proc. 11^{th} Canadian Conference on Computational Geometry*, pages 51–54, Vancouver, August 1999.

[19] F. Kuhn and R. Wattenhofer. Constant-Time Distributed Dominating Set Approximation. In *Proc. of the 22^{nd} ACM Symposium on the Principles of Distributed Computing (PODC)*, 2003.

[20] F. Kuhn, R. Wattenhofer, Y. Zhang, and A. Zollinger. Geometric Routing: Of Theory and Practice. In *Proc. of the 22^{nd} ACM Symposium on the Principles of Distributed Computing (PODC)*, 2003.

[21] F. Kuhn, R. Wattenhofer, and A. Zollinger. Asymptotically Optimal Geometric Mobile Ad-Hoc Routing. In *Proc. of the 6^{th} International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications (DIAL-M)*, pages 24–33. ACM Press, 2002.

[22] F. Kuhn, R. Wattenhofer, and A. Zollinger. Worst-Case Optimal and Average-Case Efficient Geometric Ad-Hoc Routing. In *Proc. of the 4^{th} ACM Int. Symposium on Mobile Ad-Hoc Networking and Computing (MOBIHOC)*, 2003.

[23] D. Peleg. *Distributed Computing, A Locality-Sensitive Approach*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2000.

[24] D. Peleg and A. A. Schäffer. Graph Spanners. *Journal of Graph Theory*, 13(1):99–116, 1989.

[25] C. Perkins and E. Royer. Ad-Hoc On-Demand Distance Vector Routing. In *Proc. of the 2^{nd} IEEE Workshop on Mobile Computing Systems and Applications*, pages 90–100, 1999.

[26] R. Rajaraman. Topology Control and Routing in Ad hoc Networks: A Survey. *SIGACT News*, 33:60–73, June 2002.

[27] H. Takagi and L. Kleinrock. Optimal transmission ranges for randomly distributed packet radio terminals. *IEEE Transactions on Communications*, 32(3):246–257, 1984.

[28] G. Toussaint. The Relative Neighborhood Graph of a Finite Planar Set. *Pattern Recognition*, 12(4):261–268, 1980.

[29] Y. Wang and X.-Y. Li. Geometric Spanners for Wireless Ad Hoc Networks. In *Proc. of the 22^{nd} International Conference on Distributed Computing Systems (ICDCS)*, 2002.

[30] R. Wattenhofer, L. Li, P. Bahl, and Y.-M. Wang. Distributed Topology Control for Power Efficient Operation in Multihop Wireless Ad Hoc Networks. In *Proc. of the 20^{th} Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, pages 1388–1397, 2001.