Traffic and Hop Efficient Position-based Routing using a Cell Structure*

Stefan Rührup† Christian Schindelhauer‡
Heinz Nixdorf Institute
University of Paderborn, Germany

Abstract

We investigate point-to-point routing in mobile ad hoc networks based on position information with local network knowledge. We show that such routing cannot be efficiently performed by reactive communication protocols, i.e. delivering a packet needs $\Omega(n)$ traffic for n mobile hosts. Therefore we consider k-hop-proactive protocols, where network information may be transmitted to a k-hop-neighborhood.

Known position-based routing algorithms combine greedy forwarding with a recovery strategy to circumvent regions without mobile hosts. If a region cannot be bridged because of the restricted transmission radius, we call it a barrier. The efficiency of position-based routing algorithms depends on these barriers. To formalize this measure we use a grid with cell size linear in the transmission radius and count the border cells m of relevant barriers. We show that this cell structure can easily be established. It can also be used by a recovery strategy, because it implicitly provides a planarization of the network graph, that is needed to prevent cyclic routing.

We analyze position-based routing algorithms with respect to the size of barriers m and the length of the shortest path between source and destination h and compare single- and multi-message strategies. We show that a multiple message strategy consisting of greedy forwarding and greedy recovery has dilation $\mathcal{O}(h+m)$ and needs $\mathcal{O}(h^3)$ messages using expanding ring search based on a hop-counter. If this mechanism is replaced by expanding ellipses as boundary we observe dilation $\mathcal{O}((h+m)\log h)$ and traffic $\mathcal{O}(\min\{m \cdot h \log h, h^2\})$. The corresponding single message strategy using greedy forwarding and guideline recovery has dilation and traffic $\mathcal{O}\left(h + \left(1 + \log \frac{h}{\sqrt{m}}\right) m\right)$.

A fast straightforward multiple message solution is flooding with dilation $\mathcal{O}(h)$ and $\mathcal{O}(h^2)$ messages. As a highlight we present a multiple massage routing algorithm, called Chessboard routing, with dilation $\mathcal{O}(h)$ and and traffic $\mathcal{O}(\min\{h^2, h^{\frac{3}{2}} + m\sqrt{h}\})$. If the number of barriers is known traffic reduces to $\mathcal{O}(\sqrt{m}h)$ with dilation $\mathcal{O}(h)$.

At last we prove a lower bound for the dilation of $\Omega(\min\{m, h^2\})$ for single message $\mathcal{O}(1)$-hop-proactive routing protocols and a lower bound for the traffic of $\Omega(\min\{m, h^2\})$ for multi-message $\mathcal{O}(1)$-hop-proactive routing protocols. All analyses of the algorithms except the Chessboard routing are tight up to constant factors.

*Supported by the DFG Sonderforschungsbereich 376: "Massive Parallelität: Algorithmen, Entwurfsmethoden, Anwendungen." Partially supported by the EU within the 6th Framework Programme under contract 001907 "Dynamically Evolving, Large Scale Information Systems" (DELIS)

†DFG Graduiertenkolleg 776 "Automatische Konfigurierung in offenen Systemen", sr@uni-paderborn.de.

‡Institute of Computer Science, schindel@uni-paderborn.de.

1 Introduction

A mobile ad hoc network is a communication network of mobile hosts without a dedicated infrastructure. Mobile hosts act as clients and as routers and self-configure the network. Often the resources of the mobile hosts are restricted, especially the computational power and energy available for radio transmission. This limits the transmission range of the radio signals and multiple hops are needed to deliver messages. Nearby message transmission causes conflicts because of radio interferences. So, a communication protocol has to take all these problems into account while it must be kept simple because of the restricted resources.

For some applications participants of a mobile ad hoc network are equipped with a positioning system. The position information is a valuable aid for solving a routing problem. In such case it is not necessary to exchange long routing tables or to swamp the network with control messages indicating possible routes. It suffices to route a messages in the rough direction of the target and in the case of obstacles some recovery strategy has to be applied. This method of message delivery is called position-based routing. Clearly, such a scheme can easily cope with network changes and the mobility of participants.

An applicable situation is if swarms of robots explore an unknown terrain and gather environmental data. Then, it may suffice to address robots by a position instead of a network address, i.e., if a certain job has to be carried out in a specific position, this job is assigned to the robot that is nearest to this position. The order can be forwarded to this position by position-based routing. A robot nearby chooses to perform the task. If it is necessary to communicate with a specific node using position-based routing, however, a location service is needed to translate the node's address into a geographic position.

There are approaches for complete reactive position-based routing protocols [5, 7]. Here, all communication is demand-driven. We will see that in some worst-case scenario this implies high and bursty communication overhead. A better solution is to periodically inform the local neighborhood about the network situation using beacon signals. We formalize this approach by so-called k-hop proactive routing protocols. We show that using a 2-hop-proactive protocol networks can be described by a plain cell structure forming a grid.

Still, problems arise when many deserted regions form a maze misleading all greedy approaches for message forwarding. For this situation Kuhn et al. [14] have proven tight upper and lower bounds for the number of necessary messages. We do not think that such mazes occur in practice. Therfore we describe the complexity of the network scenarios by the number of cells, i.e. border cells, neighbored to deserted cells obstructing communication. According to this measure position-based routing protocols can be characterized more accurately for real-world applications.

1.1 Background and Related Work

There are many different approaches to solve routing problems based on position information. Basically a framework for position-based routing consists of a *location service*, that is used by a source node to obtain the position of the destination node, and a *forwarding strategy*, that determines to which neighbor a packet has to be passed by intermediate nodes. A survey of these approaches can be found in [15, 6, 17].

The problem of locating the destination node is somehow similar to the route discovery problem in ad hoc network protocols. *Proactive* schemes disseminate routing information before it is needed, whereas *reactive* schemes start the route discovery on demand. Accordingly, a location service may work in a proactive manner (like proposed in [2]): Each node maintains a location database and distributes its own position throughout the network at regular intervals. To reduce the update cost, such location databases can also be maintained by a small number of dedicated nodes, that act as location servers. The location-aided routing (LAR) protocols by Ko and Vaidya [11] work completely reactive. Position information is only used, if it is available. If the destination's position is unknown, the network is flooded with route requests.

The forwarding strategy is a distinctive feature of position-based routing algorithms. There are essentially three strategies: greedy forwarding, face routing and restricted directional flooding.

Greedy forwarding is based on a locally optimal decision of a node to pass a packet to a neighbored node that is nearer to the destination. Local decision strategies are e.g. MFR (most forward within the transmission radius) [19], which tries to minimize the number of hops, NFP (nearest with forwarding progress) [8], which tries to minimize energy consumption and compass routing [12], that tries to minimize the path length by choosing the node with the minimal angular distance to the direction of the destination. Since

a packet could be stuck in a local maximum, greedy forwarding needs a recovery strategy like perimeter routing [10], where a packet is passed along the edges of the face incident to the dead end node until a forwarding progress according to the greedy strategy can be made. Therefore the network topology must be a planar embedding of a graph (e.g. the Delaunay triangulation or the relative neighborhood graph [9]) that can be constructed with the aid of the position information of neighbored nodes.

Face routing [12] (see also [4]) is based on passing the packets around the faces of a planar network topology. First a line segment between the source and the destination is considered and the face that is intersected by this segment is chosen first. Then, the idea is to traverse the edges on the boundary of the face in order to ascertain all the edges that are intersected by the line segment. The edge with the intersection point that is nearest to the destination determines the face that is traversed next. This procedure is reiterated until the destination is reached. This algorithm visits at most $3|E|$ edges. As packet delivery is guaranteed, face routing can also be used as a recovery strategy in combination with greedy forwarding. A variant of face routing, called perimeter routing, is proposed by Karp and Kung [10] as recovery strategy. Bose et al. [4] propose an algorithm, that is very similar to perimeter routing, called face-2 routing, that visits $\Omega(n^2)$ edges in worst cases, but performs well in simulations. A further variant is described by Kuhn et al. [14] (called OFR), where the complete interior of a face is explored in order to determine the node that minimizes the distance to the destination. Then, at this node the next (adjacent) face toward the destination is chosen. The authors also give an overview of methods to combine greedy and face routing. In [13] Kuhn et al. propose an adaptive face routing (AFR). The idea is to bound the faces by an elliptic region to prevent detours. If no route to the destination within this region can be found, the region is successively enlarged. The algorithm yields paths with cost $O(h^2)$, where h denotes the cost of the optimal route, measured by the number of hops, distance or energy consumption. The authors apply a so called $\Omega(1)$-model that bounds the minimal distance between nodes in the network and show a lower bound of $\Omega(h^2)$.

Restricted directional flooding [2] is a method to deal with outdated position information. From the last known position of the destination node and its maximum velocity a destination region is obtained. Now, the sender transmits a packet to all nodes that lie in the direction of this region in order to flood the whole area in which the destination node lies. This strategy can be used to reduce the overhead of flooding used by reactive ad hoc routing protocols in the route discovery phase (location-aided routing, see [11]).

Besides these strategies there are hierarchical approaches that combine geometric forwarding with other ad hoc routing protocols, e.g. in the terminodes project [3] a position-based routing is used to forward packets over long distances, whereas a proactive distance routing is responsible for the local level.

1.2 The Model

There are n mobile hosts in the plane. We require that each mobile host knows its absolute position coordinates. Each mobile host can communicate wirelessly with a fixed transmission range r in a synchronized model, with one message per step. Only if exactly one message is disseminated from a node in distance at most r, then this messages can be received. For multi-hop communication, we require that the source knows also the geographical position of the destination. We assume that data transmission is much faster compared to the movement of these hosts, i.e. during a packet delivery the positions of the mobile hosts are fixed.

Definition 1 *The* position-based routing problem *is to find a route from a source to a destination, only with the aid of position information of the mobile hosts and the location of the destination.*

We concentrate on point-to-point routing (unicast). Our main concern is the number of messages triggered in the attempt of sending the packet from source to destination, i.e. the *traffic* and the *dilation* of the message on its way from source to destination.

Definition 2 *The* traffic *induced by delivering a packet is the number of radio transmissions of this packet, its duplicates, and corresponding control messages during the attempt of delivering the packet.*

Definition 3 (Stojmenovic and Lin [18]) *The* flooding rate *is the ratio of the number of message transmissions (traffic) and the shortest possible hop count between source and destination.*

1.3 Outline of the paper

In Section 2 we see that a reactive network needs linear traffic to deliver a message even in a two-hop neighborhood. Hence, it makes sense to allow some proactiveness, which we model by the notion of k-hop-neighborhood. Then, the traffic is divided into a proactive portion, which in this paper is linear in the number of nodes and a reactive portion using proactively distributed information from a k-hop-neighborhood.

In Section 3 we show that a 2-hop-proactive protocol can establish a grid-like cell structure with designated gateway nodes. This cell structure regarding dilation and traffic is equivalent to the original setting up to small constant factors. This way routing reduces to a neat communication problem of cellular automata of unknown structure: Given a grid of an arbitrary combination of "good cells" and "bad cells" the task is to find the shortest path from the source cell to the destination cell using only good cells and the local knowledge of which neighbored cells are good or bad. Good cells are *node* or *link cells* that allow communications, while bad cells are deserted cells, called *barrier cells*, obstructing the communication because their lack of relay stations.

An important observation is that the number of reachable barrier cells m gives us a measure for the complexity of the scenario. E.g., if all participants are closely gathered in a square or circle then $m = \mathcal{O}(n)$, if in each cell a constant number of nodes reside. This network construction is also compatible to the reactive ad hoc network of Kuhn, Wattenhofer and Zollinger [13] which allows only a constant number of nodes in the transmission radius of a node. In their paper, tight bounds have been proven, which also apply here. Their lower bounds can be transformed to a complex scenario with $m = \Theta(n^2)$. We discuss here the question, whether routing in less complex scenarios can be performed more efficiently.

In Section 3 we discuss a variety of position-based routing algorithms using this cell structure. We combine greedy forwarding using only a single message or multiple messages with different recovery strategies and termination mechanisms. All these strategies use expanding ring search. As recovery strategies for the case when greedy forwarding collides with a barrier we first consider a greedy recovery strategy. Here messages follow the obstructing barriers until a point at the barrier is reached which is closer than the collision cell. Combined with a hop-counter termination mechanism for the expanding ring search this algorithm is called G2-TTL and has dilation $\mathcal{O}(h + m)$ and traffic $\mathcal{O}(h^3)$. Less traffic occurs if we replace the termination mechanism by an ellipse construction defining the G2-Ell routing. Here dilation is $\mathcal{O}((h + m) \log h)$ and traffic $\mathcal{O}(\min\{m \cdot h \log h, h^2\})$ messages. These multi message algorithm are outperformed by a single-message algorithm where the recovery mode is left if the direct line between source and target is rediscovered on the other side of the barrier, called G-Guide with dilation and traffic $h + (1 + \log \frac{h}{\sqrt{m}})m$.

While the traffic of G-Guide is close to optimal all these greedy routing algorithms are rather slow with respect to the number of barrier cells and compared with a simple Flooding algorithm with dilation $\mathcal{O}(h)$ and traffic $\mathcal{O}(h^2)$. The Chessboard routing algorithm has traffic $\mathcal{O}(\min\{h^2, h^{2-\alpha} + mh^{\alpha}\})$ and dilation $\mathcal{O}(h)$ for an $\alpha \in [0, 1]$. If the number of barriers is known, this algorithm performs with traffic $\mathcal{O}(\sqrt{m}h)$. All analyses of the algorithms except Chessboard routing are tight up to constant factors. Further note that for these algorithms m refers only the number of reachable barrier nodes in some $\mathcal{O}(h)$ vicinity. Hence, m can vary from 0 to $\mathcal{O}(h^2)$. If we sometimes also refer to m as the border cells, note that this number and the number of reachable barrier cells are linearly correlated.

At last we present general lower bounds for relying on $\mathcal{O}(1)$-proactive position-based routing algorithms. Single message protocols need dilation and traffic $\Omega(m + h)$, which is also a lower bound on the traffic for multiple message protocols. For multiple message routing algorithms only the trivial lower dilation bound of $\Omega(h)$ is known. Table 1 summarizes the results presented in this paper.

2 Proactive versus Reactive Routing

Proactive routing protocols are often referred to as table-driven protocols, whereas reactive routing is also called on-demand routing. In position-based routing proactive communication is often used to exchange position information among neighbored nodes. This motivates us to define an intermediate notion of k-hop-proactive routing, where information is only spread locally under closely neighbored nodes. This leads to quick adaption of network changes and reduces the traffic of the proactive part of the protocol.

Name	M/S message(s)	Dilation	Traffic
G2-TTL	Multiple	$\mathcal{O}(h+m)$	$\mathcal{O}(h^3)$
G2-Ell	Multiple	$\mathcal{O}((h+m)\log h)$	$\mathcal{O}(\min\{m \cdot h \log h, h^2\})$
G-Guide	Single	$\mathcal{O}(h+m(1+\log \frac{h^2}{m}))$	$\mathcal{O}(h+m(1+\log \frac{h^2}{m}))$
Flooding	Multiple	$\mathcal{O}(h)$	$\mathcal{O}(h^2)$
Chessboard	Multiple	$\mathcal{O}(h)$	$\mathcal{O}(\sqrt{m}h)$
Lower bound	Multiple	$\Omega(h)$	$\Omega(h+m)$
Lower bound	Single	$\Omega(h+m)$	$\Omega(h+m)$

Table 1: Upper and lower bounds for the position-based routing strategies with minimal hop-distance h and barriers with m border cells.

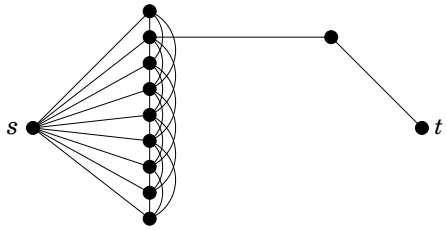

Figure 1: Worst-case example for reactive routing

Definition 4 *If a communication pattern includes a periodic exchange of control messages (even if there is no communication demand), then it is called* proactive*. If communication is triggered only by demand, we call this communication* reactive.

A communication protocol is k-hop proactive*, if proactively transmitted information of a node are available only in nodes in a hop-distance of* k*.*

The main advantage of reactive routing is the lack of communication overhead during demandless time phases. The following theorem shows the high price to be payed for this along the ideas presented in [1].

Theorem 1 *Every reactive position-based routing protocol (i.e. 0-hop proactive) has a flooding rate of* $\Omega(n)$ *in the worst case.*

Proof: In Figure 1 we depict a case where the source S can reach $n-3$ nodes directly. But only one of them is on the shortest path of length 3 from S to the destination node D. In the first round S can inform all nodes with one message except two nodes. Then, only one of them can forward it on the right path toward the destination. Which one is not known to the network and cannot be determined by the use of the destination coordinates. Therefore a deterministic algorithm can be forced to try $n-3$ attempts until succeeding by a fooling argument.

For a probabilistic algorithm we choose the first node randomly from the neighbors of S. If the attempt fails, we retry and choose one of the remaining neighbors. Hence the expected number of attempts to succeed is $n/2-1$. Thus the necessary messages to deliver the packet is at least $n/2+1$. ■

This theorem shows that at least minimum proactiveness is necessary for efficient delivery of packets. In the next subsection we show a technique for 1-hop proactive routing that solves this problem in constant (worst-case) time using a grid cell structure.

If 1-hop-proactive protocols broadcast only position information at least n messages occur. However, in a static model this price needs to be payed only once and in a dynamic setting only from time to time, i.e. with an adequate update frequency.

Theorem 2 *There is a* 1*-hop proactive routing protocol that delivers all packets within hop-distance* 3 *with expected constant traffic.*

Proof Sketch: In the proactive part every node broadcasts its position to all the nodes in its r-neighborhood.

For the demand-driven, reactive communication part, the source node broadcasts a question to its direct neighbors, whether any node knows a node in the r-neighborhood of the destination.

Every node who knows such a node will answer with probability $1/k$ if k is the number of nodes which reside it its r-neighborhood. This way, the expected number of answers is constant.

As soon as the source receives this answer, it signals that it has received this information and asks one of the responding nodes to forward the message. ■

Although traffic is small, large dilation may occur. E.g. consider a clique of size n and a single node connected to only one node of a clique. The probability that the single node receives an answer is one step is $\frac{1}{n}$, thus the expected dilation is linear. By allowing one more hop in the proactive part, we can solve this problem in constant dilation and traffic. We will show in the following section how this can be achieved.

3 The Cell Structure

One of the main ideas in this paper is to use the position information to define a global grid. The length of a grid square ℓ is a constant fraction of the transmission range r. We will call such grid squares *cells*. Each mobile host uses its positioning capabilities to determine to which cell it belongs. It can also assign its neighbors to the cells within its transmission range, based on their position coordinates. Thus, we can abstract from the view of a mobile host's environment by classifying the cells. In Section 3.1 we will see that this classification leads to a formal description of barriers.

The subdivision of the plane into cells can also be used to define a local communication schedule. If there are more than one mobile host located in on cell, we assign a constant number of gateways, which provide the connection to neighbored cells. Therefore, the maximum number of active nodes in distance r to a mobile hosts is constant, which helps us to deal with interferences.

We will pick up this idea to demonstrate, that we can abstract from the geometric properties of the network topology which is defined by the transmission radii of the mobile hosts. In the following we show that we can perform routing on the cell structure with constant overhead.

We will establish the cell structure by a 2-hop-proactive protocol with altogether $O(n)$ messages in each phase.

- Phase A: Each mobile host broadcasts its coordinates.
- Phase B: Each mobile host broadcasts the grid coordinates of cells, from which it has detected a signal in phase A.

Because the cell length is a constant fraction of the transmission radius the possible set of cells, where other nodes can be heard, is constant. So, the communication overhead for the second round is $O(n)$ bits. Because each message broadcasted in phase B contains only the classification of a constant number of cells, the message length remains constant regardless of the size of the neighborhood. Thus, the overhead of this protocol is smaller than of a typical dissemination of information among a 2-hop neighborhood.

We rely on the assumption that all mobile hosts in a cell are connected directly. So it is sufficient to find gateways for each of the cells. The maximum number of cells within the transmission range r grows quadratically by $\frac{r}{\ell}$. This is a constant, since we choose ℓ linear in r.

We will assign explicit and implicit gateway nodes. Explicit refers to a situation where a gateway node is informed about its role by the proactive protocol. Explicit gateway nodes serve as senders of messages. Messages will be sent from explicit, outgoing gateways to incoming gateway nodes, which learn their role as incoming gateway by the header information attached at the message, i.e. by the reactive part of the routing protocol. These implicit gateways will forward the message to another explicit gateway for further delivery, or directly to the destination, if it resides in the cell.

Lemma 1 *There exists a 2-hop-proactive protocol that explicitly assigns for each cell $O(1)$ outgoing gateway nodes and implicitly assigns for each cell $O(1)$ incoming gateway nodes.*

Proof: The actual algorithm starts when all (proactive) communication is completed. Then, all nodes in a cell know all their neighbors' location data. All nodes perform the following routine process in parallel without any further communication.

We use an ordering of the nodes in a cell based on their coordinates. For this, one can sort the nodes according to the coordinates (first horizontal, then vertical).

At the beginning we start with the union of all cells C that can be reached by any node of this cell and the set S of all nodes in a cell. We assign the first node as an explicit gate way to all its neighbored cells and eliminate all these cells from C and eliminate the first node from S. We repeat this until the cell set C is empty. ■

Once the grid structure based on the gateways is established, we can shift our focus from a rather geometrical, i.e. Euclidean, point of view to a discrete description. The problem of routing information through the network is reduced to shift information from one grid cell to neighbored grid cells. The neighborhood information is available at the gateways. In the next sub-section we see that we already have some local information of communication barriers.

3.1 Barriers, Cells and Routes

Every mobile host knows its absolute position within the plane given by x and y-coordinates. We choose a cell structure based on a grid subdivision with edge length ℓ of at most $\sqrt{3/20}\ r \approx 0.39r$, where r is the guaranteed maximum transmission range of all mobile hosts. We introduce three classifications for such a cell, see Figure 2:

1. A cell is a *node cell* if at least one mobile host is inside the cell.
2. A cell is a *link cell* if it does not contain any mobile host and if it is intersected by the line between two mobile hosts with distance at most r; or if all points of the cells have distance at most $r/2$ to a mobile host.
3. Every other cell is called a *barrier cell*.

Definition 5 *A* cell barrier *is a set of barrier cells, which are connected orthogonally or diagonally. The* border cells *of a barrier is the set of barrier cells having cell or link nodes as orthogonal or diagonal neighbors. The* cell barrier circumference *is the number of border cells.*

Note that all border cells of a barrier are connected orthogonally.

Definition 6 *A* **cell-based path** *is path $(u_1, \ldots, u_m)$ in a unit-disk-graph with the following properties:*

1. *$|u_i, u_{i+i}| \leq r$, for all $i \in [m-1]$.*
2. *$u_2, \ldots, u_{m-1}$ are gateway nodes, where u_{2i} are explicit gateways and u_{2i+1} are implicit gateways.*
3. *The cells of u_{2i-1} and u_{2i}, for $i \in \{1, \ldots, m/2\}$ are gateways of the same or orthogonally neighbored cells.*

A **cell route** *$(c_1, \ldots, c_m)$ consists of node or edge cells, such that c_i and c_{i+1} describe the same cell or orthogonally neighbored cells.*

We call an edge the implicant for a link cell if it intersects with the link cell. Similarly we call a node cell the implicant of a link cell, if all points in the link cell have maximum distance $r/2$ to this point. We show that paths in the network and cell routes are essentially equivalent. The crucial point for this equivalence is the *connectivity property* of a link cell C. All causing nodes or edges of a link cell are directly connected, i.e. if two edges are implicants for the link cell, then there is a direct connection between at least one node of each edge; for an edge and a node as implicants, the node is connected to at least one of the nodes of the edge; for two nodes as implicants these are directly connected.

Lemma 2 *If $\ell \leq \sqrt{3/8}\ r$ then all link cells fulfill the connectivity property.*

Proof: In the case of two nodes causing the link cell, this follows by the triangle inequality, since all points of the link cell have maximum distance $r/2$ to the nodes. In the case of an node and an edge as a implicant, there is a point in the link cell, which has at most distance $r/2$ to one of the nodes of the edge and the connectivity property follows.

If the two causing edges of the link cell cross each other in the link cell. Then this crossing point has at most distance $r/2$ to one of the nodes each. Again the triangle inequality proves the claim.

If the link cell was originated by two link cells without an intersection in the cell, then we have two lines of maximum length r where two points on the lines exist with maximum distance $\sqrt{2}\ell$. Given two lines with these properties the question is how far can be the minimum distance of pairs of nodes of different edges. The minimum distances between points of two different edges occurs for at least one end point. Then, the worst case is that this end point form a isosceles triangle with the other edge as base line and height $\sqrt{2}\ell$. This leads to the maximum distance $\sqrt{2\ell^2 + r^2/4} = r$. ■

Theorem 3 *If $\ell \leq \sqrt{3/20}\ r$ paths in the network and cell routes in the cell structure are equivalent up to a constant factor.*

1. *Every path p of the network can be replaced with cell-based path p' of length at most $2|p| + 1$, where the cell route of p' is a superset of the cell route of p.*
2. *For every cell route R there exists a cell-based path of length at most $2|R|$.*

Proof:

1. We substitute the path $p = (u_1, \ldots, u_n)$ as follows. For each edge $e_i := (u_i, u_{i+1})$ we use the explicit gateway node g_i in the cell of u_i responsible for the cell of u_{i+1} and the implicit gateway w_i in the cell u_{i+1} responsible for the cell of u_i. The cell base path is now $(u_1, g_1, w_1, g_2, w_2, \ldots, g_n, w_n, u_n)$. Since nodes in the same cell are connected this proves the first part of the claim.
2. We consider all three possible transitions between orthogonally neighbored cells:
   - Node cell to node cell: Because $\ell \leq \frac{1}{\sqrt{5}}r$ event the farthest nodes of the cells are connected.
   - Node cell to link cell: If the link cell was implied by a node u then we consider the closest point of the link cell to a node v in the node cell. If $\ell \leq \frac{1}{2}r$ then the distance from u to v is at most r using triangle inequality.
     If the link cell was implied by an edge we consider a point of the line inside the node cell. The maximum distance to a node in the node cell is $\sqrt{5}\ell$. Hence, the worst case is a isosceles triangle with the edge as base line of length r and height $\sqrt{5}\ell$. Since $\ell \leq \sqrt{\frac{3}{20}}\ r$ at least one of the end points of the causing edge is in the transmission range.
   - Edge cell to edge cell: For $\ell \leq \sqrt{\frac{3}{8}}\ r$ this follows from the link connectivity property.

■

Remember that barrier cells are deserted cells which obstruct routing. One may extend this notation also to barriers in the original meaning like walls obstructing radio signals. Our concept of barrier cells is compatible to this notion as long as the connectivity property of link and node cells is satisfied. In Figure 3 an example is given where this property is violated by barriers obstructing radio signals. In such a case the communication graph is not planar anymore and our techniques cannot be applied.

4 Routing Algorithms using the Cell Structure

Based on this cell structure we will analyze existing approaches and compare to our new algorithm. Thereby we will concentrate on the measures hop-distance and traffic. Because of Theorem 3 cell routes can replace paths if we use a 2-hop-proactive protocol. Hence, we assume that routing from one cell to

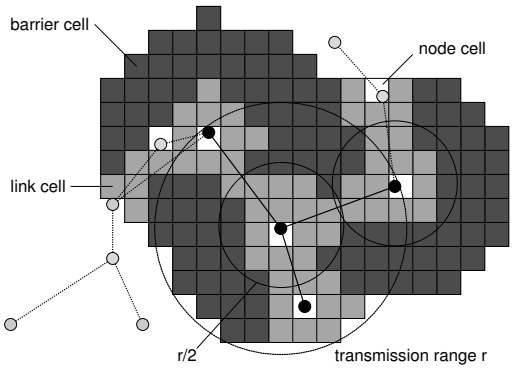

Figure 2: Classification of cells. The node in the center derives the classification of cells within its transmission range from the position information broadcasted by its neighbors. Furthermore, this view is enlarged by the cell classification of the neighbored nodes, which is also broadcasted.

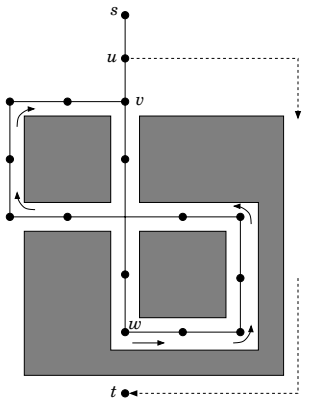

Figure 3: In general the concept of barrier cells cannot be extended to obstacles since the communication graph is not planar anymore and recovery strategies may fail.

an orthogonally neighbored needs one round and do not change the asymptotic behavior of the routing algorithm.

In the following n denotes the number of nodes, s describes the source cell and t the target or destination cell. The minimal hop-distance between s and t counted by node or edge cells is described by h, while h_0 is the minimal distance in L1-norm allowing also barrier cells. At last m describes the number of reachable and relevant border cells, which is linear in the number of reachable and relevant barrier cells.
The *relevant barriers* are the barriers that are placed in a circular area with some radius $\mathcal{O}(h)$ around the source node. The size of a barrier is given by the number of border cells it is enclosed by.

Kuhn, Wattenhofer and Zollinger [13] prove tight bounds for reactive routing, if the number of nodes in each cell is constant (we reformulate the theorem for the cell structure).

Theorem 4 *[13] Assume that in every cell at most a constant number of mobile hosts reside. Then, there exists a reactive routing algorithm, such that the maximum route length is bounded by $\mathcal{O}(\min\{h^2, n\})$ and the number of messages is bounded by $\mathcal{O}(\min\{h^2, n\})$. Furthermore, these bounds are tight.*

The assumption of a bound on the number of nodes in a cell (originally in [13] there is a minimum distance of mobile hosts of $c\,r$ for some constant $c > 0$) allows the use of a completely reactive protocol while avoiding linear overhead in the case that $h \in o(\sqrt{n})$.

There are many approaches for position-based routing [12, 10, 4, 13]. Efficient single-message strategies combine greedy forwarding with a recovery strategy. A time-efficient multi-message strategy is flooding. Flooding actually needs no position information. However, if positions are known, the flooded area can be restricted.

4.1 Greedy Forwarding and Recovery

Greedy forwarding strategies like compass routing [12] or MFR (most forward within radius) [19] can be applied to the cell structure, by forwarding a message to that node cell, that fulfills the local optimization criteria.

As a variant of compass routing, we can also apply the following forwarding rule: Given a source s and target t, we imagine a line from a point in the source cell to the target cell such that this line never crosses a corner point of the grid. This line intersects with cells, that are orthogonally connected. The greedy path $L(s,t)$ is the sequence of all these cells along the line from s to t. A node on this path determines the cell on the path (among the cells that it knows) that is nearest to the destination. If this is not a barrier cell, it forwards the message to the node that is inside or responsible for this cell. Otherwise it initiates a so-called recovery.

Recovery When greedy forwarding fails, a recovery strategy is needed. In recovery mode a message is routed along the border of a barrier according to the so-called *right-hand rule*[1]. Therefor a planarization of the underlying network topology is needed to prevent cycles. This can be achieved by using only the edges that would be present in a relative neighborhood graph or a gabriel graph (see [9]). Unfortunately, this type of planarization means cutting long edges and using only short links, which leads to many small hops in recovery mode.

The cell structure has two advantages: First, no additional planarization strategy is needed, because the border of a barrier is defined by barrier cells. Second, a node knows the structure of the barrier in its immediate environment. So it can take advantage of its whole transmission range and forward packets to nodes with the most forwarding progress in recovery mode.

Progress condition To prevent cyclic routes a message remains in recovery mode until a progress condition is fulfilled. One can choose among the following.

Let g be the cell where the greedy algorithm has transited to the recovery mode. The cell s is the source, t the target of the message. $L(s,t)$ denotes the set of cells on the direct line between s and t. $B(u)$ is the set of barrier cells in the vicinity of a node u.

1. Greedy recovery ("early return")

   Greedy recovery means to resume greedy forwarding whenever a progress is possible. The progress condition is fulfilled if at node u at least one non-barrier cell c exists that is nearer to the destination than cell g according to the L1-norm.

   $$C_{\text{greedy}}(B,c,t) := \ \exists c \notin B(u) \text{ with } ||u,c||_1 \leq 1 \ : \ ||c,t||_1 \leq ||g,t||_1 \ .$$

   This progress condition provides the basis for a multi-message strategy: A message is duplicated when it is switched to recovery mode. Then, a barrier is surrounded from both sides.

2. Guide line ("late return")

   Following the approach in [12] the following condition allows to resume the greedy mode:

   $$C_{\text{guideline}}(g,c,t) := \ \ c \in L(g,t) \ \wedge \ ||c,t||_1 \leq ||g,t||_1 \ .$$

   I.e. a message returns to greedy mode, when it has surrounded a barrier and reached a guide line that connects g and the destination t. We will choose the line between source and destination instead, as described above.

   If a barrier is surrounded by duplicated messages on both sides, this progress condition defines a common point, at which both messages may return to greedy mode.

Search area restriction There are two basic strategies to limit the search area in reactive routing algorithms, where routing paths are not known. E.g., the ad hoc network routing protocol AODV [16] uses flooding to discover a route to the destination. For such cases *expanding ring search* is a well-known technique to restrict the flooding overhead. Expanding ring search uses the time-to-live field (TTL) in the routing header of a packet. Beginning with a short TTL the search for the destination is started. If the destination cannot be reached, the search is repeatedly started with an increased TTL until the destination is found. With regard to efficiency we use expanding ring search with a multiplicative increase, i.e. the TTL value is doubled in each round.

Another strategy uses a *bounding ellipse*. As we know the positions of source and destination determine an area where a path of a length h can extend from source s to destination t. This area is bounded by an ellipse $E = \{x \mid ||s,x||_1 + ||x,t||_1 = h\}$ with source and destination as foci. h equals the length of the major axis. Now we can apply a strategy similar to expanding ring search. First we start with a bounding ellipse that contains only the directed path ($h_0 := ||s,t||_1$), and then we enlarge the ellipse ($h_{i+1} := 2h_i$)

[1] The right-hand rule is a well-known rule for traversing a maze: By keeping one hand (either the left or the right hand) in touch of the wall, it is guaranteed to find a way out of a maze.

until a path to the destination is found. Here, the messages are not deleted after a certain time but the routing paths are bounded by an artificial boundary that is given by the border of the ellipse.

The bounding ellipse is also applied by Kuhn, Wattenhofer and Zollinger [13] to face routing in oder to achieve asymptotic efficiency.

Routing strategies There are several variants to combine a greedy strategy with an appropriate recovery strategy. We concentrate on two strategies that should stand representatively for greedy routing with recovery. We have combined known techniques with respect to asymptotic efficiency.

- *Greedy Forwarding with Greedy Recovery* (G2)

  Multi-message strategy with message duplication in front of barriers; Search area restriction through bounding ellipse (G2-Ell) or TTL (G2-TTL).

- *Greedy Forwarding with Guide Line Recovery* (G-Guide)

  Single-message strategy with depth-first search on the routing graph; Search area restriction through bounding ellipse.

G2 duplicates a message if it is switched to recovery mode. Then, one message is forwarded according to the right-hand rule, the other message, i.e. the duplicate uses the left-hand recovery. If a messages in recovery mode encounters a cell that has been already visited by another message duplicate, then it is deleted. The algorithm uses bounding ellipse or expanding ring search (TTL).

G-Guide uses only the bounding ellipse. In this strategy it may happen, that a packet has to traverse most of the barriers in every step. Therefore we consider a set of paths that are defined by the guide line, the barriers and the boundary of the search area. We call this set of paths *routing graph*. The algorithm performs a depth-first search for the following reason, which can be explained by means of the example shown in Figure 4: At node v the right-hand recovery starts and the recovery path leads back visiting segments of the bounding ellipse that have been visited before. This happens at all the nodes located at a position similar to node v. Turning back earlier to greedy mode violates the progress condition, so it could happen, that the destination is not found. In order to prevent the message from such a detour one could define the visited segments of the guiding line as artificial barrier. But then it is not guaranteed that the destination is reached (The S-shaped barrier near t is a counter-example). Depth-first search on the routing graph (see Figure 5) can handle this case. It requires a list of visited nodes of the routing graph to be appended to the message.

Upper Bounds For the G2 algorithm we first determine, how long the shortest path the algorithm finds can be compared to the overall shortest path.

Lemma 3 *The ratio between the length h of the overall shortest path P and the length ℓ of the shortest path the G2 algorithm finds P' is bounded by $c \cdot h$ for a constant c.*

Proof: The shortest path is included in a circle $C(t, h)$ around the destination t with radius h. The path P' can only be enlarged through barriers. Such a barrier must not partition the circle such that source and destination are separated. Otherwise it would also lengthen the shortest path P. Actually, we have to regard an ellipse, but a weaker condition (circle) suffices in this case. Under a circle in L1-norm we denote the point set

$$C(c, r) := \{x \mid ||x, c||_1 \leq r\} .$$

If barriers are not allowed to partition the circular region, then the algorithm can find a path that is completely inside the circle $C(t, h)$, because barriers are surrounded from both sides. The length of this path is bounded by the area of $C(t, h)$, which is $c \cdot h^2$ for a constant c. ■

Theorem 5 *The G2-TTL algorithm (multi-message) has dilation $\mathcal{O}(h + m)$ and needs $\mathcal{O}(h^3)$ messages, where h is the length of the shortest path between source and destination and m the size of relevant barriers.*

Proof: The paths of all the messages form a tree. A branch ends if a message reaches the destination, comes across another path (then the tardy duplicate is deleted), or if the TTL expires.

Dilation: Assume, we know the appropriate TTL value. Then, the shortest path from source $v_0 := s$ that reaches the destination $v_k := t$ defines the dilation. This path can be partitioned into greedy sections and recovery sections. Consider a node v_i on this path and assume the message is in greedy mode. The message is heading for the destination until it is switched to recovery mode by a node v_i'. Then, it remains in recovery mode until the circle $C(t, ||t, v_i'||_1)$ is entered at some node v_{i+1}. I.e. $||t, v_i'||_1 = ||t, v_{i+1}||_1$ and therefore $\sum_{i=1}^{k} ||v_i, v_i'||_1 = ||s, t||_1 \leq h$. In recovery mode the message is always in contact with the border cells of a barrier. Thus, the recovery sections $P(v_i', v_{i+1})$ of the routing path have a total length of $\sum_{i=1}^{k} |P(v_i', v_{i+1}|) \leq m$. Altogether the path length is bounded by $h + m$.

We have to approximate the appropriate TTL by starting with an initial value $h_0 = ||s, t||_1$ and doubling it until the destination can be reached (TTL $\geq h + m$). Thus, the dilation is $D = \sum_{i=0}^{\log((h+m)/h_0)} 2^i h_0 = (2\frac{h+m}{h_0} - 1)h_0 = \mathcal{O}(h + m)$.

Note that $m \leq h$, i.e. in the worst-case the dilation is h^2 (cf. Lemma 3).

Traffic: If a shortest path of length h exists, it is inside the circle $C(t, h)$. Then the length of the path P' the algorithm finds, is bounded by $\mathcal{O}(h^2)$ (Lemma 3). There may be barriers outside the circle $C(t, h)$ that are traversed by message duplicates. Note that duplicates are only generated inside the circle (when switching to recovery mode) and not outside. Outside the messages remain in recovery mode, because a progress is made only when entering the circle again ($C(t, h)$ defines the progress condition). That has two consequences: First, there is no message duplication outside $C(t, h)$ and therefore the number of messages for each recovery path outside $C(t, h)$ is bounded by the TTL. Each path can have a length of h^2 because the TTL grows up to h^2 (length of P'). Second, the number of paths that leave the circle is bounded by the circumference of $C(t, h)$, which is $\mathcal{O}(h)$.

The number of messages inside $C(t, h)$ is bounded by $\mathcal{O}(h^2)$. Outside the circle, there are at most $\mathcal{O}(h)$ paths with maximum length h^2 each, i.e. a total traffic of $\mathcal{O}(h^3)$ messages: $T = \sum_{i=0}^{\log(h^2/h_0)} h^2 + 2^i h_0 \cdot h = \mathcal{O}(h^2 \log h + h^3)$. ■

Theorem 6 *The G2-Ell algorithm (multi-message with bounding ellipse) has dilation $\mathcal{O}((h + m) \log h)$ and needs $\mathcal{O}(\min\{m \cdot h \log h, h^2\})$ messages, where h is the length of the shortest path between source and destination and m the size of relevant barriers.*

Proof: Dilation: Assume, we know the size of the bounding ellipse and the ellipse contains a path from source to destination. The bounding ellipse is an artificial barrier with additional $\mathcal{O}(h)$ barrier cells (the circumference of the ellipse is linear in the length of the major axis).

If the straight line toward the destination is intersected by a barrier, then from the barrier cell at this intersection point there is a path along the border of the barrier to another border cell that is nearer to the destination. Otherwise the destination would be isolated. In this case we regard the bounding ellipse together with all barriers connected to it as one barrier. The algorithm uses such paths in recovery mode. In the worst case it traverses all the barriers. Additionally there are the greedy sections of the routing path, the total length of which is bounded by h. For a fixed size of the ellipse the dilation is $\mathcal{O}(h + m)$.

In this case the barrier cells can be aligned in one row such that only after enlarging the ellipse $\log h$ times the destination can be reached. I.e. the algorithm performs $\mathcal{O}(h + m)$ steps each in $\log h$ rounds, which results in a dilation of $\mathcal{O}((h + m) \log h)$.

Remark on acknowledgments: After each round the source has to wait for acknowledgments with which it can detect whether a route to the destination was found or the messages were dropped. In the latter case the next round with an enlarged ellipse is started. The acknowledgments use are forwarded along the edges of a tree that is given by the routing paths of the message duplicates. The leaves of this tree are the points where messages are dropped (because they cross the path of another message) or reach the destination. At each branching point, the early acknowledgment of one subtree waits for the tardy acknowledgment of the other subtree. Then only one acknowledgment is forwarded to the next branching point. The additional dilation by this acknowledging process is given by the length of the longest path in the routing tree. It is bounded by the length of the greedy sections h_0, the length of the recovery sections (which are at most m)

and the circumference of the bounding ellipse (which is $\mathcal{O}(h)$). Thus, the acknowledging process increases neither the dilation nor the traffic asymptotically.

Traffic: The routing paths of the message duplicates form a tree. Each path consists of greedy sections and recovery sections. Since a border cell of a barrier is visited by at most one message (message duplicates that cross an already visited recovery path are deleted), we regard a recovery section as one node in the tree. Such a node defines a branch, where a duplicate of a message is created. The greedy sections have a length of at most $h_0 \leq h$ because of the progress condition. Thus, the tree has depth at most h and at most m nodes, i.e. it has a size of at most $h \cdot m$. Additionally there are m messages needed for traversing the borders of all relevant barriers. This amount of messages is needed in each of $\log h$ rounds in which the ellipse is enlarged. Thus, the overall traffic is $\mathcal{O}((h \cdot m + m) \log h)$.

It is also obvious that there are not more than h duplicates that can be generated (duplicates are only created when in greedy mode a barrier is encountered). I.e. if $m > h$ then the barriers are traversed, but they cannot force more than h paths to be created. In that case the traffic can easily bounded by the area of the bounding ellipse. The area of the ellipse in round i is bounded by $c \cdot h_i^2$ for a constant c, where $h_i = 2^i h_0$ is the length of the major axis.

$T = \sum_{i=0}^{\log(h/h_0)} (2^i h_0)^2 = \mathcal{O}(h^2)$ ■

Theorem 7 *The G-Guide algorithm (single message) has dilation and traffic* $h + (1 + \log \frac{h}{\sqrt{m}})m$.

Proof: The algorithm uses greedy paths and recovery paths: A greedy path is a segment on the guide line. A recovery path begins where the guide line is disrupted by a barrier. It leads along the border of a barrier and possibly along the border of the bounding ellipse until reaching the guide line again. We can abstract from this imagination and define a graph $G = (V, E)$ that contains the greedy paths and recovery paths as edges and the crossing points between these paths as nodes:

$u \in V$, if u is a cell on the guide line and adjacent to a barrier cell. $\{u, v\} \in E$, if $u, v \in V$ and there is a greedy path or a recovery path connecting u and v. There are at most 3 edges connecting two nodes (greedy path, left hand or right hand recovery path). An edge weight $|\{u, v\}|$ is given by the number of cells on the path connecting u and v.

The algorithm performs a depth-first search on G and thus visits $\mathcal{O}(\sum_{e \in E} |e|)$ cells in each round. The sum of edge weights is bounded by the length of the guide line h_0, the number of barrier cells m and the circumference C of the bounding ellipse, which is an artificial barrier (C is linear in the length of the major axis). In round i the size of the routing graph $\sum_{e \in E} |e|$ is bounded by $\mathcal{O}(h_i + m)$ where $h_i = 2^i \cdot h_0$ represents the length of the major axis.

The algorithm increases the bounding ellipse until the destination can be reached. If there is a path between source and destination inside the bounding ellipse, then, obviously, it is found by the algorithm. Otherwise the destination would be isolated by barriers. The shortest path between source and destination with length h can be covered by an ellipse with size (length of the major axis) of at most h. Therefore, $\log h$ rounds are sufficient to increase the ellipse such that the destination can be reached.

How can we delay the algorithm with $\mathcal{O}(m)$ barrier cells? Consider $2m$ barrier cells. m barrier cells are used to form a quadratic maze of side length $\sqrt{m}$ such that a path through the maze has length m. The remaining m barrier cells are aligned in a row orthogonal to the guide line, in order to force the bounding ellipse to be enlarged.

It takes $\log \sqrt{m}$ rounds to discover the maze, i.e. after $\log \sqrt{m}$ the maze is included in the ellipse. In each of these rounds the dilation is bounded by the area the ellipse covers. So, for the first $\log \sqrt{m}$ rounds the dilation is $D = \sum_{i=0}^{\log(\sqrt{m}/h_0)} (2^i \cdot h_0)^2 \leq \sum_{i=0}^{\log \sqrt{m}} (2^i)^2 = \mathcal{O}(m)$. Then there are $\log(h/\sqrt{m})$ remaining rounds, the dilation is $c \cdot m$ per round for a constant c, because in each round the maze must be discovered. For these remaining rounds the dilation is $D = \sum_{i=\log \sqrt{m}}^{\log h} c \cdot m + 2^i = \log \frac{h}{\sqrt{m}} \cdot c \cdot m + 2h - 2\sqrt{m}$.

Altogether the dilation is $\mathcal{O}(h + (1 + \log \frac{h}{\sqrt{m}})m)$. This bound holds also for the traffic because this is a single message strategy. ■

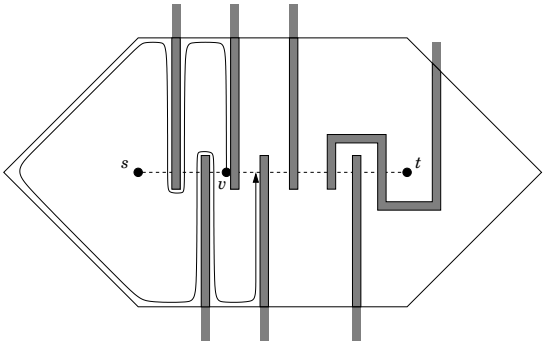

Figure 4: Source s and destination node t are connected by a guiding line that is intersected by some barriers. The search area is bounded by an ellipse (L1-norm). The right-hand recovery initiated at node v leads along barriers and sections of the bounding ellipse that have been visited before.

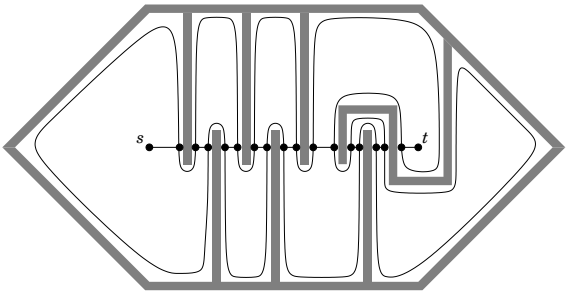

Figure 5: Example for the G-Guide algorithm. The possible paths (greedy and recovery paths) with their interconnection points form the routing graph. The G-Guide algorithm performs a depth-first search on this graph.

4.2 Flooding

Flooding is the fastest way to route a message to the destination; unfortunately it needs the highest number of messages. Often it is unnecessary to flood the whole network. Instead of that, one may use expanding ring search starting with search depth $h_0 := ||s, d||_1$. This way we take advantage of the position information. We bound the flooding area by setting the time-to-live of the message to h_0. Then, we wait for an acknowledgment from the destination. If no acknowledgment arrives, we double the search depth, i.e. in step i we set the time-to-live to $h_i = h_0 \cdot 2^i$. This strategy works also without position information: Then we begin with $h_0 := 1$.

Proposition 1 *Flooding has dilation $\mathcal{O}(h)$ and needs traffic $\mathcal{O}(h^2)$.*

4.3 Chessboard Routing

In this section we present a position-based routing algorithm using the cell structure with the same asymptotic time behavior as flooding but with substantial less traffic if the number m of relevant border cells is small. As a building block of this algorithm we consider the following message multicast problem

Definition 7 *The frame multicast problem is defined for a square of $h \times h$ cells and a set of entry points $s_1, \ldots, s_k$ which are started at certain time points $t_1, \ldots, t_k$. The task is to multicast the message to all cells on the frame which can be reached by paths using only cells of the square.*

Such a routing scheme is called α-competitive if for each cell u on the frame of the $h \times h$ square S a message is delivered to u in at most $\min_{i \in [k]}\{t_i + \alpha\, d_S(s_i, u)\}$ steps, where $d_S(s, u)$ denotes the shortest cell route from s to u using only cells of S.

Of course, by definition the simple flooding algorithm is 1-competitive. The main disadvantage of flooding is the large number of message, namely h^2, regardless whether few or many border cells are involved. The following lemma relies on a simple technique to reduce the traffic with only constant factor slow down.

Lemma 4 *There is a $\mathcal{O}(1)$-competitive frame multicast algorithm that requires traffic $\mathcal{O}(\min\{h^2 \log h, h + m^2 \log h\})$, where m is the number of border cells in the $h \times h$-square.*

Proof: Every entry point s_j coordinates message delivery in a disjoint partition of the square. A cell belongs to a partition of S, if the chronologically first received message stems from s_j.

Each of the entry points starts as soon as they are activated and work in rounds, until the entry point's partition cannot be extended anymore.

If in the preceding round more than $\frac{1}{4}2^i$ border cells were detected, the algorithm skips the first surrounding phase and starts with flooding. In the surrounding phase two messages are sent for 2^i steps along the edges of the square. If barriers intersect with the edge cells of the $h \times h$ square then the messages follows the barrier through the border cells by a left hand or right hand traversal. The whole message transmission stops if an already informed cell is found. Along the two paths all found border cells are counted and eventually after reaching the depth restriction of 2^i, this number is reported to the entry point.

The entry point triggers a flooding algorithm of depth 2^i, where branches of the delivery are cut, if already informed cells are reached or a barrier cell prevents further forwarding. Cells informed by the same entry point in a previous round are seen as uninformed. Furthermore, frame cells which were informed in a surrounding phase of the same or a different entry point are also seen as uninformed, and flooded once more. The message transmission describes a breadth first search (BFS) tree. After 2^i steps the transmission process will be reversed and the number of border cells will be reported to the entry node with the same number of messages as before.

For the timing, note that if in the surrounding part only $\frac{1}{4}2^i$ border cells are found, then all frame cells within distance $\frac{1}{2}2^i$ can be informed. If the algorithm switches to flooding, then of course all frame cells within distance 2^i are reached in this round.

For the time behavior consider an entry point informed at time 0 with distance $d_S(s,u) = d$ to a frame cell u. So, it takes $2 + \log d$ rounds until this frame cell is in the reach of s at the latest. The i-th round needs $4 \cdot 2^i$ rounds for the surrounding and flooding phase. This gives a constant competitive ratio of 16.

For the number of messages let m_i denote the number of border cells found in the i-th round. Now if $m_i \leq \frac{1}{4}2^i$ then the number of messages is bounded by $4 \cdot 2^i$, because only the surrounding part is executed. Otherwise we have at most $4 \cdot 2^i + 2 \cdot \min\{2^{2i}, h^2\}$ messages in this round.

Now we sum over all rounds and receive the following traffic $M_0(h,m)$.

$$\begin{aligned} M_0(h,m) &\leq \sum_{i \in [\log h]:\ m_i \leq \frac{1}{4}2^i} 4 \cdot 2^i + \sum_{i \in [\log h]\ :\ m_i > \frac{1}{4}2^i} 4 \cdot 2^i + 2 \cdot \min\{2^{2i}, h^2\} \\ &\leq 8h + \sum_{i\ :\ m_i > \frac{1}{4}2^i} \min\{2 \cdot 2^{2i}, 2 \cdot h^2\} \\ &\leq 8h + \underbrace{\sum_{i\ :\ \frac{1}{4}2^i \leq m_i \leq h} 2 \cdot 2^{2i}}_{=\mathcal{O}(m^2 \log h)} + \underbrace{\sum_{i\ :\ m_i > h} 2h^2}_{=\mathcal{O}(h^2 \log h)} \end{aligned}$$

If there exists a round where $m_i \geq h$, then after some $\log h$ additional rounds the process stops, since the maximum search depth is limited by h^2. This gives an upper bound of $\mathcal{O}(h^2 \log h)$ messages.

Now assume that in each round we have $m_i \leq h$. This gives a bound of $\mathcal{O}(m^2 \log h + h)$. ■

The number of messages can be reduced by a logarithmic factor if one uses the explored BFS-trees of the previous round. For this, one resends messages into a branch of the tree if there is at least an open leaf left. The following lemma proves, that the number of messages is only linear in the size of the overall-tree.

Given a tree T let T_i denote a sub-graph of tree which contains all paths from the root to the leaves with minimal length i, which are pruned at depth i. This way, the sub-tree T_i has all leaves in depth i. Let $s(T)$ denote the size of the tree, i.e. the number of nodes.

Lemma 5 *For all trees T with depth $d(T)$ it holds $\sum_{i=0}^{\log d(T)} s(T_{2^i}) \ \leq\ 2 \cdot s(T)$.*

Proof: For each node u of a sub-tree T_{2^i} we put a mark on a node of the original tree T. If the depth of u is larger than 2^{i-1}, we put the mark on u. If the depth $d(u)$ is at most 2^i we choose an arbitrary successor in depth $d(u) + 2^{i-1}$. This way every node of T receives at most two marks which implies the claim. ■

Applying this observation to the frame multicast algorithm presented in Lemma 4 we can reduce the traffic by a logarithmic factor.

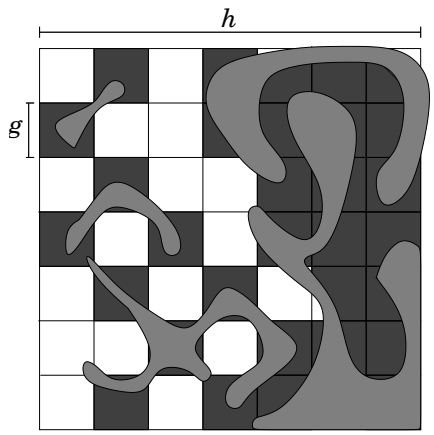

Figure 6: The principle behavior of Chessboard routing relies on a grid where for each sub-square the number of found border cells indicate whether in a sub-square the flooding or the surrounding strategy is applied.

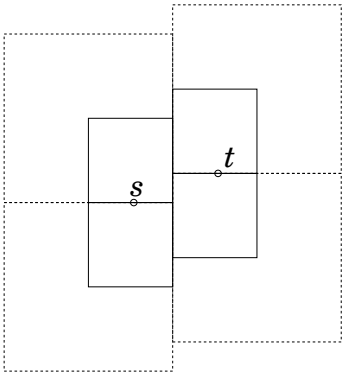

Figure 7: Combining competitive frame multicast algorithms for position-based routing

Lemma 6 *There is a $\mathcal{O}(1)$-competitive frame multicast algorithm that requires traffic $\mathcal{O}(\min\{h^2, h + m^2\})$, where m is the number of border cells in the $h \times h$-square.*

The proof follows from the above considerations and is analogous to the construction used in the following lemma. It will appear in the full paper.

We will now discuss a technique to combine traffic efficient frame multicast algorithms within a grid structure to reduce traffic asymptotically while keeping constant competitive ratio, see Figure 6.

Lemma 7 *For all $g \leq h$ there is a $\mathcal{O}(1)$-competitive frame multicast algorithm with traffic $\mathcal{O}(\min\{h^2, \frac{h^2}{g} + m \cdot g\})$.*

Proof: For readability we do not discuss problems induced by rounding. It can be easily seen, that this causes only a constant factor change. We partition the $h \times h$-square into a $\frac{h}{g} \times \frac{h}{g}$ grid of squares of size $g \times g$.

Every entry point of the $h \times h$-square controls the following algorithm which is somewhat a generalized version of the algorithm in Lemma 4.

Again the multicast process works in rounds. If in the preceding round $\frac{1}{4}2^i$ border cells were detected, then the following phase will be skipped and the algorithm proceeds with Grid-BFS. Two message are sent along the surrounding for at most 2^i steps by a left hand and right hand rule. This forwarding stops when an informed cell (not counting the surrounding messages of the previous round) was found. Then, a message is sent back noticing the entry point about the number of border cells found.

Now the Grid-BFS phase of the round starts if at least $\frac{1}{4}2^i$ border cells have been found. In this round a BFS-tree of depth 2^i will be built using the c-competitive frame multicast inside the sub-grids of size $g \times g$. This process reuses a BFS-tree of the preceding round, which shows sub-branches where the leaves remain to be explored. Every time a message reaches the outside of such a sub-square, an entry point at the neighbor cell is created and again the c-competitive sub-routine starts. The BFS stops if messages from different entry points of the square were found.

After the forwarding phase, the BFS-tree will be used for gathering information for the coordinating entry points. This information is the number m_i of newly found border cells, and the paths which should be explored in the next round.

For the time and traffic analysis we concentrate on the case of one entry point. For more than one entry point the proof is analogous and the overall traffic and time remains the same.

The time behavior follows from the c-competitivity of the sub-grids and the structure of the search. For the traffic let $m_{i,j}$ denote the number of border cells found in sub-square j in the ith round. Because we count only the newly found border cells we have $\sum_{i,j} m_{i,j} = m$.

Then there are the following cases.

1. $m_i \leq \frac{1}{4}2^i$: Then at most $4 \cdot 2^i$ messages are transmitted. Summing over all rounds yields traffic $\mathcal{O}(h)$.

2. $\frac{1}{4}2^i \leq m_i \leq 2^{2i}$: Then the traffic in the i-th round is linearly bounded by the following term.

$$\begin{aligned} \sum_{j=1}^{2^{2i}/g^2} \min\{g^2, g + (m_{i,j})^2\} &\leq \sum_{j=1}^{2^{2i}/g^2} g + \sum_{j=1}^{2^{2i}/g^2} \min\{g^2, (m_{i,j})^2\} \\ \leq \frac{2^{2i}}{g^2}g + \sum_{j\,:\,m_{i,j}>g} g^2 + \sum_{j\,:\,m_{i,j}\leq g} (m_{i,j})^2 &\leq \frac{2^{2i}}{g} + \frac{m_i}{g}g^2 + \frac{m_i}{g}g^2 \leq \frac{2^{2i}}{g^2}g + 2m_i g \end{aligned}$$

Note that $\sum_i m_i = m$ and thus the overall traffic is bounded by $\mathcal{O}(\frac{h^2}{g} + mg)$.

■

Combining these Lemmas we can state the following theorem.

Theorem 8 *For every $g \leq h$ Chessboard routing as a position-based multi-message routing algorithm has dilation $\mathcal{O}(h)$ and traffic $\mathcal{O}(\min\{h^2, \frac{h^2}{g} + mg\})$.*

Proof: We start with four squares of size $h_0 := ||s,t||_1$, such that s and t are on the borders of two of the $h_0 \times h_0$ squares as depicted in Figure 7. Let $i \leftarrow 0$. For the four squares the multicast algorithm is started to construct a BFS tree of depth h_i. If the search for t fails, then the size $h_{i+1} \leftarrow 2h_i$ will be doubled, i will be incremented and the larger squares arranged as shown in Figure 7. Clearly, this algorithm gives time behavior $\sum_{i=\log h_0}^{\log h} c2^i = \mathcal{O}(h)$.

For the traffic we store in each square the paths that cannot be completely traversed because of the time limit. In the next round the BFS-search starts at these leaves, and saves a logarithmic factor (analogously as in Lemma 6 and Lemma 7.) Let m_i be the number of border cells discovered in the i-th round. Then the traffic is linearly bounded by

$$\sum_{i=\log h_0}^{\log h} \min\left\{2^{2i}, \frac{2^{2i}}{g} + m_i g\right\} \leq \sum_{i=\log h_0}^{\log h} 2^{2i} \leq 2h^2 \qquad \text{and by}$$

$$\sum_{i=\log h_0}^{\log h} \min\left\{2^{2i}, \frac{2^{2i}}{g} + m_i g\right\} \leq \sum_{i=\log h_0}^{\log h} \frac{2^{2i}}{g} + m_i g \leq \frac{2h^2}{g} + \sum_{i=\log h_0}^{\log h} m_i g = \frac{2h^2}{g} + mg \; .$$

■

This theorem proves that if the number of border cells is asymptotically smaller than h^2, then routing in linear time can be performed with less traffic than flooding in the worst case.

Corollary 1 *For $g = h^\alpha$, where $\alpha \in [0,1]$ the Chessboard routing algorithm has dilation $\mathcal{O}(h)$ and needs traffic $\mathcal{O}(\min\{h^2, h^{2-\alpha} + mh^\alpha\})$. If an estimation of the number of border cells is known a priori, Chessboard routing needs traffic $\mathcal{O}(\sqrt{m}h)$ with dilation $\mathcal{O}(h)$.*

This corollary shows an asymptotic reduction of traffic compared to all protocols discussed above, e.g. if $\alpha = \frac{1}{2}$ then $m = o(h^{\frac{3}{2}})$. If the number of border cells m is reported by a first run of the routing protocol, one can switch to the second choice $g = \frac{h}{\sqrt{m}}$ which for all $m = o(h^2)$ asymptotically needs less traffic than h^2.

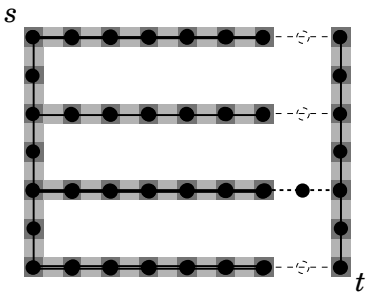

Figure 8: The graph family used for the lower bound.

5 Lower Bounds

Theorem 9 *Every $O(1)$-proactive routing algorithms needs at least traffic $\Omega(m)$ and dilation $\Omega(h)$ for $m > h$ border cells and minimum distance h between source and target.*

Proof: We consider a family of k node sets $\mathcal{G} = \{G_1, \ldots, G_h\}$. Each of the node sets consists of two vertical chains of $2h$ nodes and m/h horizontal chains of h nodes, where the distance of each node to its neighbors in a chain is the maximum transmission range r. All horizontal chains are parallel and have a minimum distance of $2r$. Their starting points are within the transmission range of the left vertical chain, which contains the starting node of the routing problem. The target node is in the right vertical chain, which cannot be reached from all but one of the horizontal chain nodes (cf. Fig. 8).

First note that for a $O(1)$-proactive routing algorithm the information available at all but some $O(1)$ nodes of each horizontal chains on the right side is exactly the same.

Using less than $m/2 - \mathcal{O}(1)$ messages can only delivered to half of the rightmost nodes of the horizontal chains. Now choose a random graph of this family. With probability $\frac{1}{2}$ every routing algorithm must fail to reach the target using less than $m/2 - \mathcal{O}(1)$ messages. ■

Theorem 10 *Every $O(1)$-proactive single message routing algorithms needs at least dilation and traffic $\Omega(m)$.*

Proof: The proof uses the same graph family $\mathcal{G}$, see Fig. 8, and is analogous to the proof of Theorem 9. ■

References

[1] Reuven Bar-Yehuda, Oded Goldreich, and Alon Itai. On the time-complexity of broadcast in multihop radio networks: An exponential gap between determinism and randomization. *Journal of Computer and System Sciences*, 45(1):104–126, August 1992.

[2] Stefano Basagni, Imrich Chlamtac, Violet R. Syrotiuk, and Barry A. Woodward. A distance routing effect algorithm for mobility (DREAM). In *Proc. 4th Annual ACM/IEEE Int. Conf. on Mobile Computing and Networking, MOBICOM'98*, pages 76–84. ACM Press, 1998.

[3] Ljubica Blažević, Levente Buttyán, Srdjan Čapkun, Silvia Giordano, Jean-Pierre Hubaux, and Jean-Yves Le Boudec. Self-organization in mobile ad-hoc networks: the approach of terminodes. *IEEE Communications Magazine*, 39(6):166–174, June 2001.

[4] Prosenjit Bose, Pat Morin, Ivan Stojmenovic, and Jorge Urrutia. Routing with guaranteed delivery in ad hoc wireless networks. *Wireless Networks*, 7(6):609–616, 2001.

[5] Holger Füßler, Jörg Widmer, Martin Mauve, and Hannes Hartenstein. A novel forwarding paradigm for position-based routing (with implicit addressing). In *Proc. of IEEE 18th Annual Workshop on Computer Communications (CCW 2003)*, pages 194–200, October 2003.

[6] Silvia Giordano, Ivan Stojmenovic, and Ljubica Blažević. Position based routing algorithms for ad hoc networks: A taxonomy. To appear in *Ad Hoc Wireless Networking*, Kluwer, 2003.

[7] Marc Heissenbüttel and Torsten Braun. A novel position-based and beacon-less routing algorithm for mobile ad-hoc networks. In *Proceedings of the 3rd IEEE Workshop on Applications and Services in Wireless Networks (ASWN' 03)*, pages 197–209, Bern, Switzerland, July 2003.

[8] Tingh-Chao Hou and Victor Li. Transmission range control in multihop packet radio networks. *IEEE Transactions on Communications*, 34(1):38–44, January 1986.

[9] Jerzy W. Jaromczyk and Godfried T. Toussaint. Relative neighborhood graphs and their relatives. *Proceedings of the IEEE*, 80:1502–1517, 1992.

[10] Brad Karp and H. T. Kung. GPSR: greedy perimeter stateless routing for wireless networks. In *Mobile Computing and Networking*, pages 243–254, 2000.

[11] Young-Bae Ko and Nitin H. Vaidya. Location-aided routing (LAR) in mobile ad hoc networks. *Wireless Networks*, 6(4):307–321, 2000.

[12] Evangelos Kranakis, Harvinder Singh, and Jorge Urrutia. Compass routing on geometric networks. In *Proc. 11th Canadian Conference on Computational Geometry*, pages 51–54, Vancouver, August 1999.

[13] Fabian Kuhn, Roger Wattenhofer, and Aaron Zollinger. Asymptotically optimal geometric mobile ad-hoc routing. In *Proceedings of the 6th international workshop on Discrete algorithms and methods for mobile computing and communications*, pages 24–33. ACM Press, 2002.

[14] Fabian Kuhn, Roger Wattenhofer, and Aaron Zollinger. Worst-case optimal and average-case efficient geometric ad-hoc routing. In *Proceedings of the fourth ACM international symposium on Mobile ad hoc networking and computing*, pages 267–278. ACM Press, 2003.

[15] Martin Mauve, Jörg Widmer, and Hannes Hartenstein. A survey on position-based routing in mobile ad hoc networks. *IEEE Network Magazine*, 15(6):30–39, November 2001.

[16] Charles E. Perkins, Elizabeth M. Belding-Royer, and Samir Das. Ad hoc on-demand distance vector (AODV) routing. IETF RFC 3561, July 2003.

[17] Ivan Stojmenovic. Position based routing in ad hoc networks. *IEEE Commmunications Magazine*, 40(7):128–134, July 2002.

[18] Ivan Stojmenovic and Xu Lin. Loop-free hybrid single-path/flooding routing algorithms with guaranteed delivery for wireless networks. *IEEE Transactions on Parallel and Distributed Systems*, 12(10):1023–1032, 2001.

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