Ant-based Mobile Routing Architecture in large-scale mobile ad-hoc networks

Diplomarbeit
der Philosophisch-naturwissenschaftlichen Fakultät
der Universität Bern

vorgelegt von

Thomas Huber
2004

Leiter der Arbeit:
Professor Dr. Torsten Braun
Institut für Informatik und angewandte Mathematik

Contents

List of Figures

Acknowledgement

First of all I would like to thank Professor Dr. Thorsten Braun that I could write this diploma thesis in his research group *Computer Networks and Distributed Systems*. Also many thanks go to Marc Heissenbüttel for supervising this work and giving me lots of helpful proposals if needed. Last but not least I would like to thank all the people who gave me any support. A very special thank goes to my family which gave me a generous support during all my studies.

Chapter 1

Introduction

Recently, *mobile computing* has increasingly become the focus of interest. Cellular networks as well as wireless local area networks (WLANs) and personal area networks (PANs) have come into commercial focus. A new technology not yet used in mainstream products are *mobile ad-hoc networks*.

In cellular networks, mobile devices communicate over a fixed installed infrastructure. Only the last hop of the communication from the fixed antennas to the mobile devices is done with a wireless transmission. Also, in WLANs and PANs, the wireless transmissions are normally used to connect a wireless device over only one hop to a fixed network or directly to the communication peer.

In mobile ad-hoc networks the whole data communication from one member of the network to its communication peer is wireless. If the two communicating nodes are not within transmission range of each other, intermediate members of the mobile ad-hoc networks between the two communication partners serve as infrastructure to route the data traffic through the network.

A big advantage of a mobile ad-hoc network is that no fixed infrastructure is needed at all. After environmental disasters with no available working infrastructure, in military operations or as sensor networks in inaccessible regions, mobile ad-hoc networks could have lots of advantages. An even higher goal would be to substitute today's cellular networks completely by a mobile ad-hoc network.

One of the main problems in mobile ad-hoc networks is how the routing of data packets should be done. Because all the participant nodes may change their position frequently, the network topology changes quickly and therefore topology based routing algorithms may run into problems.
One approach to solve this problem are position-based routing protocols. Assuming that every node knows its own geographical position and also the position of its direct communication partners. Data packets are routed through the network according to a destination position information stored in the packet. This kind of routing leads to other problems such as routing around a mobile-node-free zone. Ant-based routing protocols are suggested to improve the routing decisions. They try to adapt natural behavior of insect swarms in order to achieve a better routing.

In this thesis, an ant-based routing approach is introduced that is designed for routing in large-scale mobile ad-hoc networks. The basic idea of the algorithm was first proposed in the

technical report: *Ants-Based Routing in Large Scale Mobile Ad-Hoc Networks* [1]. Large-scale mobile ad-hoc networks stand for mobile ad-hoc networks with several thousand participant mobile nodes. To be able to run simulations with that amount of mobile nodes, a simple network simulator was implemented on which the newly algorithm was tested.

In chapters 2 and 3, some related work and the basic ideas of position and ant-based routing is introduced. Our ant-based routing architecture is described in detail in chapter 4. To simulate mobile ad-hoc networks, the individual movements of the nodes are coordinated by a mobility model. Chapter 5 presents such mobility models for mobile nodes. The simulator that was used for the simulations is introduced in chapter 6. The results gained from the simulations are shown in chapter 7 and finally in chapter 8 the conclusion and some ideas for future work will be laid on forth.

Chapter 2

Position-Based Routing Algorithms

A data packet is sent through the network according to its geographical destination information, set by the sending node. Intermediate nodes analyze this information and send it further in the appropriate direction.

Other terms for Position-Based Routing are *geometric*, *geographic* or *location-based* routing all these terms being used as synonyms.

To use position-based routing algorithms, every node in the network needs to know its own geographical position, for instance from a GPS (Global Positioning System) receiver. A node that intends to send a packet also needs to know the geographical position of the destination node. How the sending node achieves this information is not part of this thesis. Several approaches were proposed in the literature.

2.1 Location Services

If a node needs to know the position of a communication peer, it uses a location service. Today's cellular networks have designated position servers with a well-known address where all the position information of the participating nodes is maintained.

In a mobile ad-hoc network, no centralized service is available. Nodes cannot request the position of other nodes at a known address. A centralized service would contradict the ideas of mobile ad-hoc networks.

Having a service that is a part of the mobile ad-hoc network poses a kind of chicken-and-egg problem, because somehow a node that provides this service must be found. Additionally, it is also difficult to guarantee that such a position server will always be reachable, since in an ad-hoc network nodes may or may not be present at a certain time.

In this section, only one location service called *Homezone* [2] is briefly presented, as an example how the location service problem could be solved. Other possible solutions would be the *Grid Location Service* [3], the *Quorum-Based Location Service* [4] or the *Distance Routing Effect Algorithm for Mobility* [5].

Homezone

Every node belongs to a geographical area where its position information is stored, the *homezone*. All nodes within the area of the node's homezone save and maintain the position of the node, which sends the needed information back to its homezone regularly. If a node needs to know the position of a communication partner, it sends a request to the homezone of the partner, and one of the nodes currently within that homezone replays with the position of the node that was asked for. The position of the homezone of this node can be derived from the node identifier by applying a hash function. Further information can be found in [2].

2.2 Greedy Packet Forwarding

A packet that must be sent is marked with the position of the receiving node. With the information of the target position, intermediate nodes that receive the packet for redirecting, forward the packet to a neighboring node into the general direction of the destination. This is done by every intermediate node and if everything works out fine, the packet finally arrives at its destination.

If a node has more than one neighbor within its transmission range and between the destination position and its own, it selects one of them according to different possible algorithms: *Closest to destination* [6], *Most Forward within Radius (MFR)* [7], *Nearest Forward Progress* [8], *Compass routing* [9], *Nearest Closer* [10].

Closest to Destination

In the closest-to-destination method, the node that most shortens the remaining distance to the destination is chosen [6]. In figure 2.1 node b would be selected as next node from the current sender s, because its distance to the destination $\overline{b, d}$ is shorter than the distance of node a.

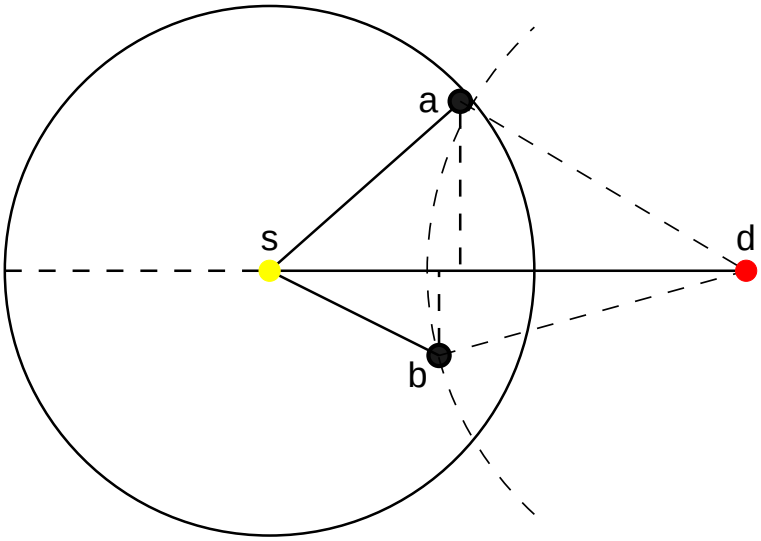


Figure 2.1: MFR and Closest-to-destination node selection

Most Forward Within Radius (MFR)

Proposed by [7], this algorithm chooses the node which makes the biggest progress if projected on the straight line from sender s to destination d. In the situation of figure 2.1 node a would be chosen as the next intermediate node by MFR. Unlike for this example, in most cases, the node chosen by MFR is the same as *closest to destination* would choose.

Compass Routing

In compass routing, proposed by [9], the angles between the straight line from sender to destination and the lines from sender to candidate nodes are decisive. Only nodes with a positive forward progress are taken to account. Finally, the neighbor with the smallest calculated angle is chosen. In the example of figure 2.2 node c is the next intermediate node.

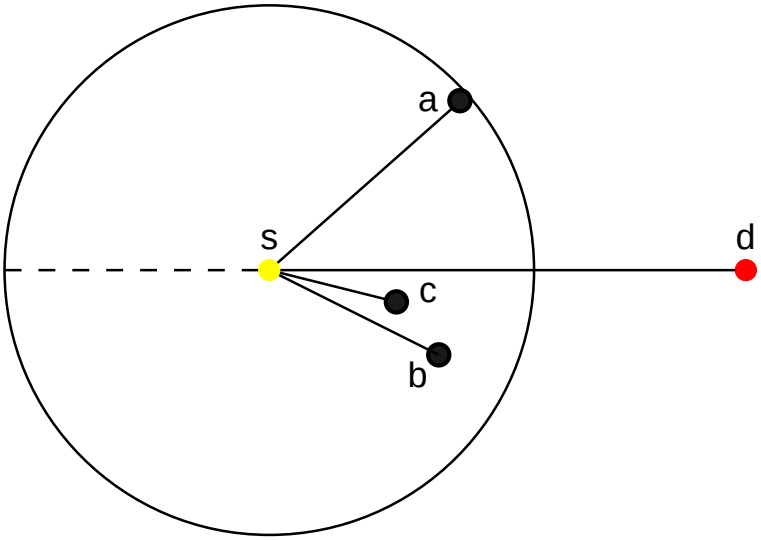


Figure 2.2: Compass-routing node selection

Drawback of Greedy Routing

Because packets can only be forwarded in the general direction towards the destination node, the greedy routing algorithms may not find a path from source to destination even if one does exist. Figure 2.3 shows a network situation where a packet sent from source node s will not arrive at the destination node d using only greedy forwarding. The packet gets stuck at node e because it cannot be forwarded to the destination d in a greedy manner via a neighbor and with positive forward progress. It will not arrive at its destination even if there is a path over the nodes e, f, g, h, i, j, d.

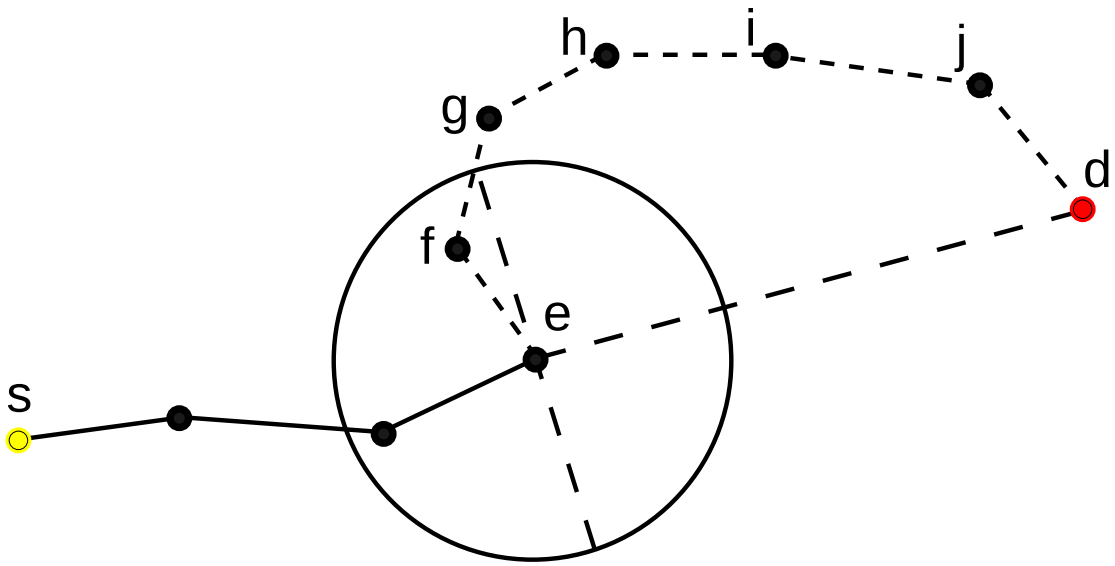


Figure 2.3: Failed Greedy routing

2.3 Planar Graphs

A network with nodes that have a certain transmission range can be seen as a graph where each node is a vertex. Existing links between two nodes in the network are represented as edges in the graph. Only the two dimensional case is discussed here for reasons of simplicity.

If every vertex is connected with every other vertex within its transmission range, we get an *unit-disk graph*, as shown in figure 2.4, with many crossing edges. The goal of a *planar graph* is to eliminate the crossing of edges but without splitting the graph. If the graph is connected in the unit-disk graph it must still be connected in the planar graph. A planar graph does not have any crossing edges. In this thesis two kinds of planar graphs will be discussed, the *gabriel graph* [11] and the *relative-neighborhood graph* [12]. Many other exist, but for practical reasons planar graphs should be locally computable. To compute the gabriel graph or the relative-neighborhood graph, a node only needs to know the positions of all its neighboring nodes within its transmission range.

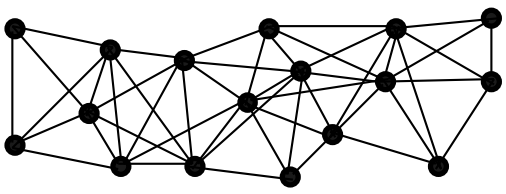


Figure 2.4: Unit-Disk-Graph example

2.3.1 Gabriel Graph

In the gabriel graph two vertices a and b, which are within the transmission range of each other, are connected by an edge if there is no other vertex within the circle drawn through the vertices at a diameter of $\overline{ab}$. The example of figure 2.5 shows a situation where a and b are connected by an edge because there is no other node within the circle area. Every edge in the Unit-Disk Graph is now checked if the condition is fulfilled otherwise the edge is not part of the gabriel graph.

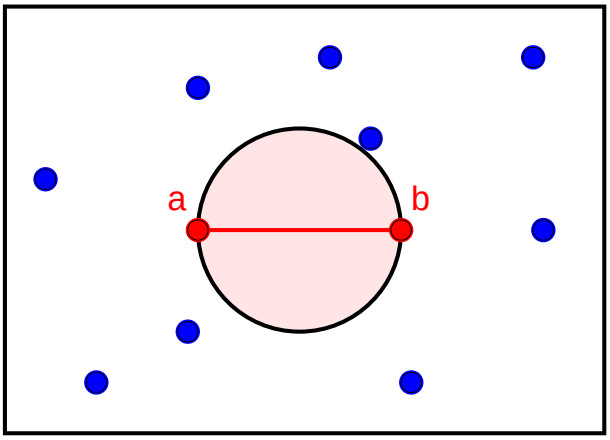


Figure 2.5: Gabriel Graph

2.3.2 Relative-Neighborhood Graph

The Relative Neighborhood Graph is very similar to the gabriel graph but uses another area where no other vertex is allowed to exist. Around both vertices a circle with radius $\overline{ab}$ is drawn. To have an edge between a and b, no other vertex is allowed to stay within the intersection of the two circles as shown in figure 2.6.

2.3.3 Comparison

Both described graphs accomplish the two conditions that no edges cross in the extracted planar graph and that the graph is connected. The graph constructed by the Relative Neighborhood Graph is less dense than the one constructed with the gabriel graph algorithm, due to a bigger zone in which no vertices are allowed. Figure 2.7 shows the different graphs built by the two planar graphs with the same network situation from the unit-disk graph in figure 2.4.

2.4 GFG/GPSR

In the GFG/GPSR (Greedy Face Greedy/Greedy Perimeter Stateless Routing) algorithm [13] and [14], every node in the network needs to know its own and the geographical position of all its neighbors within the transmission range (single-hop neighbors). To indicate its own position

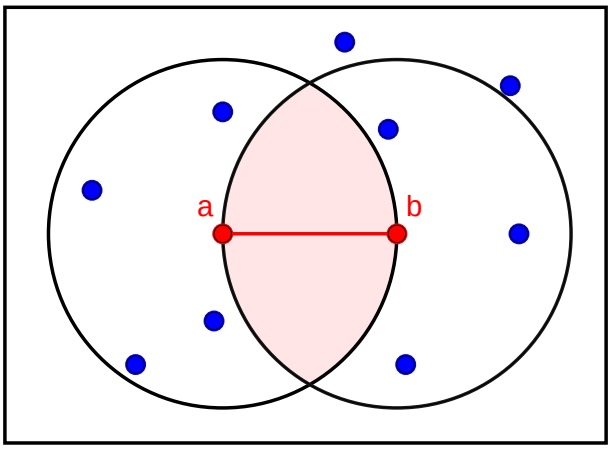
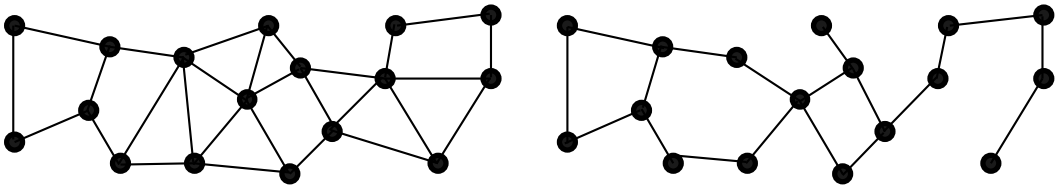


Figure 2.6: Relative Neighborhood Graph



(a) Gabriel Graph

(b) Relative Neighborhood Graph

Figure 2.7: Comparison of planar graphs

to the potential neighbors, a node periodically sends a Hello message with position information (known as beaconing). All nodes within transmission range can thus update their neighbor tables.

GFG/GPSR contains two routing modes, the *greedy mode* and the *perimeter mode*. Basically GFG/GPSR tries to route a packet in the greedy mode to obtain a good performance, but as mentioned above in section 2.2 greedy routing may get stuck in certain network topologies. In a situation where the packet cannot be redirected further with greedy routing, GFG/GPSR switches to the perimeter mode (backup mode) that is also called *face-routing* mode.

In the perimeter mode, a node only forwards packets to nodes where it has a connection according to the rules of a planar graph as described in section 2.3. When using the right-hand rule on a planar graph, it is certain that a packet will arrive at the destination if a path exists, as shown in [13]. To select the next node in the right-hand rule, a straight line is laid through the last sending and the current node. This line is turned around the current node counterclockwise until the line hits a node that is connected with the current node in the planar graph. This node is the next node towards which the packet is sent to.

The right-hand rule allows to route a packet around the face between the destination and the node where greedy got stuck. In figure 2.8, at node a, the routing mode changed from greedy to

perimeter because the greedy algorithm got stuck.

The position of the node where the mode changed from greedy to perimeter is stored in the transmitted packet. As soon as the packet arrives at a node that is closer to the destination than the node where the algorithm switched to perimeter mode, the packet is sent in greedy mode again, until it gets stuck once more and has to change back to perimeter mode. In figure 2.8 at node b the algorithm changed back from perimeter to greedy mode.

To avoid endless loops, a packet that should be sent via the same link for a second time is immediately dropped because the destination node is not reachable if a loop occurs.

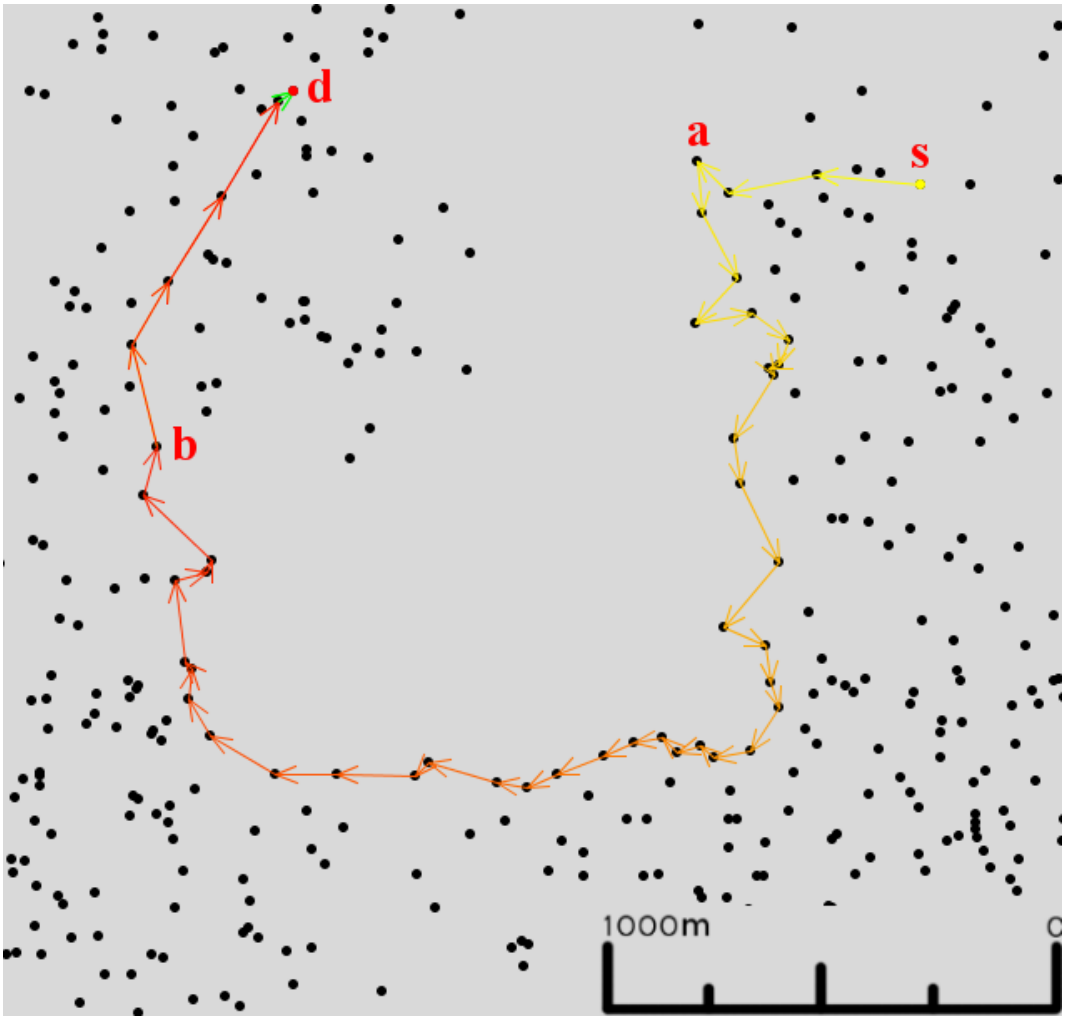


Figure 2.8: Example of sending a packet with GFG/GPSR from node s to node d: At node a greedy routing fails and the packet is redirected in perimeter mode until node b, where routing is changed back to greedy mode, as b is closer to the destination than a.

2.5 GOAFR$^+$

GOAFR$^+$ (Greedy Other Adaptive Face Routing Plus) is another position-based routing algorithm that works, similar to GFG/GPSR, in two modes, the greedy and the face-routing mode. The greedy mode is exactly the same as in GFG/GPSR and uses the *closest-to-destination* method to choose the next node. If greedy mode gets stuck, a face-routing algorithm is used as a backup mode. GOAFR$^+$ is proofed to be asypmtotically optimal and also efficient on average-case graphs [15].

The backup mode is an adapted face-routing algorithm that was first proposed in [9]. The face routing works on the planar graph and routes along the boundaries of the faces. In GOAFR$^+$ the area where the face routing searches a path to the destination is restricted by a circle, the packet may not be routed beyond this circle. Only if the search is not successful the search area is enlarged.

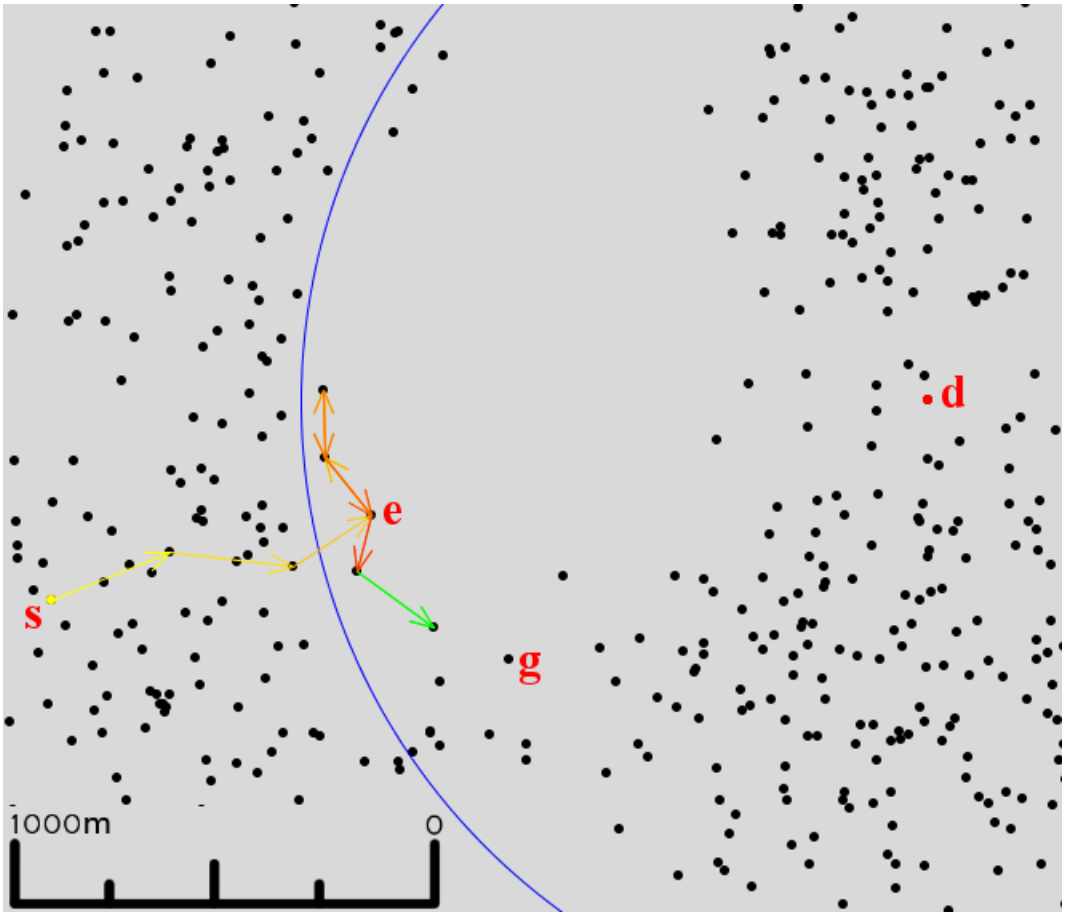


Figure 2.9: Example of sending a packet with GOAFR$^+$ from node s to node d. At node e the greedy algorithm gets stuck, after exploring the upper boundary of the circle in perimeter mode, the packet is sent greedy again from node g.

If a source node s begins to send a packet, it initializes a circle C with a radius $r_C = \rho_0|\overline{sd}|$ where $1 < \rho_0$. If the packet is greedy forwarded, the radius of C is reduced whenever possible by $r_C = r_C/\rho$ ($\rho_0 < \rho$) ensuring that the current node always stays within the circle.

If the greedy mode gets stuck at node e, the backup mode tries to find a closer node to the destination d than e within the circle C using face routing. If the packet hits the boundaries of C for the first time, the packet turns and explores the other way around the face. If the packet cannot find any node that is closer to d than u, the radius of C is enlarged, and the packet starts again to explore the boundaries of the circle.

When a node g is found that is closer to the destination d than the node e, where the backup mode was started, the packet is forwarded greedy again from this node g after exploring the boundaries of the circle. Finally, the radius of C is adapted with the former rules. There is a special rule according to which the algorithm can also fall back immediately into greedy mode when a node that is closer to d than e is found, for further information see [15].

If no node that is closer to d than e can be found, a disconnection report is sent back to the source s, which is also sent with GOAFR$^+$.

Figure 2.9 shows the circle in which the face routing tries to find a closer node than e, where the greedy mode got stuck. As can be seen, due to the circle functioning as a boundary for path exploring, the packet only takes a few hops into the wrong direction and then turns to explore the other side, where it will switch back to greedy mode at node g.

2.6 Discussion

The main drawback of these position-based routing algorithms is that they are stateless. A stateless network has no possibility to change the routing behavior if the efficiency of the routing decisions is low. In the same network topology, the described algorithms in this section will always route a packet along the same path.

The algorithm introduced in this thesis (AMRA) enlarges these position-based algorithms with a memory about past traffic. The decision along which path a packet is routed is dependent on collected and stored information that is gained out of the overheard network traffic. It might differ from one packet to the next, even if the network topology did not change at all.

Chapter 3

Ant-Based Routing Algorithms

In ant-based routing algorithms, natural behavior of insect swarms is taken as an example. With a few basic rules that every individual of a swarm has to follow, complex task are solved by cooperation. Further information on swarm intelligence can be found in [16],[17], and [18].

In this section two algorithms for mobile ad-hoc networks are discussed. Ant-based algorithms are also proposed for fix-net routing as in *AntNet*[19].

3.1 Behavior of Ants

Ants deposit pheromone while walking to find the way back to their ant-hill and also to help the other members of the colony to orientate themselves. Pheromone that has been deposited decreases its concentration after a while due to diffusion. In figure 3.1(a) ants are looking for food starting at the ant-hill having two different paths available. Every ant deposits a pheromone trail. The pheromone deposited by the ants taking the same path is added. After loading food, the ants go back the path they came, while the ones that accidently took the shorter path will return to the ant-hill earlier. In figure 3.1(b) the blue circles mark the ants that took the shorter path. Because they also deposit pheromone on their way back, the shorter path gets a higher concentration of pheromone than the longer one, on which the returning ants are not on the way back to the ant-hill yet. Following ants will favor the path with the higher pheromone concentration and therefore more and more ants will choose to use the shorter path. Finally, the longer path is not used any more and the ant colony has optimized food winning as shown in figure 3.1(c). A problem is a path with a dead end that might be favored because the wrongly directed ants return faster. This is avoided because ants deposit more pheromone if they are loaded with food. Like this a path that successfully leads to a foot place gets a stronger pheromone trail.

The behavior of food-searching ants is now adapted in mobile ad-hoc networks to find the most efficient path throughout the network from a specific source to a destination node.

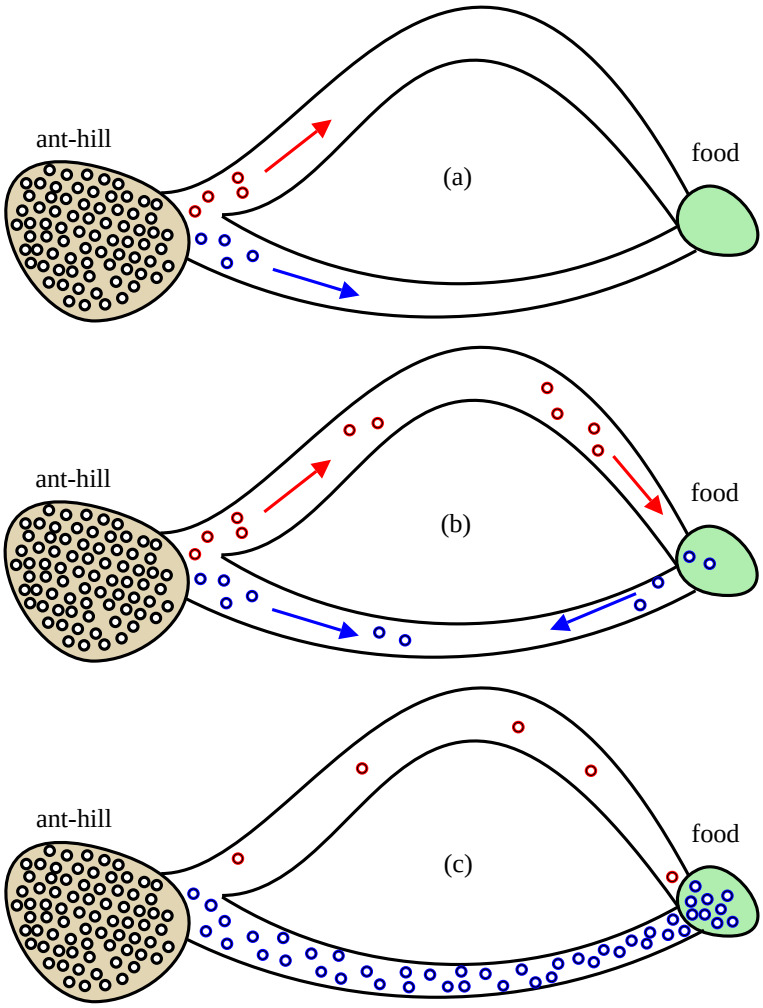


Figure 3.1: Ants optimizing the food winning

3.2 Ant-colony-based Routing Algorithm (ARA)

This algorithm uses ants to establish and maintain paths from a source node to a destination node and was published in [20]. It can be divided into three parts, the *Route Discovery Phase*, the *Route Maintenance* and the *Route-Failure Handling*.

3.2.1 Route Discovery Phase

In this phase new paths between a source and a destination are discovered, therefore two kind of ants are generated, the *forward ant* (FANT) and the *backward ant* (BANT). FANTs are flooded from the initiating source node to the destination node and they set a pheromone trail in the opposite direction from the destination to the source. Then a BANT routes along the best trail the FANTs just generated back to the source and lays thereby a pheromone trail from the source

to the destination.

Nodes maintain a routing table where entries are stored as tribles *(destination address, next hop, pheromone value)*. If a FANT is received by a node, a record is generated in the routing table, using the source address of the FANT as its destination, the node that last sent the packet as its next hop and the number of hops done by the FANT to calculate the pheromone value. The packet is then redirected to the neighbors of the node. To avoid that a packet generates more than one entry, they are marked with a sequence number.

The effect of such a FANT is a trail with pheromone that is set from the current position of the FANT back to the source. All the intermediate nodes have an entry in their table to which they have to send a packet that must be sent to the source address.

If a FANT reaches the destination node, a BANT is generated that follows the just set pheromone trail back to the source node. At every node it is redirected, an entry similar to the one made by the FANT that directs to the destination node is generated. As soon as the BANT arrives at the source node, pheromone trails are set in both directions from the source to the destination node and back, and the network is prepared for data traffic between these two nodes.

3.2.2 Route Maintenance

During the communication between the source and the destination node, the connection needs to be maintained. For this task, no special packets like FANTs or BANTs are needed, only the regular data traffic is used. When a data packet is sent along an existing pheromone trail, additional pheromone is added to the respective entries in the routing tables of the intermediate nodes. This ensures that used and working trails are kept alive.

As pheromone of real ants that volatilizes after a while, the pheromone values in the routing tables are decreased in regular time intervals. Unused paths will disappear.

3.2.3 Route-Failure Handling

Routing failures can occure if a link to a node with a high pheromone value in the routing table is not reachable anymore. This is mostly caused by the movements of the nodes and often happens in a mobile ad-hoc network.

If a node cannot succeed in redirecting a packet to a former neighboring node, the pheromone entry of that node is deactivated. The node then searches for an alternative path in its routing table to redirect the packet further towards its destination. If there is no other path available, the node informs the source. A new route-discovery process is needed.

3.3 Termite

Termite is another routing protocol that is based on swarm intelligence using pheromone to mark high-performance paths throughout a mobile ad-hoc network [21][22].

A routing table containing a row for each neighbor within the transmission range and a column for every destination the node knows of is maintained in every node. The size of the table depends on the number of destinations the node overheard and on the number of current neighbors. It changes continuously according to the current network situation.

A packet that arrives at a node increments the pheromone concentration by a constant value in the appropriate row that represents the neighboring node that last transmitted the packet. Like this, similarly to the ARA protocol, pheromone trails are laid throughout the network. In this algorithm, the pheromone values have an *initial*, a *maximum* and a *minimum value*. The initial value is set if a new destination is discovered, the maximum value is the pheromone ceiling that prevents extreme differences among the values. Over time, added pheromone is decreased exponentially, which makes values of unused links fall below a minimum value. If all the values in a column of the routing table are below that minimum value, the node assumes that the destination represented by that column is not reachable any more and therefore the whole column is deleted. If direct neighbors are not reachable anymore, the corresponding rows are also deleted from the routing table.

Data packets are randomly forwarded to the next node based on the amount of pheromone in the routing table for the specific destination of the packet, but never to the node the packet just came from.

Termite protocol knows five different types of packets. Four of them are used as control messages to maintain the routing table and one type are the *data packets* that are routed through the network using the collected routing information. One type of control messages are the *route-request packets* that nodes can use if they have no path to a certain destination to which they need a connection. *Route-reply packets* are the answer to route-request packets; they are sent from nodes that have the information the requesting node needs. With *hello packets* the neighboring nodes inform each other about their presence and finally *seed packets* can be used to actively spread pheromone pointing towards the node that generated the seed packet.

3.4 Discussion

The algorithms discussed in this section only scale in small networks with a few participating nodes, because every path is stored for a specific destination node. The bigger the network, the more information must be stored and maintained in the routing tables. Another problem is the effort that is needed to find a new path if a link between two nodes breaks due to mobility: new ants must be sent to reestablish the path.
The paths in the discussed algorithms are stored hop by hop and therefore one broken link corrupts a whole path. In a network with high mobility, links break all the time what makes the maintenance of existing paths almost impossible.

The AMRA algorithm of section 4 reduces scalability problems by grouping nodes. Pathes are not maintained to specific nodes, but to geographical regions. The amount of stored information

is not dependent on the number of nodes participating the network, it is dependent on the size of the area that is covered by the network. A path through a network is not stored hop by hop, the nodes only store a general direction in which the packet must be sent. Thus, a broken link between to nodes has no further consequences, the packet is simply sent to another node in approximately the same direction.

Chapter 4

Ant-Based Mobile Routing Architecture (AMRA)

The Ants based Mobile-Routing Architecture is a table-driven routing algorithm operating on an underlying position-based algorithm (for example GFG/GPSR explained in section 2.4). To maintain the information in the tables, it uses the ongoing data traffic in the network and generates additional traffic (ants) to discover new or better paths through the whole network.

For the purpose of this thesis a better path means less hops from the source to the destination node. To achieve this objective, the backup mode of the underlying routing algorithm is avoided whenever possible, by directing the data traffic along efficient paths. To do so, AMRA sets *Anchor Points* along the path over which the packet is to be routed as a kind of direction signs. An Anchor Point is a virtual position coordinate, to which the data packet is routed.

This method works best, if the underlying algorithm forwards packets with a *closest-to-destination*-progress greedy algorithm (section 2.2). This avoids additional hops made by the data packets; if the direction that AMRA proposes is good the backup mode is avoided. The preliminary idea was already published earlier in [1] and was now slightly modified.

4.1 Overview

The AMRA can be divided in four independent parts, working on two layers. On the lower level, the Topology Abstraction Protocol (TAP) explained in section 4.3, builds clusters of nodes to supply a simplified topology of the network to the upper layer. This topology consists of *Logical Routers* and *Logical Links*. Every mobile node is a member of the Logical Router it is currently positioned in.

The Mobile Ant-Based Routing (MABR) protocol, explained in section 4.4, is used on the upper layer to route the data traffic from one Logical Router to the next over the Logical Links. The Logical Link the MABR protocol will use depends on pheromone values that are stored in the routing table of the currently-sending node. These routing tables are maintained by the MABR protocol with the help of the data traffic a node overhears. A Logical Link connects two neighboring Logical Router unidirectionally. Because each Logical Router has eight neighbors, each

Logical Router has eight Logical Links. With this elaborated routing information, the MABR protocol continuously generates new *Anchor Points* for the data traffic.

Finally, the sending of the data packets from one node to the other is the task of the Straight Packet Forwarding (StPF), working on the lower layer, which is explained in section 4.2. This underlying protocol would also work on its own without the enhancements of AMRA, but with lower efficiency. GFG/GPSR is the protocol used for StPF in this work.

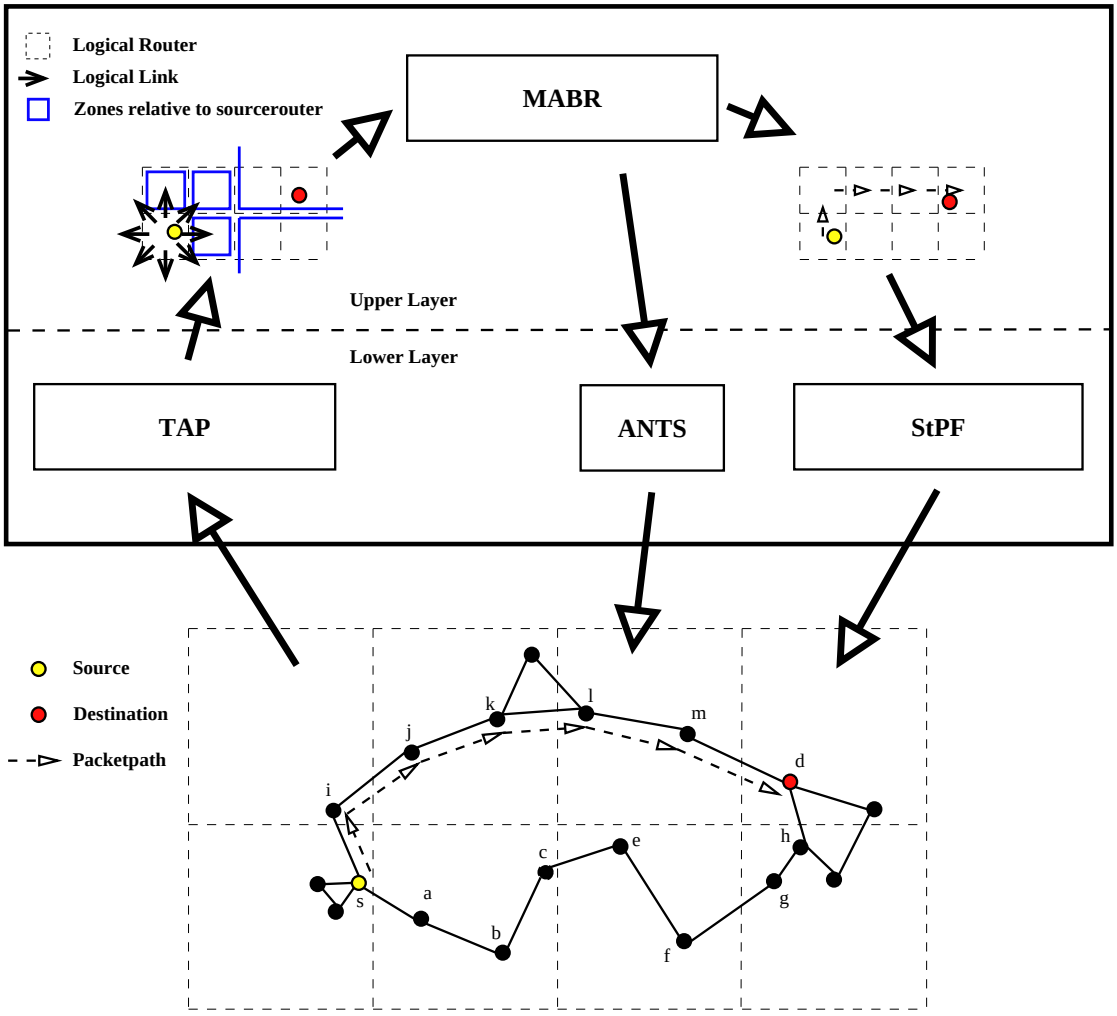


Figure 4.1: AMRA overview

An other part of the AMRA protocol are the *ants*, which are empty data packets sent through the network to help the MABR protocol determine the pheromone values in the routing tables.

Figure 4.1 shows how a data packet might be sent with AMRA and the interaction of the two layers. GFG/GPSR would route the data packet from the source to the destination node over the path $\overline{s, a, b, c, e, f, g, h, d}$. AMRA, in contrast, uses the routing information of the routing tables and knows that it is better to send the packet first to the direction of the router on top of it. So the path $\overline{s, i, j, k, l, m, d}$ is used, which reduces the amount of hop counts from eight hops to six.

4.2 Straight-Packet Forwarding (StPF)

The StPF is responsible for sending a packet from one node to the next on the lower layer of the AMRA protocol. Any position-based algorithm that does not maintain its own routing tables can be used as StPF. In this thesis, the only used algorithm is GFG/GPSR, but also other routing protocols as GOAFR+ could be used. The only condition that the StPF should fulfill so that AMRA can come into its own, is for it to have a greedy routing algorithm as its default routing mode.

In this thesis a slight adaptation of the original GFG/GPSR algorithm in the perimeter mode protocol is used. Namely that a packet can be defined to use ether the left-hand rule or the right-hand rule in the perimeter mode. The effect of this adaption is that the perimeter mode just routes around a face in the other direction. Using only GFG/GPSR, this new possibility would not produce any better routing results because the nodes do not know anything about the state of the network and therefore do not know which mode would perform better. In a table-driven protocol as AMRA it is useful, because in such a case nodes know more about the network state.

4.3 Topology-Abstraction Protocol (TAP)

TAP offers a simplified topology of the mobile network by grouping nodes together and provide *logical* components that are used by the upper layer.

4.3.1 Logical Routers and Zones

A grid divides the whole network area into squares of the same size which serve as Logical Routers on the upper layer. A reasonable choice for the side length of the squares is the transmission range of the nodes. Every mobile node belongs to the Logical Router it is geographically located in. As soon as a node crosses the border from one Logical Router to another, it changes its affiliation to the new one.

From the view of one specific Logical Router, its surrounding Logical Routers are hierarchically grouped into *zones*. The farther away they are located, the more Logical Routers are grouped together within a zone. Figure 4.2 shows an example of the zones around a Logical Router. The Logical Routers that directly flank the one at the origin build a zone themselves, consisting of only one Logical Router numbered as $Z_{1,x}$. These eight zones together build a ring around the origin-logical router with a *zone distance* of one. The zones are numbered with two indices, i

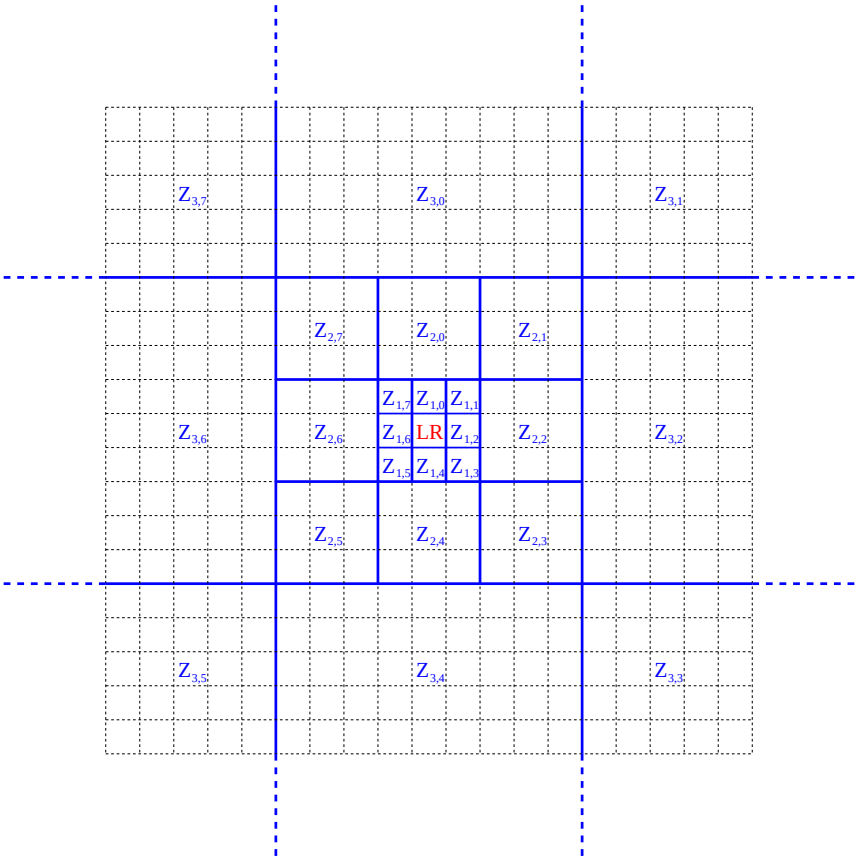


Figure 4.2: Zones surrounding a logical router: The farther away from the origin Logial Router (LR), the more Logical Routers are grouped together within a zone.

indicates the zone distance, j is the index of the zone in the specific ring, starting with 0 at the top, increasing by one turning clockwise around the ring, thus building eight zones.

In the next step, nine Logical Routers are always grouped together, and they form one zone. Eight such zones, numbered as $Z_{2,x}$, with nine Logical Routers each, in turn build a ring around the origin Logical Router and the first ring of Logical Routers. This hierarchical system is processed until all Logical Routers are members of a zone.

It is important for each logical router to build its own zones relative to its position. Thus every Logical Router belongs to different zones, depending on the position of the individual Logical Routers.

4.3.2 Routing Tables

Every mobile node maintains a table with a row for every zone. Every row has nine entries. Eight are used to store the pheromone values of the Logical Links to the neighboring Logical Routers. One entry is reserved for a *mean value* μ. This value μ is calculated with information about the Euclidean distance packets covered that pass the footprint of the node. In each row,

only packets of the zone represented by that row are taken into account, the row with index i, j in the routing table indicating the zone $Z_{i,j}$. The calculation of the pheromone and mean values will be explained more precisely in section 4.4.1.

i . j	n=0	n=1	n=2	n=3	n=4	n=5	n=6	n=7	μ
1 . 0	0.96	0.001	0.033	0.001	0.001	0.001	0.001	0.001	1.395
1 . 1	0.4	0.28	0.312	0.001	0.001	0.001	0.001	0.001	1.845
1 . 2	0.056	0	0.943	0	0	0	0	0	0.867
1 . 3	0.052	0.052	0.129	0.372	0.242	0.052	0.052	0.052	0.89
1 . 4	0.004	0.004	0.004	0.004	0.975	0.004	0.004	0.004	0.527
1 . 5	0.009	0.009	0.009	0.009	0.558	0.151	0.245	0.009	1.571
1 . 6	0.016	0.001	0.001	0.001	0.02	0.041	0.919	0.001	1.381
1 . 7	0.145	0.035	0.035	0.035	0.035	0.094	0.339	0.283	1.64
2 . 0	0.622	0.111	0.076	0	0	0.013	0.001	0.178	3.402
2 . 1	0.047	0.028	0.925	0	0	0	0	0	4.174
2 . 2	0.016	0	0.284	0.59	0.109	0	0	0	2.601
2 . 3	0	0	0.325	0.244	0.407	0.025	0	0	4.347
2 . 4	0	0	0	0	0.981	0.019	0	0	2.972
2 . 5	0.04	0	0	0	0.835	0.126	0	0	5.032
2 . 6	0.001	0.001	0.001	0.001	0.206	0.411	0.379	0.001	3.205
2 . 7	0	0	0	0	0	0.363	0.131	0.506	4.758
3 . 0	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0
3 . 1	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0
3 . 2	0.204	0.016	0.48	0.235	0.016	0.016	0.016	0.016	5.534
3 . 3	0.07	0.07	0.322	0.257	0.07	0.07	0.07	0.07	6.888
3 . 4	0.009	0.009	0.08	0.009	0.59	0.287	0.009	0.009	5.929
3 . 5	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0
3 . 6	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0
3 . 7	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0

Figure 4.3: Example of a TAP routing table

Figure 4.3 shows an example of a routing table of a node. This table has already been used and is updated with information generated from network traffic. The value of the entries in the first eight columns represent the efficiency of a Logical Link to achieve the destination node. The higher a value is, the better the efficiency of the Logical Link. The last column contains the μ values that are scaled down with the length of the transmission range. Thus, the value shows how many transmission radii are needed to get to that zone on average.

When a node connects to the network for the first time, its routing tables are empty. Furthermore the value of every Logical Link is set to 0.125. Entries are probabilities to select that Logical Link for a destination located within the corresponding zone. If after much network traffic a whole row still has the default values, it is highly probable that the zone represented by that row is free of nodes or the zone is disconnected from the rest of the network.
The size of the routing table depends on the area covered by the network and the size of the Logical Routers. The side length of the square covered by a routing table depends on the maximal zone distance i_{max} and the side length of the Logical Routers as shown in equation 4.1.

$$l_{covered\ area} = l_{Logical\ Router} * 3^{i_{max}} \tag{4.1}$$

Figure 4.4 shows a calculation example with a Logical Router size of $250m$. The side length of the area covered by a routing table with a depth of eleven zone distances is $44'287km$. This is enough to embrace the whole equator. Supporting a maximal zone distance of eleven, the routing table has 88 rows, eight per every zone distance. Multiplied with the 9 entries per row needed, 792 values per node have to be stored.

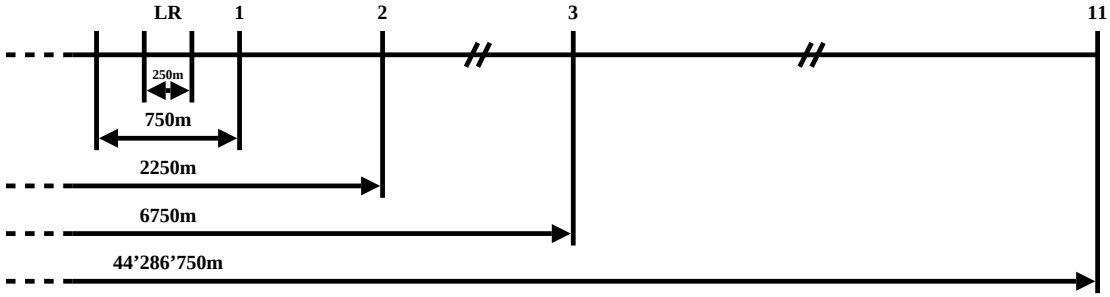


Figure 4.4: Sidelength of the covered area

4.3.3 Logical Links and Anchor Points

A Logical Router has eight Logical Links. Each entry in the first eight columns of the routing table represents one of these Logical Links.

A Logical Link points from the Logical Router to a specific Anchor Point. Figure 4.5 shows all the Logical Links out of a Logical Router with their corresponding Anchor Points and their numbering corresponding to the column number in the routing table. The position of an Anchor Point is set on the outside border or in the distant corner of a neighboring Logical Router according to figure 4.5: the small red squares mark the Anchor Point positions. These Anchor Points are used to direct a packet along the desired path, they are continuously recalculated by the nodes according to their routing tables.

The MABR protocol of the upper layer supplies the StPF on the lower layer with the position of the Anchor Point to which the packet has to be forwarded. Which one out of the eight possible Logical Links is selected is part of the MABR and is explained in chapter 4.4.2.

4.4 Mobile Ant-Based Routing (MABR)

This section describes how the values in the routing table of the TAP are updated and how these values are used to route data packets.

4.4.1 Calculation of the Pheromone Values

The calculation of the pheromone values is done by each network node individually. An exchange of routing information between two different nodes is not supported. For the calculation

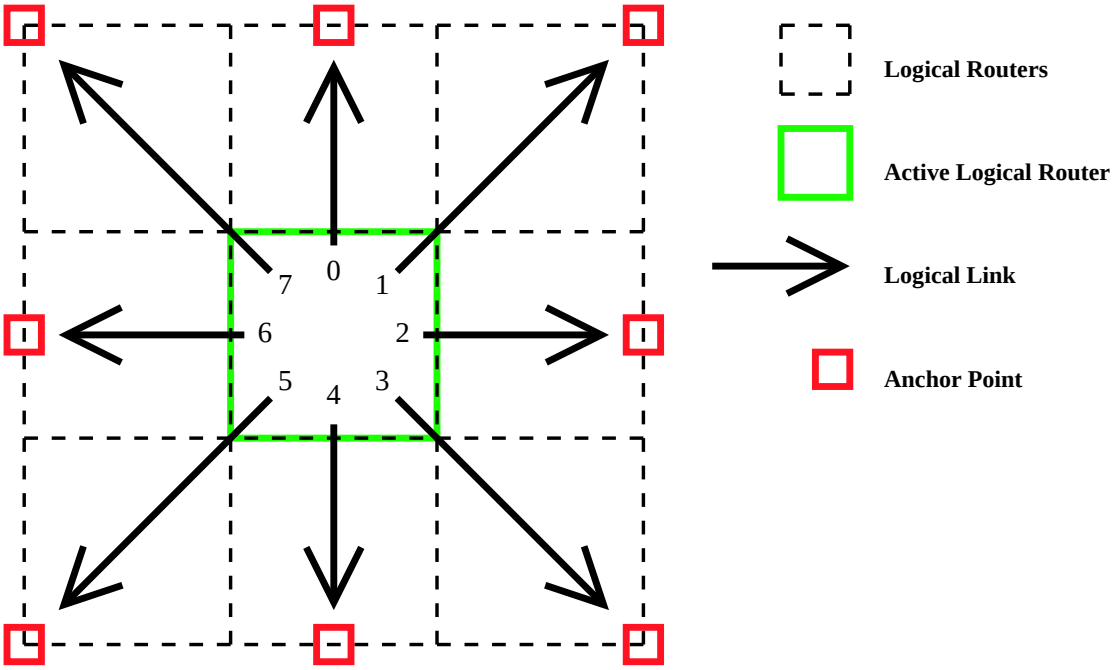


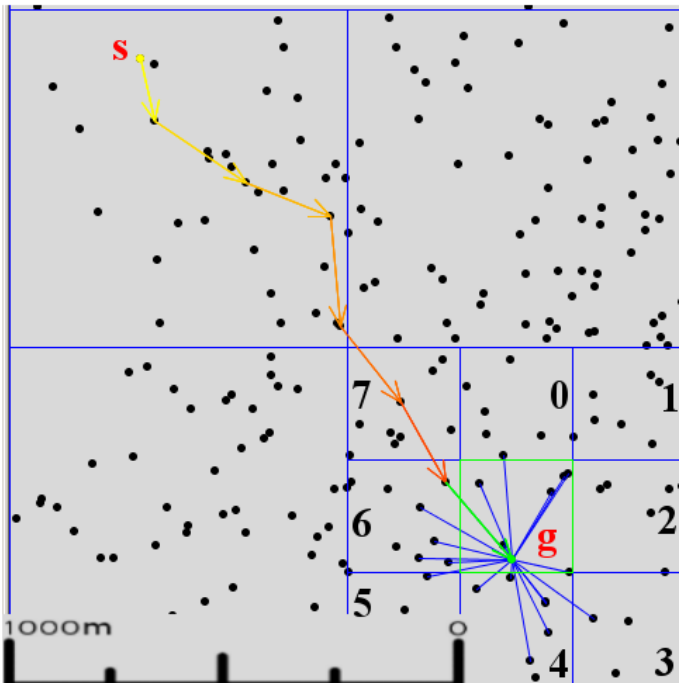
Figure 4.5: Logical Links out of a Logical Router and the corresponding Anchor Points

a node uses the network traffic it overhears even if the network packet itself is not send over.

In every packet header, two geographical pieces of information that are important for the calculation are stored: the position of the source node and the position of the last Logical Router the packet passed. Furthermore, the Euclidean distance the packet traveled to the current position is stored in the packet header. These three information suffice to calculate the pheromone values and the mean value in the routing tables.

When a node overhears a packet, it first analyzes in which zone it was generated. This is done in correlation to with Logical Router the node is currently in. With the geographical information about the source node that generated the packet and its current position this zone can be calculated.

The node uses this information to determine in which row of the routing table the calculations should be done. Figure 4.6 shows a possible situation in a network, where the green node g updates its routing table in row $Z_{2,7}$. $Z_{2,7}$ represents the zone where the packet was generated at the yellow source node s (compare with figure 4.2). The arrow line shows the path the packet has traveled so far, changing its color from yellow to red the closer the packet came to its destination. Not only the green node g, but also all the other nodes overhearing the packet will update their routing tables immediately, but only once per packet the first time they overhear it. A sequence number and a unique source node address in the packet header help distinguish packets that they are not used several times.



(a) The Network with the zones

i_j	n=0	n=1	n=2	n=3	n=4	n=5	n=6	n=7	μ
1_0	0.433	0.039	0.039	0.039	0.039	0.039	0.335	0.039	1.369
1_1	0.067	0.533	0.067	0.067	0.067	0.067	0.067	0.067	1.685
1_2	0.065	0.065	0.368	0.065	0.24	0.065	0.065	0.065	1.501
1_3	0.053	0.053	0.053	0.631	0.053	0.053	0.053	0.053	0.895
1_4	0	0	0	0	0.959	0.041	0	0	0.682
1_5	0.072	0.072	0.072	0.072	0.072	0.265	0.301	0.072	1.008
1_6	0.001	0.001	0.001	0.001	0.001	0.005	0.992	0.001	1.028
1_7	0.314	0.007	0.007	0.007	0.007	0.007	0.422	0.229	2.308
2_0	0.15	0.029	0	0	0	0	0.82	0	3.577
2_1	0.191	0.181	0	0.207	0.416	0	0.005	0	4.559
2_2	0.036	0	0.204	0.207	0.553	0	0	0	3.465
2_3	0.205	0	0	0.141	0.652	0	0	0	4.504
2_4	0	0	0	0.16	0.829	0.011	0	0	2.851
2_5	0	0	0	0	0.75	0.008	0.242	0	4.118
2_6	0.08	0	0	0	0.004	0.001	0.784	0.131	3.075
2_7	0.15	0.001	0.001	0.001	0.001	0.019	0.804	0.023	4.676
3_0	0.04	0.439	0.004	0.004	0.004	0.004	0.502	0.004	7.2
3_1	0.094	0.344	0.094	0.094	0.094	0.094	0.094	0.094	7.714
3_2	0.202	0.166	0.017	0.017	0.546	0.017	0.017	0.017	6.299
3_3	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0
3_4	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0
3_5	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0
3_6	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0
3_7	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0

(b) Current routing table

Figure 4.6: Example for the row selection in MABR calculations

The mean value μ of the appropriate row is calculated as follows. If the first packet arrives from that zone, what means that μ is still set to zero, the Euclidean distance (d_e) traveled through the network so far is taken as the initial value. Every other packet that arrives initiates a recalculation of μ according to formula 4.2 as an exponential moving average. Setting $n = 0.1$ is a reasonable choice.

$$\mu = \mu + n(d_e - \mu) \quad (n \approx 0.1) \tag{4.2}$$

To define the new pheromone values of the Logical Links, r' is calculated with the formula 4.3. This indicates how strongly the pheromone values must be adapted following the new information gained from the data packet just overheard. With the current observed distance value d_e and the mean value μ, r' gives a rough estimation about the viability of d_e. The bigger r' is, the higher the influence on the pheromone entries. A reasonable choice for the constant C is $C = 3$. The value r' is limited to 0.8, which limits the maximal influence a single packet can have on the pheromone.

Only one entry of one row in the routing table can be increased per overheard packet. All other values in the same row will be reduced. To know which of the eight values in the current row of the routing table must be increased, we need to know from the packet header, which Logical Router was last passed over. The pheromone entry of the Logical Link that refers to the Logical Router the packet was received from will be increased. In the example of figure 4.6 it is the marked entry $Z^6_{2,7}$. The upper index 6 indicates the number of the Logical Link and therefore the row of the routing table. The pheromone trail a data packet marks on its path from its source to its destination refers therefore back to the source in the opposite direction from which the data packet was transmitted.

Every packet has a positive influence on the amount of pheromone of the Logical Link it is coming from and a negative one on all the others of the same routing table row. The amount of the increase depends on the Euclidean distance value of the current packet compared to the distance values of previously overheard packets. For example, a packet that traveled a longer Euclidean distance than previous packets from the same zone cannot cause big changes suddenly to the pheromone, but it is nevertheless a possible connection and has therefore a some effect on the pheromone.

$$r' = \begin{cases} \dfrac{\mu}{d_e * C} & \frac{\mu}{d_e * C} < 0.8 \\ 0.8 & \text{else} \end{cases} \tag{4.3}$$

$$r_p = (1 - Z^p_{i,j}) * (r')^2 \tag{4.4}$$

$$r_n = Z^n_{i,j} * (r')^2 \tag{4.5}$$
$$r_p = \sum r_n \tag{4.6}$$
$$1 = \sum_{n=0}^{7} Z^n_{i,j} \tag{4.7}$$
$$Z^p_{i,j} = Z^p_{i,j} + r_p \tag{4.8}$$
$$Z^n_{i,j} = Z^n_{i,j} - r_p \tag{4.9}$$

The sum of the pheromone values of a whole row in the routing table must always be one according to formula 4.7. Thus, the amount of the increased pheromone for one Logical Link must be the same as the sum of the reduction of all the other Logical Links of that zone. The outcome of this rule is formula 4.6, where r_p is the positive amount of the pheromone added to the corresponding Logical Link and r_n the negative. Using formula 4.4 to calculate the increase of the pheromone, where $Z^p_{i,j}$ is the appropriate table entry, and formula 4.5 to calculate the reduction of the other seven Logical Links, the basic requirements (formula 4.6 and 4.7) are fulfilled. It is vital to ensure that for all the seven Logical Links the r_n is calculated separately with respect to its previous value. Finally, the table entries are increased or decreased by formulas 4.8 and 4.9.

4.4.2 Redirecting a Data Packet

A packet sent through a network has an AMRA header (described in section 4.6) added in front of the header of the StPF. The main information about source and destination node is kept in the AMRA header, while in the header of the StPF only information for the current path section is stored. The values in the StPF header will change several times on its way from its source to its destination, because a data packet is sent towards an Anchor Point that is set as destination in the StPF header. When the AMRA protocol calculates a new Anchor Point towards the direction the data packet is routed, the coordinates of the new Anchor Point are written into the StPF header.

When a node receives a data packet to send it on towards the destination, it analyzes the data-packet header in order to decide if the packet can simply be redirected to the next node, or if routing-information changes in the header fields of the packet are needed.

Redirecting According to Routing-Table Information

A received data packet that shall be redirected usually has the *Target-Router* field set in its AMRA header. A set Target Router indicates the validity of the Anchor-Point position set in the StPF as destination. A Target-Router field that is not set indicates that the data packet is not sent with MABR information. This case will be discussed later in the chapter.

A node first checks if the last sender of the packet is positioned in another Logical Router than itself. This means that the packet was transmitted across the border from a neighboring Logical Router. If the current node is still in the same Logical Router as the last sender, no changes

are made in the header fields, and the packet is sent further to the next node in direction of the Anchor Point. If a router change happened, the node sets the entry of the last Logical Router index in the AMRA header to the one the packet just came from and calculates a new Target Router for the packet with the help of its routing table.

For the calculation of a new Anchor Point and a new Target Router, the row from the routing table that represents the zone where the destination of the packet is positioned is needed. From the eight entries of that row only the Logical Links with a value bigger than 0.15 are selected as possible candidates. Figure 4.7(a) shows a possible situation in a network, where the green node g has to redirect the packet. The current routing table of the green node g is shown in figure 4.7(b). The row marked represents the zone where the red destination node d is positioned. From the eight Logical Links only the ones with $n = 4, n = 5, n = 6$ have more pheromone set than 0.15 and are taken into account for further redirection decisions.

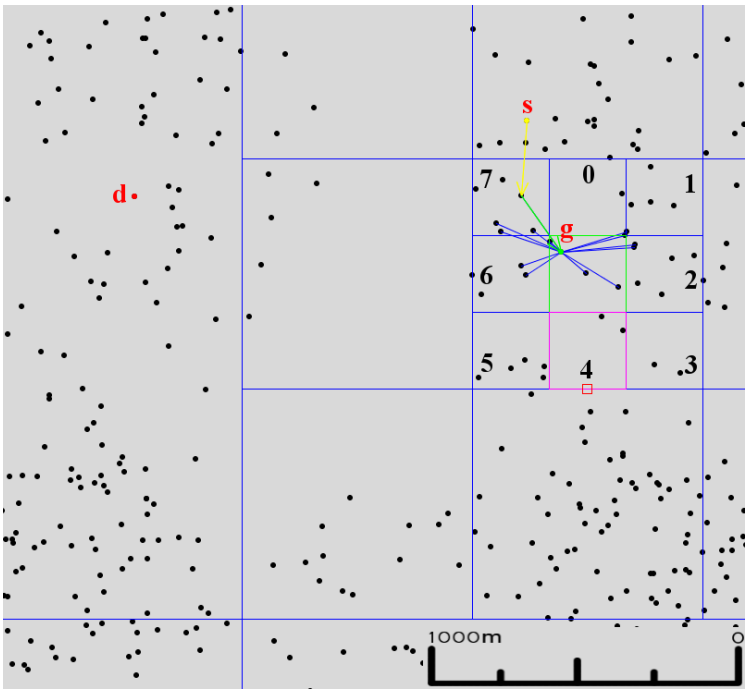
Some of the Logical Links with a pheromone value higher than 0.15 might also be rejected according to the routing rule that inhibits routing towards the direction the packet just came from. This means that there are blocked Logical Routers to which the Logical Links are suppressed. Figure 4.8 shows the blocked Logical Routers in red, while the blue dashed arrow line shows the path the packet traveled. This constraint to the routing is necessary to prevent a packet from flipping between two Logical Routers due to accidental routing tables. To prevent a loop over three Logical Routers not only the router the packet came from, but also the two neighbors in the same direction are blocked for redirection.

In the example of figure 4.7 after removing the blocked Logical Routers, (0,7,6) as the packet was received from 7, the Logical Links with $n = 4, n = 5$ are left over. When at this stage of sending possible Logical Links are left over, the node has *usable routing information* and the data packet can be sent further using MABR. From the Logical Links that remain, the node chooses one randomly based on the amount of pheromone. A Logical Link with a higher value is selected more often than one with a lower value depending on the aspect ratio. In the above example, the Logical Link $n = 4$ is chosen. The small red square in figure 4.7 shows the position of the new Anchor Point that is set as destination for the StPF protocol, and the pink square (numbered 4) marks the new Target Router that is set into the AMRA header.

Special Routing Cases

A node that receives a data packet first checks if the destination node is within transmission range and therefore reachable with only one hop. If this is the case, it sends the data packet directly to its destination by copying the destination field from the AMRA header into the destination field in the StPF header if this has not already been done by a previous node.

If a packet did not cross a Logical Router border, the Target Router information in the AMRA header remains valid and does not need to be changed. The packet can be forwarded towards the Anchor Point set in the StPF header leaving all entries the way they are.



(a) The Network with the zones

i_j	n=0	n=1	n=2	n=3	n=4	n=5	n=6	n=7	μ
1_0	1	0	0	0	0	0	0	0	1.15
1_1	0.007	0.993	0.001	0	0	0	0	0	1.666
1_2	0	0.018	0.982	0	0	0	0	0	0.847
1_3	0.032	0.032	0.776	0.032	0.032	0.032	0.032	0.032	1.694
1_4	0.001	0.001	0.001	0.001	0.881	0.064	0.051	0.001	2.081
1_5	0.001	0.001	0.027	0.001	0.029	0.357	0.584	0.001	3.082
1_6	0	0	0	0	0	0	1	0	5.536
1_7	0.547	0	0	0	0	0	0.189	0.264	1.871
2_0	0.651	0	0	0	0	0	0.349	0	4.397
2_1	0.282	0.647	0.072	0	0	0	0	0	4.131
2_2	0.005	0.835	0.146	0	0	0	0.014	0	2.959
2_3	0	0	0.011	0	0.675	0.001	0.313	0	10.6...
2_4	0	0	0.354	0	0.108	0	0.537	0	7.149
2_5	0	0	0	0	0	0.707	0.293	0	9.881
2_6	0.016	0.016	0.016	0.016	0.06	0.428	0.431	0.016	18.5...
2_7	0	0	0.1	0	0.009	0.89	0	0	34.5...
3_0	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0
3_1	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0
3_2	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0
3_3	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0
3_4	0	0	0.467	0	0.265	0.053	0.215	0	9.495
3_5	0	0	0	0	0.011	0.85	0.139	0	19.4...
3_6	0	0	0	0	0.191	0.388	0.421	0	25.1...
3_7	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0

(b) Current routing table

Figure 4.7: Example of redirecting a packet

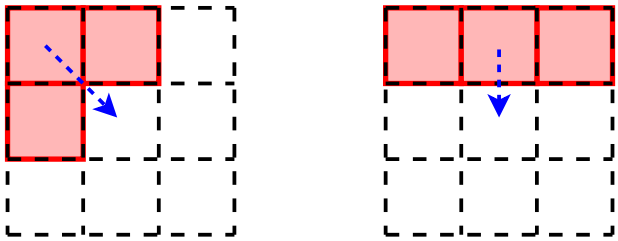


Figure 4.8: Logical Routers that are blocked for packet forwarding

If the destination node is more than just one hop away, the node compares the distance the data packet has already traveled with the distance the source node expected the data packet to travel at most. This maximum was calculated by the source node according to formula 4.10. If available, three times the mean value μ is used, else the maximum distance is limited to five times the direct Euclidean distance from source to destination d_{sd}.

Should the traveled distance be bigger than the expected one, the data packet will be sent only with the StPF protocol without using any further information the MABR might have in its routing tables. This was introduced because simulations showed that a data packet using MABR infrequently loops due to accidental routing-table entries around an area of several Logical Routers. These loops are not prevented by the forbidden Logical Routers rule, which only eliminates small loops.

$$expected\ distance = \begin{cases} 3 * \mu & \mu > 0 \\ 5 * d_{sd} & \text{else} \end{cases} \tag{4.10}$$

It happens that a node does not have usable routing information to set a new Anchor Point to redirect the data packet to, either because it has not received any information from the zone the packet should be sent to or because all the potentially good links in the routing table are blocked according to the blocked Logical Router rule. In this situation the node deletes all the routing information in the AMRA header of the data packet and sends it towards the destination node only with the StPF protocol. If later another node has usable routing information for that data packet, it sets the data packet back into MABR mode and uses the routing information from the routing tables for further redirection.

This behavior again includes the possibility of generating loops. If a node changes the packet from MABR mode into StPF mode and some hops away in another Logical Router a node changes the data packet back to MABR mode, the packet may accidently loop between these two Logical Routers even if the blocked Logical Router rule is respected. To prevent such loops, the source node sets a value called *recoverFromStPF* in the AMRA header that defines how many times the send mode of the data packet can be changed back from StPF to MABR mode on the whole path. This ensures that the loops just described will be aborted after a finite number of rounds. As an initial recoverFromStPF value the zone distance from the source to the destination node is a reasonable choice. This allows more mode changes the farther away the

source and destination nodes are, but limits them to only a few per packet.

Figure 4.9 shows an example of a node that has unusable routing information for further redirection. Seen from the node g there are two completely different paths to route to zone $Z_{3,0}$. This is due to the shape of the network, which is formed like horseshoe. One path leads to the source and one to the destination. Looking at the routing table, one can see that the node received most information from packets sent from the right side of zone $Z_{3,0}$, which forces almost all the pheromone onto the two links $Z^0_{3,0}$ and $Z^1_{3,0}$. In the current situation these two links are not allowed to redirect due to the blocked router rule explained in section 4.4.2. Therefore the node has no Logical Link to choose and it sets the packet into StPF mode to send it on. This works well in this specific example. The next node that receives the packet for redirection has usable routing information and sets the packet back to MABR mode to route it with the help of its table entries.

It is imaginable that there are situations where the packet is routed just into the wrong direction when changing it to StPF mode, which would force a loop onto the packet path. These loops are the main reason why the recoverFromStPF counter was introduced.
Figure 4.10 shows all the decisions a node makes to redirect a data packet in a flowchart. Whatever the single decisions are, the final result is always the same, either the packet is sent towards an Anchor Point with MABR routing information or no MABR routing information is used and the packet is only sent towards the destination node with the StPF.

4.4.3 Sending a Data Packet

If a node wants to send a data packet to a destination node, it first calculates several values every redirecting node between source and destination will need in order to route the packet in an efficient manner. These values are the expected Euclidean distance the packet is allowed to have and the recoverFromStPF value that must be set. These two values are calculated only once per data packet by the node that generates them. Decrementing the recoverFromStPF value is the only change redirecting nodes are allowed to accomplish on these two entries in the AMRA packet header.

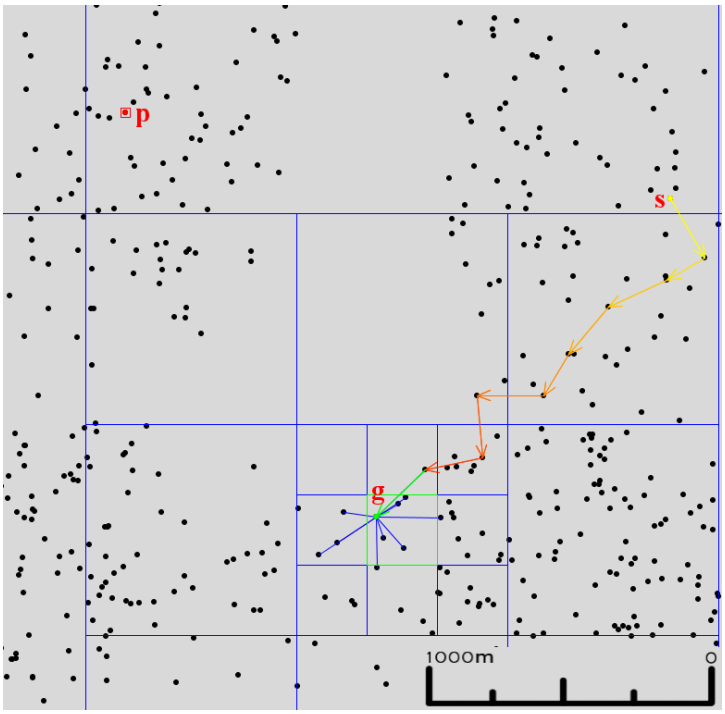
Transmitting the packet for the first time to another node operates similarly to redirecting, except that there are no blocked Logical Routers and no target zones set in the packet header.

Data packets are sent with additional information for the GFG/GPSR protocol used as StPF. The packets are marked with a flag in the packet header if GFG/GPSR must route the packet with left or right-hand rule if the backup mode is needed. A packet is sent randomly either with right or left-hand rule with the same probability.

The advantage of this behavior is that different routes are tried if the routing with the information in the routing tables fails.

4.4.4 Balancing out of Pheromone While Moving

When a node moves, the routing table it stores becomes imprecise. The information was collected at a different position in the network. Especially the information to close zones may



(a) The Network with the zones

i_j	n=0	n=1	n=2	n=3	n=4	n=5	n=6	n=7	μ
1_0	0.344	0.094	0.094	0.094	0.094	0.094	0.094	0.094	0.957
1_1	0.502	0.071	0.071	0.071	0.071	0.071	0.071	0.071	1.472
1_2	0.055	0.013	0.867	0.013	0.013	0.013	0.013	0.013	1.306
1_3	0.044	0.003	0.039	0.641	0.263	0.003	0.003	0.003	2.154
1_4	0.01	0.003	0.003	0.376	0.6	0.003	0.003	0.003	2.653
1_5	0.034	0.034	0.034	0.034	0.034	0.034	0.763	0.034	1.574
1_6	0.036	0.036	0.036	0.036	0.036	0.036	0.749	0.036	0.784
1_7	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0
2_0	0.064	0.544	0.064	0.064	0.064	0.064	0.069	0.064	10.5...
2_1	0.206	0.489	0.298	0	0.006	0	0	0	5.884
2_2	0.499	0.111	0.234	0	0.025	0	0.131	0	4.429
2_3	0.018	0	0.642	0.083	0.017	0	0.239	0	10.1...
2_4	0.03	0.03	0.231	0.03	0.109	0.03	0.51	0.03	2.789
2_5	0.048	0.048	0.048	0.048	0.455	0.048	0.258	0.048	4.108
2_6	0	0	0	0	0.001	0	0.999	0	12.6...
2_7	0	0	0	0	0.03	0	0.969	0	26.6...
3_0	0.719	0.197	0.004	0	0.009	0	0.071	0	15.0...
3_1	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0
3_2	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0
3_3	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0
3_4	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0
3_5	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0
3_6	0	0	0	0	0.627	0	0.373	0	9.573
3_7	0.003	0.003	0.003	0.003	0.024	0.003	0.957	0.003	36.16

(b) Current routing table

Figure 4.9: Example of node with unusable routing information

Figure 4.10: Flowchart for redirecting of a data packet

become inaccurate very quickly. Thus, the correct redirecting of a packet may be harmed. Entries to zones farther away are more stable and might still be correct for routing packets. Many bad entries in the routing table enhance the possibility of loops and generally diminish routing efficiency.

To prevent malicious routing information possibly due to node movement, all the pheromone entries are balanced out towards an equal value of 0.125 for all the eight Logical Links per zone. Every time a node crosses the border from one Logical Router to another all the values in the routing table are recalculated according to formula 4.11. The farther away a zone is positioned from a node, the smaller the change of the pheromone values. The change to close-zone values balanced out towards 0.125. Every time a node crosses the border from one Logical Router to another all the values in the routing table are recalculated according to formula 4.11. The farther away a zone is positioned from a node, the smaller the change of the pheromone values. The change to close-zone values is higher. This is ensured by using the zone distance i as exponent in the numerator. For the constant K a value of $K = 1$ is a reasonable choice. The higher the chosen value of K, the faster the pheromone values are balanced out towards 0.125. With formula 4.11 it is assured that the condition of formula 4.7 is fulfilled.

$$Z_{i,j}^{k} = Z_{i,j}^{k} + (0.125 - Z_{i,j}^{k}) * \frac{K}{3^{i}} \tag{4.11}$$

It is obvious that in many cases pheromone is taken away from links that are still correct. But this is not as bad as having favoring the wrong links due to malicious entries.

Figure 4.11 shows the routing table just before and just after the node moved form one Logical Router to another. The marked fields show how the distance of a zone has an influence on the amount of the adapted pheromone.

A disadvantage is the mean value μ that also becomes imprecise when the node moves. This value cannot be adapted or corrected because the moving node does not know in which direction the value should be changed. This is not too hurtful to the whole system because the value adapts very fast to newly measured distances according to formula 4.2, and the value itself is not used directly to redirect data packages. The only effect it may have is that an arriving data packet gets a wrong weight to change the pheromone values. This is done according to the value r' (formula 4.3) that is influenced by the mean value μ.

4.5 Ants

Ants are empty packets that are sent through the network in order to find new and better paths through the network and mark them with pheromone. They have the same AMRA header as data packets. Additionally a flag marks them as ants.

i , j	n=0	n=1	n=2	n=3	n=4	n=5	n=6	n=7	μ
1 , 0	0.702	0.043	0.043	0.043	0.043	0.043	0.043	0.043	1.356
1 , 1	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0
1 , 2	0.061	0.061	0.576	0.061	0.061	0.061	0.061	0.061	0.906
1 , 3	0.053	0.053	0.053	0.631	0.053	0.053	0.053	0.053	0.957
1 , 4	0.032	0.032	0.032	0.032	0.777	0.032	0.032	0.032	0.71
1 , 5	0.003	0.003	0.003	0.003	0.003	0.978	0.003	0.003	1.183
1 , 6	0.017	0.017	0.017	0.017	0.017	0.119	0.777	0.017	1.693
1 , 7	0.052	0.014	0.014	0.014	0.014	0.014	0.014	0.863	1.968
2 , 0	0.952	0.001	0.001	0.001	0.001	0.001	0.001	0.043	4.589
2 , 1	0.118	0.003	0.832	0.034	0.003	0.003	0.003	0.003	4.84
2 , 2	0	0	0.973	0.026	0	0	0	0	3.329
2 , 3	0	0	0.136	0.313	0.551	0	0	0	4.665
2 , 4	0	0	0	0.01	0.97	0.02	0	0	3.354
2 , 5	0	0	0	0	0.965	0.025	0.01	0	5.784
2 , 6	0	0	0	0	0.606	0.078	0.304	0.011	3.808
2 , 7	0.278	0	0	0	0.005	0.515	0	0.202	6.937
3 , 0	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0
3 , 1	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0
3 , 2	0	0	0.943	0.042	0.014	0	0	0	6.519
3 , 3	0.048	0.048	0.048	0.533	0.177	0.048	0.048	0.048	7.778
3 , 4	0.001	0.001	0.001	0.023	0.873	0.098	0.001	0.001	6.434
3 , 5	0.064	0.064	0.064	0.064	0.236	0.064	0.378	0.064	9.632
3 , 6	0	0	0	0	0.34	0.658	0	0	7.265
3 , 7	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0

(a) Before movement

i , j	n=0	n=1	n=2	n=3	n=4	n=5	n=6	n=7	μ
1 , 0	0.51	0.07	0.07	0.07	0.07	0.07	0.07	0.07	1.356
1 , 1	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0
1 , 2	0.082	0.082	0.426	0.082	0.082	0.082	0.082	0.082	0.906
1 , 3	0.077	0.077	0.077	0.462	0.077	0.077	0.077	0.077	0.957
1 , 4	0.063	0.063	0.063	0.063	0.56	0.063	0.063	0.063	0.71
1 , 5	0.044	0.044	0.044	0.044	0.044	0.694	0.044	0.044	1.183
1 , 6	0.053	0.053	0.053	0.053	0.053	0.121	0.559	0.053	1.693
1 , 7	0.076	0.051	0.051	0.051	0.051	0.051	0.051	0.617	1.968
2 , 0	0.86	0.015	0.015	0.015	0.015	0.015	0.015	0.052	4.589
2 , 1	0.119	0.017	0.753	0.044	0.017	0.017	0.017	0.017	4.84
2 , 2	0.014	0.014	0.879	0.037	0.014	0.014	0.014	0.014	3.329
2 , 3	0.014	0.014	0.135	0.292	0.504	0.014	0.014	0.014	4.665
2 , 4	0.014	0.014	0.014	0.023	0.876	0.031	0.014	0.014	3.354
2 , 5	0.014	0.014	0.014	0.014	0.871	0.036	0.023	0.014	5.784
2 , 6	0.014	0.014	0.014	0.014	0.553	0.084	0.285	0.024	3.808
2 , 7	0.261	0.014	0.014	0.014	0.018	0.471	0.014	0.194	6.937
3 , 0	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0
3 , 1	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0
3 , 2	0.005	0.005	0.913	0.045	0.019	0.005	0.005	0.005	6.519
3 , 3	0.051	0.051	0.051	0.518	0.175	0.051	0.051	0.051	7.778
3 , 4	0.006	0.006	0.006	0.026	0.846	0.099	0.006	0.006	6.434
3 , 5	0.067	0.067	0.067	0.067	0.232	0.067	0.369	0.067	9.632
3 , 6	0.005	0.005	0.005	0.005	0.332	0.638	0.005	0.005	7.265
3 , 7	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0

(b) After movement

Figure 4.11: Balancing out of pheromone if a node crosses a Logical Router border. Close-distant-zone values are changed more than far-distant-zone values.

Ants are routed by the routing rules of the StPF. Thus, no routing tables are used. The reason for not using the routing tables is that new paths can only be found if the ant tries to route in directions where no pheromone is set yet. To maintain the existing paths the data packets are used to reinforce the pheromone of the Logical Links.

Nodes decide when to send the next ant themselves. In this thesis, a node sends an ant every x seconds with a slight jitter to a random position in the network. Other sending rules are possible, such as sending an ant if the node did not hear any traffic for a certain time.

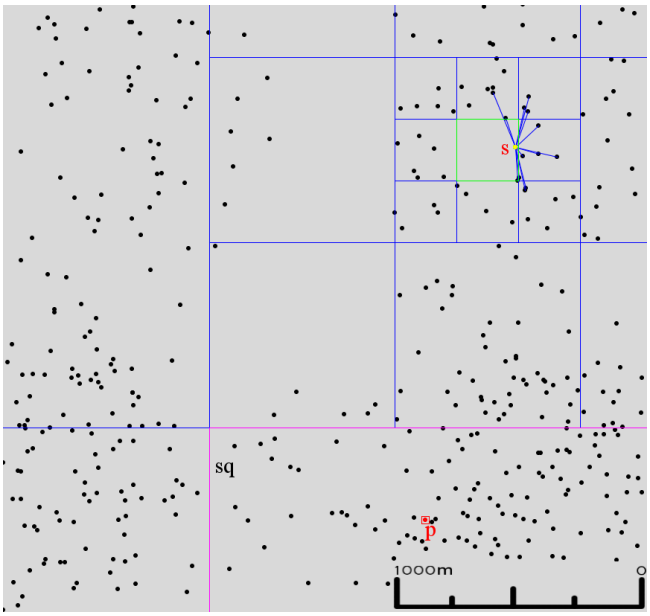
To generate a new ant, the node first chooses a random Logical Router to which it wants to send the ant. The geographical center of the Logical Router is used as the destination in the StPF header of the ant packet. It is possible that the Logical Router the ant should be sent to is not reachable at all because there are no nodes in and around it. For this case, the generating node uses the Target Router fields in the AMRA header to store a *Target Zone* information. This zone has the same boundaries as the zone the selected ant destination is positioned in from the point of view of the source node. When an ant is on its way to its destination, it is sent with a simple greedy algorithm as soon as the ant arrives in the Target Zone as close as possible to the destination position. The node that has no closer neighbor to the destination position finally drops the ant in order to avoid infinitely looping ants.

Figure 4.12(a) shows the Target Zone copied into the AMRA header of the ant as a pink

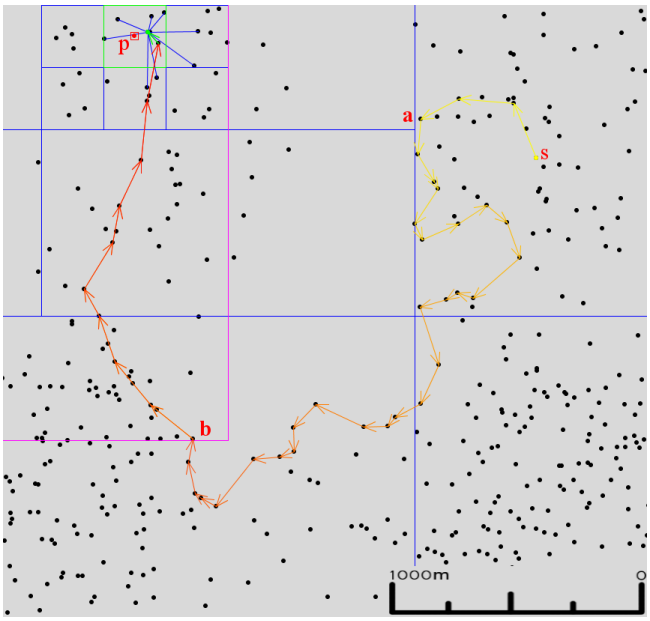
square sq before it is sent on its way from the yellow source node s to the red destination position p.

An example path an ant takes is shown in figure 4.12(b). GFG/GPSR is used for the StPF protocol. At node a the GFG/GPSR switched into perimeter mode and at node b the send mode for the ant is set to greedy forwarding because it entered the Target Zone.

To benefit better from the ants, the GFG/GPSR protocol is changed a little bit. The source node of the ant can determine if the perimeter mode of the GFG/GPSR should work with right or with left-hand rule. If two ants are sent from the same Logical Router to the same position but one with right and one with left-hand rule, the nodes close to the destination position will then put more pheromone to the shorter path and therefore route data packets toward the zone from which the ants came more efficiently. In this thesis ants are always sent with equal probability in right and left-hand mode. The difference between right and left-hand ants is shown in figure 4.13 where two ants are sent from the same source s to the same Logical Router as destination for the ants. One ant sent with right and one sent with left hand rule, the paths the two ants take differ completely.

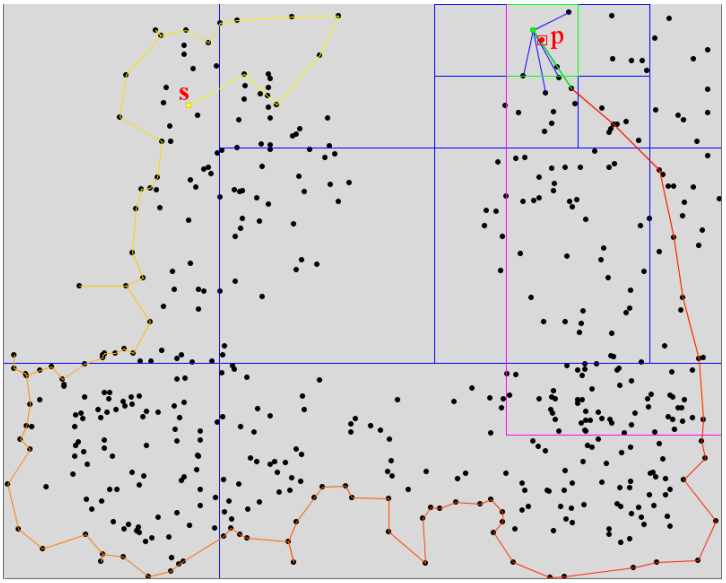


(a) Starting an ant

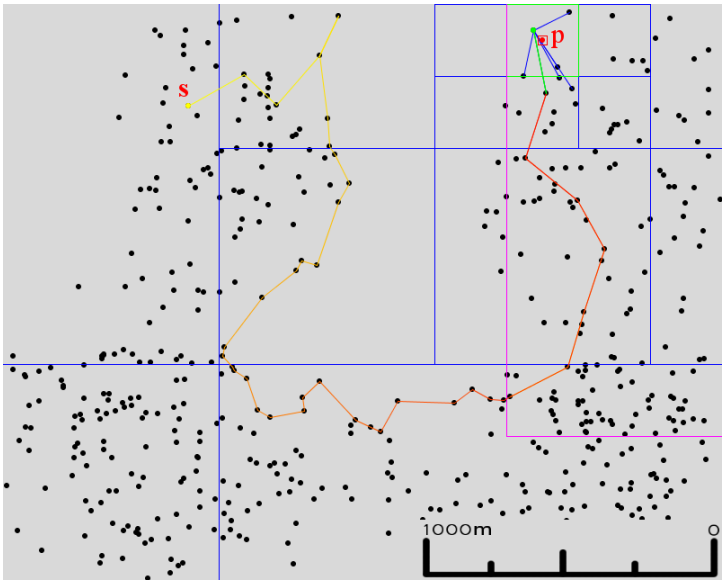


(b) Path of a sent ant

Figure 4.12: Sending ants



(a) Right-hand ant



(b) Left-hand ant

Figure 4.13: Right and left-hand-rule ants

4.6 The AMRA Packet Header

The AMRA protocol needs several additional values in the packet header to work properly. The length of the header depends on the precision of the position information that is needed.

Table 4.1 shows the header fields of the AMRA protocol additionally required. This header is put in front of the header of the used StPF protocol.

Table 4.1: The necessary fields in the AMRA header

sourceNodeID	the unique address of the source node
destinationNodeID	the unique address of the deatination node
seqNr	the sequence number of the packet
sourceNodePosX	the x position of the source node
sourceNodePosY	the y position of the source node
destinationNodePosX	the x position of the destination node
destinationNodePosY	the y position of the destination node
lastLRIndex	the index of the last Logical Router
euclDist	the distance coverd by the packet so far
expectedDistance	the expected distance the packet will cover
hops	number of hops done by the packet so far
packetMode	the kind of traffic (ant, data, lefhand, righthand)
targetZonePosX	the x position of the target zone
targetZonePosY	the y position of the target zone
targetZoneLength	the length of the target zone
targetZoneWidth	the width of the target zone
recoverFromStPF	how many times this packet may switch back form StPF mode to AMRA mode

4.7 Conclusions

The architecture is built as single modules. It is possible to change one part almost without any effects to the other ones. For instance one can use a different way of sending ants or use a different algorithm for the StPF. Furthermore, it is imaginable to change the proposed topology in the TAP protocol where an obvious change could be to have the single Logical Routers formed as hexagons, as is usual in cellular network topologies. Instead, in this thesis regular squares are used for simplicity reasons.

The exchange of the StPF would require minor changes to the MABR because some optimizations are built under the assumption that GFG/GPSR is used as protocol for StPF. For instance sending ants as well as data packets with right or left-hand rule. Also instead of MABR another routing protocol could be used such as AODV.

For the mean value μ in the routing table, use of the Euclidean distance as calculation base is proposed, this mainly because of the limitations of the used network simulator explained in

section 6 and used for this thesis. An interesting alternative to use for the μ calculations would be the end-to-end delay the packets had so far. The advantage would be that special situations such as overloaded network parts would also be taken into account to determine the quality of a link.

Chapter 5

Mobility Models

The goal of a mobility model is to approximate the movements of simulated mobile nodes as accurately as possible to the real movements of the mobile nodes. The choice of the mobility model can have a major influence on the test results.

The mobility models can be divided into two major groups, the entity-mobility models and the group-mobility models. In the group-mobility model the movement of a single node depend also from the movements of its neighbors whereas in the entity mobility model the movements of a mobile node are completely independent from each other.

5.1 Entity Mobility Models

In the entity mobility model every node moves absolutely independently from all the other nodes in the mobile network. Only the general parameters of the model influence the movements of a single node.

5.1.1 Random-Waypoint Mobility Model

A mobile node in the *Random-Waypoint Mobility Model* gets a random target position it must move to at a random speed. Once the target position is reached, the node pauses for a certain time (pause time) before it gets a new target position and speed. The speed a node gets is uniformly distributed between $[minspeed, maxspeed]$. For a closer look on speed consideration see [23]. The nodes distribution is more dense at the center of the simulation area than towards the borders no matter how the nodes were distributed at the beginning of the simulation (uniform, random, grid) [24]. The distribution can have a major influence on the simulation results [25].

Figure 5.1 shows the movements of two mobile nodes in a Random-Waypoint Mobility Model. The circles are the start and end position of the nodes and the dots show where the nodes spent the pause time before they moved towards a new random position.

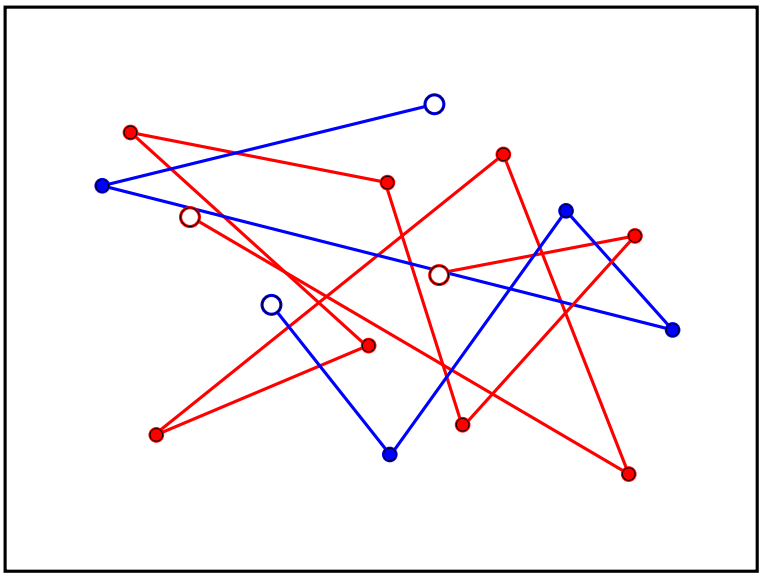


Figure 5.1: Random-Waypoint Mobility

5.1.2 Random-Direction Mobility Model

In the *Random-Direction Mobility Model* a node gets a random direction in which the node moves until it reaches the simulation area boundary [26]. There it pauses for a certain period of time before it gets a new random direction in which to travel.
Compared to the Random-Waypoint Mobility Model, the density difference of the nodes over the simulation area is much smaller. Because the nodes move until they reach the border and pause there, the node movements are rather uncommon if the mobility model is meant to simulate realistic behavior of mobile nodes. The more evenly distributed nodes cause a higher hop count of the data packets in simulations as the average distance from one hop to another is longer.

An example of two nodes moving according to this Mobility Model is showing figure 5.2. The circles are the start and end positions, and the dots show where the nodes reached the simulation boundary and therefore paused before they moved on into another random direction.

5.1.3 Restricted-Random-Waypoint Mobility Model

The main goal of the Restricted-Random-Waypoint Mobility Model introduced in [27] is to have more realistic node movements in large-scale ad-hoc networks. In large simulation areas if nodes are assumed to be small personal devices it is not usual for a node to select a random position in the whole large network every time as they would do with Random Waypoint Mobility. More likely, the node will make several short movements within one town before it moves a longer distance, for instance to another town.

To simulate this ”natural” behavior of the mobile node, *towns* and *highways* that connect the

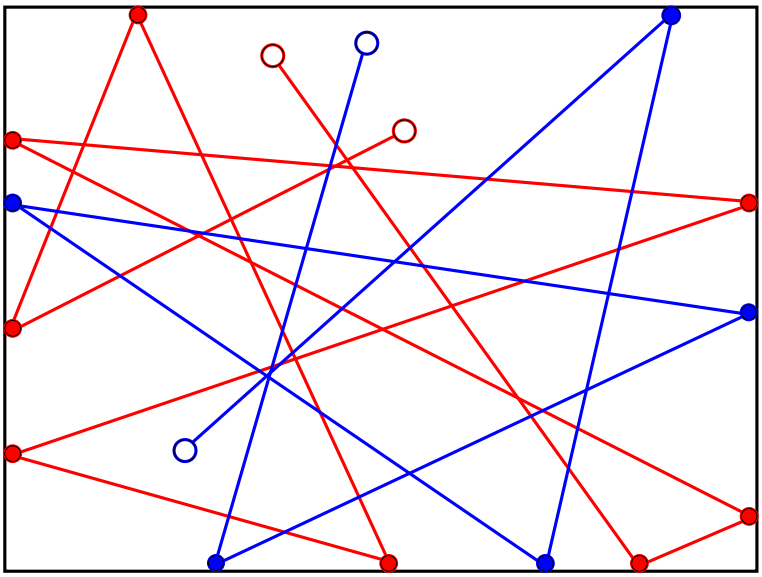


Figure 5.2: Random-Direction Mobility

towns are introduced in the Restricted-Random-Waypoint Mobility. Nodes move within a town with the rules of Random-Waypoint-Mobility Model, but with a certain probability between $[0, 1]$ they may choose a position in another town connected by a highway.

The nodes are therefore not at all uniformly distributed over the whole simulation area; on the contrary, there are areas with no nodes at all and areas (the towns) with a very high density of nodes. To have an even more realistic behavior, the speed of the nodes can be different for movements within a town or for movements on highways.

Figure 5.3 shows two nodes moving in a Restricted-Random-Waypoint Mobility Model with node a having a higher probability of staying in the same town than node b, which changes towns more often. Nodes that have a high probability of changing the town are also called *commuters*. In simulations they are needed to ensure a certain connectivity of the network so that there are always enough nodes on the highways to prevent the net from splitting up into smaller parts.

5.2 Group Mobility Models

Differently from the entity mobility models, the group mobility models try to simulate the movements of nodes grouped together. Examples would be people travelling in trains or soldiers as a part of military troops with the same order. The movements of a single node is no longer independent from the movements of other nodes.

The only model presented here is the *Reference-Point-Group Mobility Model* because it is a very flexible one and because group mobility models are not used in the simulations for this thesis.

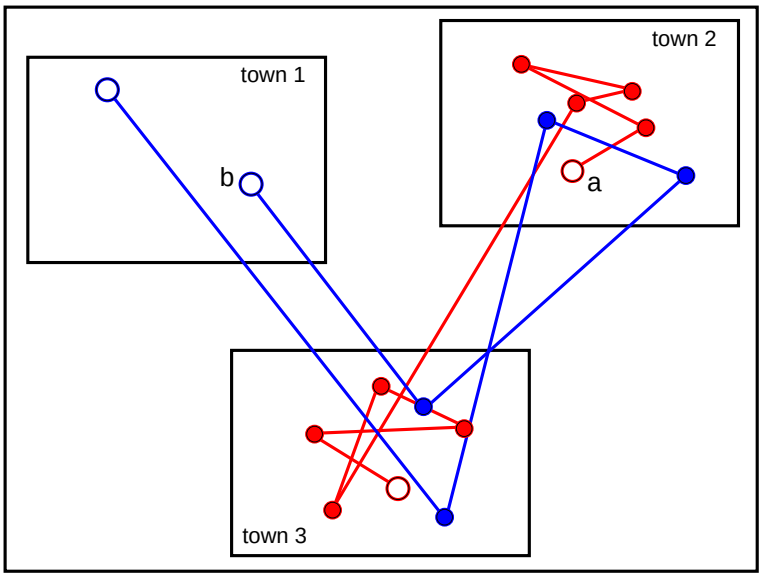


Figure 5.3: Restricted-Random-Waypoint Mobility

5.2.1 Reference-Point-Group Mobility Model (RPGM)

In this mobility model presented by [28], the movement of a mobile node depends on two vectors: one for the group it is a part of and one for the individual node movements, usually smaller than the group movement. The vector for the group motion $\overrightarrow{GM}$ is calculated for a logical center of the group. Every mobile node that is a member of the same group has a reference point R_p relative to the logical group center.

If a group gets a new group motion vector, an individual node vector $\overrightarrow{RM}$ (random move) is also applied to every node of the group. Additionally the node must move to the new position relative to its reference point within the group defined by $\overrightarrow{RM}$. The group vectors are set after the rules of Random-Waypoint Mobility Model.

In figure 5.4 the movement of a group with three nodes is shown. The circles are the positions the individual nodes have, the dots are the reference points $R_p(t)$ at a specific time t belonging to the nodes. $\overrightarrow{GM}$ is the group motion vector and $\overrightarrow{RM}$ are the random motion vectors of the individual nodes.

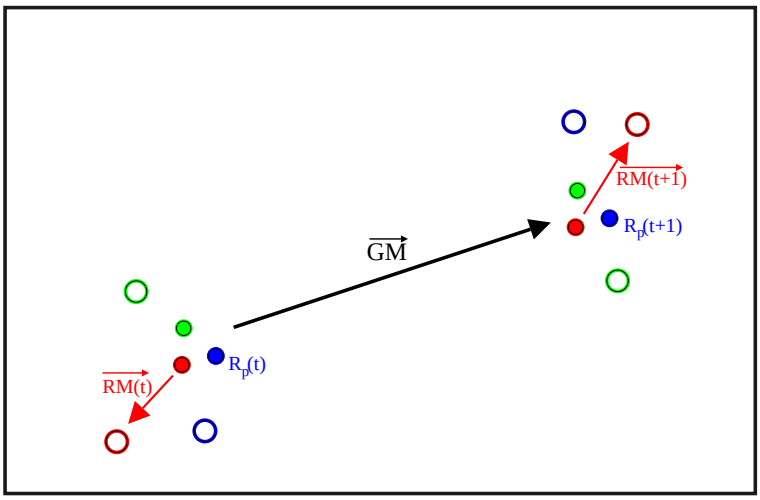


Figure 5.4: Reference-Point-Group Mobility Model

Chapter 6

Simulation Environment

In this thesis, a mobile ad-hoc net-routing protocol designed for large scale networks with several thousand participants is discussed. Today's network simulators such as Qualnet or NS2 cannot handle that amount of nodes in a simulation with a complex routing protocol. These network simulators work as discrete event simulators; they break the simulation time into small time slices. Every node eventually generates an event if it is involved in any network action. The more nodes are simulated, the more resources are needed to run a simulation. The required resources often increase at least quadratic to the increase of additional simulated mobile nodes.

To simulate the AMRA protocol in an adequate environment, an existing simple java simulator that only works on static networks is used. This basic simulator was written to evaluate the GOAFR+ protocol described in 2.5. It is used to compare the routing path taken by different protocols in a static network. To be able to evaluate the AMRA protocol this basic simulator had to be enlarged with node mobility, because the protocol is to be tested in an environment of mobile nodes.

This simulator can handle a large amount of nodes because it does away with many time-consuming tasks. For instance sending a packet from one node to another is not simulated with a MAC protocol such as 802.11. All the processes of lower network layers are not simulated. A packet is given to the recipient node if they are within the transmission range of each other without any further calculations. Node mobility happens only between two data-packet sendings; while a packet is routed from a source to a destination, the network stays static.

All the tasks in the simulator are done one after the other, never at the same time. Simulation events are therefore written into a simulation stack and then proceeded in the given sequence, an example stack is shown in table 6.1. There is no real time implementation in this simulator, the speed of nodes depends on the number of calls of the *move hosts* method that moves host according to a set distance every time it is called. The speed of nodes as well as the number of sent packets are controlled by setting the number of calls of the *send procedure* into relation with the other events.

As an example, a simulation that lasts for 900 seconds might be devided in 900 parts, one for every second. Once per part the nodes are moved, and the number of sent data packets between

Table 6.1: Simulation stack

send packet from node 1 to node 2
send packet from node 2 to node 1
send packet from node 3 to node 4
move hosts
send packet from node 1 to node 2
send packet from node 2 to node 1
send packet from node 3 to node 4
move hosts

node-move procedure calls defines the amount of data traffic per second. Similarly to the time scale, distances are not set absolutely but in relation to the transmission range.
Because tasks are processed one after the other, overloaded nodes cannot occure. And due to the lack of a time scale, packets arrive at the destination node with no delay at all. Simulations with measurements of packet delays are therefore not possible with this simulator. Packet delivery is only measureable according to hops and according to the covered Euclidean distance if took a packet to be routed from source to destination.

6.1 The Graphical User Interface

The simulator contains an optional *graphical user interface* (GUI) that is shown in figure 6.1(a). This GUI is helpful to visually control the routing path of a data packet. An implementation of a new algorithm is quickly checked as to its correctness, and the behaviour of implemented algorithms is easier to understand. For simulations of an algorithm, the GUI is not used. It is only meant as a help during development.

A packet can be forwarded hop by hop or also at once over the hole path from source to destination node. To visually check the behavior of a table-driven routing algorithm, a node can be directly relocated manually or all the nodes can be moved according to the used mobility model at once. To be able to train a table-driven algorithm, several messages can be sent through the network between random positions.

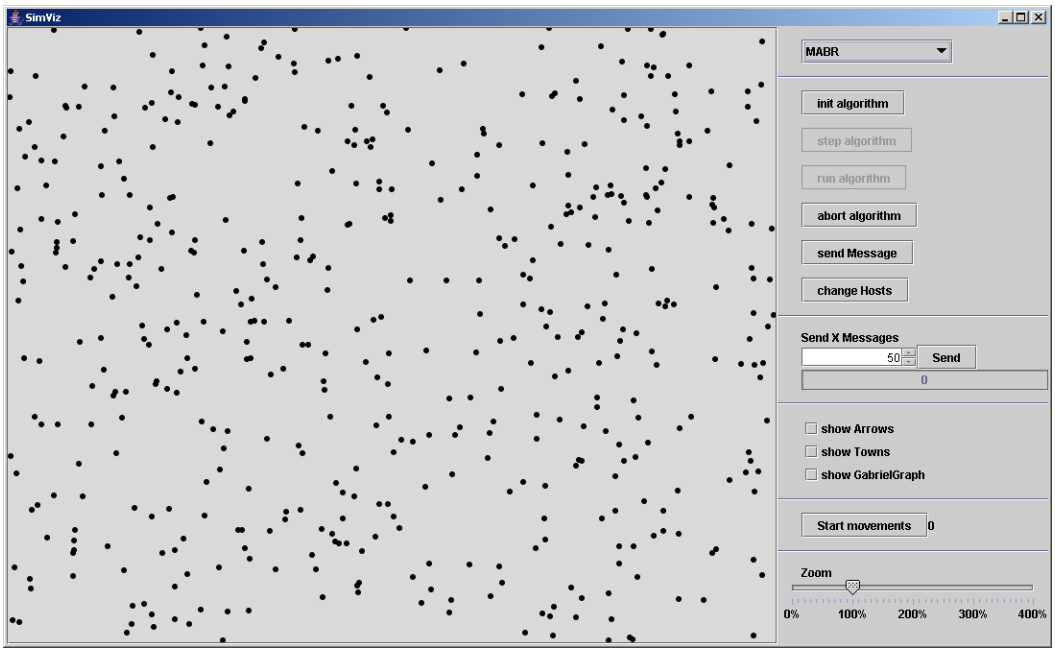
For the AMRA protocol a separate GUI add-on is implemented that shows the current routing table of the selected node. This part of the GUI is shown in figure 6.1(b).

6.2 Implemented Routing Algorithms

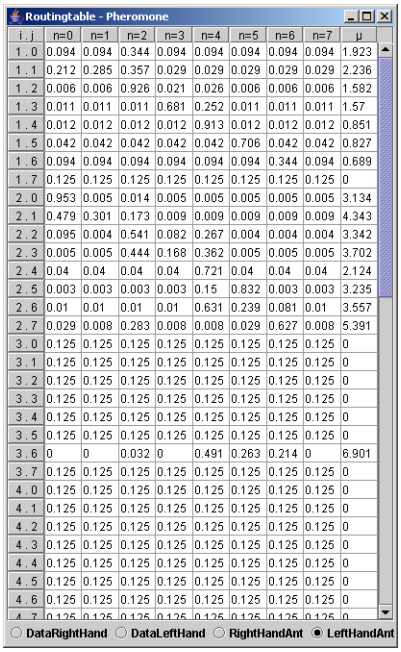
Table 6.2 lists all the algorithms which are implemented in the network simulator so far.

Table 6.2: Implemented Routing Algorithms

GFG/GPSR	introduced in section 2.4
AMRA	introduced in section 4
Greedy	introduced in section 2.2
Face Routing	A routing algorithm that routes exclusively along paths of a planar graph[9]
Adaptive Face Routing (AFR)	Another algorithm that routes along faces[29]
Greedy Other Adaptive Face Routing (GOAFR)	A combination of Greedy and AFR[30]
GAFR$^+$	An improvement of GOAFR, introduced in section 2.5
Shortest Path Hops	Not a real routing algorithm, calculates path with fewest hops from source to destination in the current situation
Shortest Path Euclidean	Not a real routing algorithm, calculates path with shortest Euclidean distance from source to destination in the current situation



(a) GUI of the simulator



i , j	n=0	n=1	n=2	n=3	n=4	n=5	n=6	n=7	μ
1 , 0	0.094	0.094	0.344	0.094	0.094	0.094	0.094	0.094	1.923
1 , 1	0.212	0.285	0.357	0.029	0.029	0.029	0.029	0.029	2.236
1 , 2	0.006	0.006	0.926	0.021	0.026	0.006	0.006	0.006	1.582
1 , 3	0.011	0.011	0.011	0.681	0.252	0.011	0.011	0.011	1.57
1 , 4	0.012	0.012	0.012	0.012	0.913	0.012	0.012	0.012	0.851
1 , 5	0.042	0.042	0.042	0.042	0.042	0.706	0.042	0.042	0.827
1 , 6	0.094	0.094	0.094	0.094	0.094	0.094	0.344	0.094	0.689
1 , 7	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0
2 , 0	0.953	0.005	0.014	0.005	0.005	0.005	0.005	0.005	3.134
2 , 1	0.479	0.301	0.173	0.009	0.009	0.009	0.009	0.009	4.343
2 , 2	0.095	0.004	0.541	0.082	0.267	0.004	0.004	0.004	3.342
2 , 3	0.005	0.005	0.444	0.168	0.362	0.005	0.005	0.005	3.702
2 , 4	0.04	0.04	0.04	0.04	0.721	0.04	0.04	0.04	2.124
2 , 5	0.003	0.003	0.003	0.003	0.15	0.832	0.003	0.003	3.235
2 , 6	0.01	0.01	0.01	0.01	0.631	0.239	0.081	0.01	3.557
2 , 7	0.029	0.008	0.283	0.008	0.008	0.029	0.627	0.008	5.391
3 , 0	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0
3 , 1	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0
3 , 2	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0
3 , 3	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0
3 , 4	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0
3 , 5	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0
3 , 6	0	0	0.032	0	0.491	0.263	0.214	0	6.901
3 , 7	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0
4 , 0	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0
4 , 1	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0
4 , 2	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0
4 , 3	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0
4 , 4	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0
4 , 5	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0
4 , 6	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0
4 , 7	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0

(b) AMRA GUI add-on

Figure 6.1: GUIs

Chapter 7

Simulation Results

The main focus of the simulations is laid on the comparison of the number of hops a packet needs to be routed from the source to the destination node. The fewer hops needed, the better the rating of the taken path.

To evaluate the performance gain if using AMRA, the results of sending a packet only with GFG/GPSR and sending packet with AMRA and GFG/GPSR as underlying algorithm are compared. The packets with and without AMRA are sent in exactly the same network situation from the same node to the same destination.

7.1 Simulation Scenarios

To evaluate the performance of the whole AMRA protocol, the simulations are done in three different network scenarios. The main scenario is a topology with 500 nodes distributed over 4 towns connected with highways as shown in figure 7.1. In this scenario, the effects of several parameters on the protocol efficiency are tested. With the parameters won out of these simulations, the behavior of the protocol is evaluated in the other two scenarios, to check if AMRA performs similarly. One of the two control scenarios is very simple, with 200 nodes in only one town as shown in figure 7.2, complex (figure 7.3) with 19 towns and $10'000$ nodes participating in the network.

For all the simulations, the Restricted-Random-Waypoint Mobility Model explained in section 5.1.3 is used. In the scenario with only one town, the Restricted-Random-Waypoint Mobility Model is reduced to a Random-Waypoint Mobility Model of section 5.1.1 because nodes do not have the possibility of changing the towns.

Due to the characteristics of the simulator used (section 6), movements of nodes are done in discrete steps once every second. During the sending of data and ant packets, the nodes are not moving at all. The knowledge of the neighbor nodes positions is an assumption.

Most of the simulations are done with unidirectional and bidirectional traffic. Unidirectional

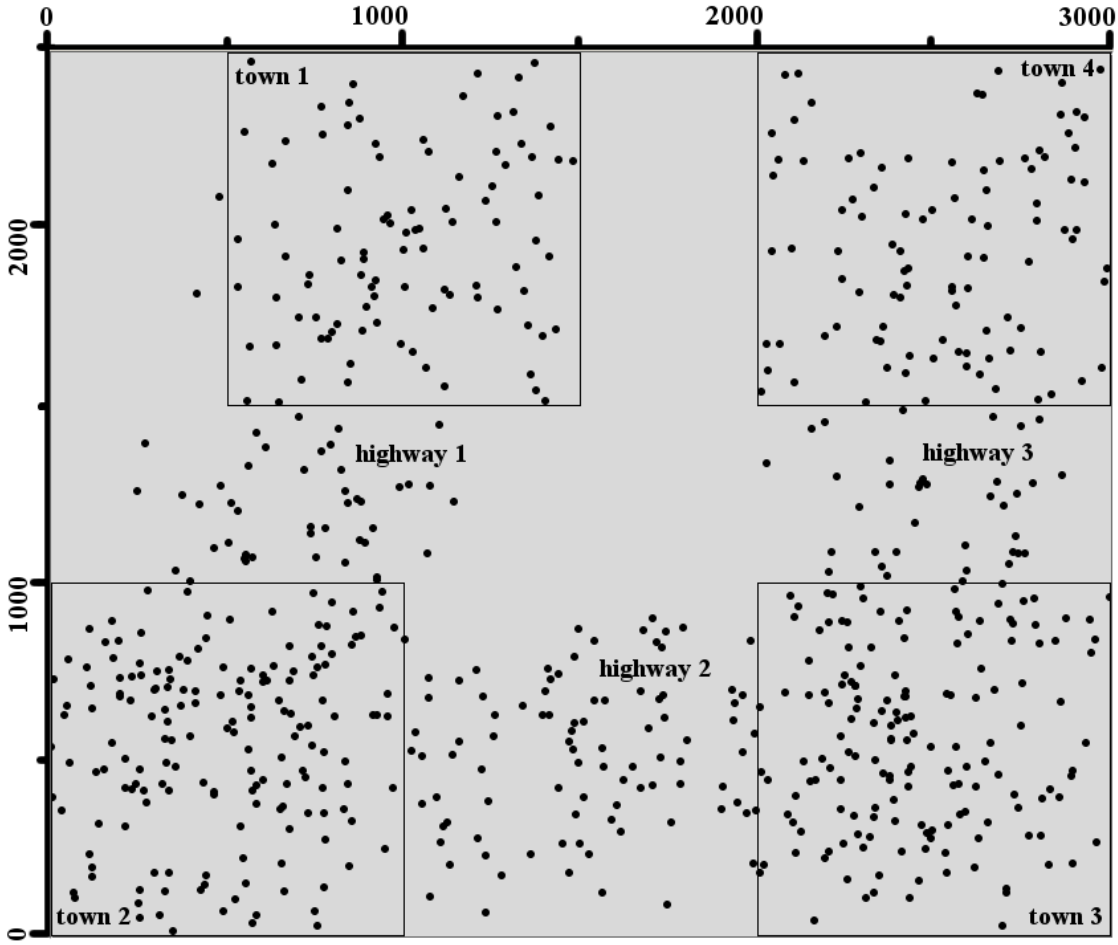


Figure 7.1: Main simulation scenario

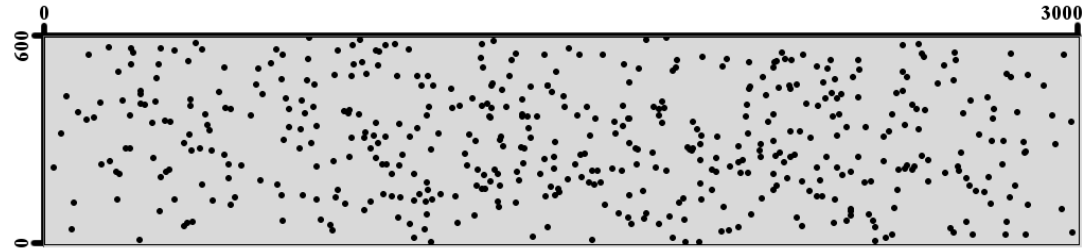


Figure 7.2: Simple scenario

traffic is generated at source nodes; all of them choose a random destination node. Data packets are only sent in one direction and one packet every second per source. The chosen destination node stays the same for the whole simulation.

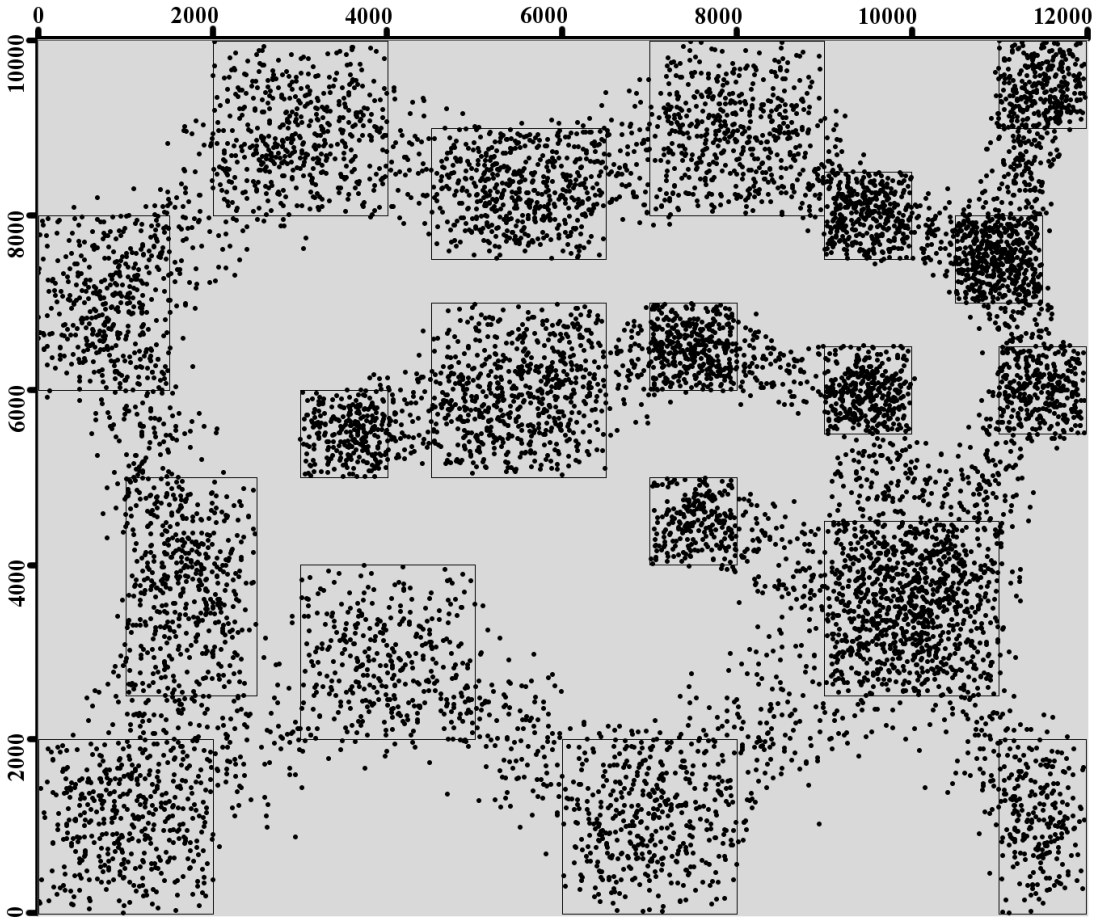


Figure 7.3: Complex scenario

If the simulation is done with bidirectional traffic, every source is also the destination for the data packets of its communication peer. Every second a data packet is sent in both directions.

The simulation time is always 1800 seconds. The first 900 seconds are used to level off the node movements of the used mobility model; no packets are sent during that time. This is done so that the network has approximately the same structure while the measurements about data packets routing are done in the second 900 seconds of the simulation, which helps avoiding problems that might occur if we start simulations with a *steady-state* distribution [23]. Such problems occur for instance if all the nodes of the simulation start with a pause time.

In all the simulations, the following default parameters are used if not marked as being set differently:

- Node speed within towns $= 1 - 15m/s$

- Node speed on highways $= 10 - 30m/s$
- Transmission range $= 250m$
- Pause time commuters $= 1s$
- Pause time normal nodes $= 120s$
- Number of ants per second $= 50$
- Logical Router size $= 250m$
- Number of sources $= 10$
- Constant C in formula 4.3 $= 3$

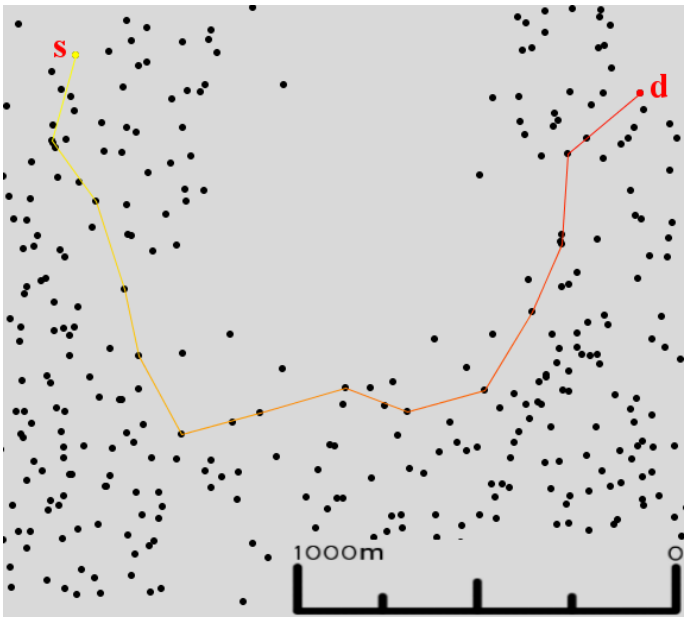
The absolute number of hops a packet needs from source to destination depends heavily on the position of the source and the destination node in the network. To be able to compare the measured hops among each other, they are put in relation to the best possible path (*Shortest Path*), as a lower theoretical bound, from source to destination when the measured packet was sent. With the help of this factor it is possible to compare the obtained results directly even if the path has a different length. The same processing is done with the results with regard to the Euclidean distance a packet traveled, but are then compared with the Shortest Path in Euclidean sense. The shortest path taken with regard to hop counts is not imperatively the same path as the one taken with shortest path in the Euclidean sense as shown in figure 7.4 where the path taken from source s to destination d differs slightly.

7.2 Main Simulation Scenario

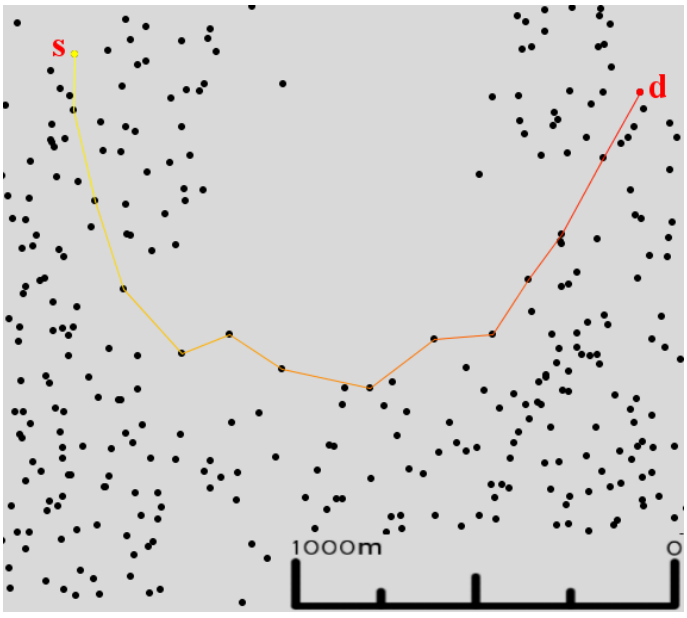
The scenario of figure 7.1 is selected because in this topology network traffic can be generated that is certainly not deliverable using only greedy routing. This is important in order to obtain meaningful measurements of the AMRA protocol if it is really meant to be able to improve the underlying algorithm.

Four towns with the center of the towns at the coordinates $(1000, 2000)$, $(500, 500)$, $(2500, 500)$, $(2500, 2000)$ are positioned in an area of $3000m * 2000m$. Three highways connect the four towns so that they form a horseshoe. The 500 nodes follow the rules of the Restricted-Random-Waypoint Mobility Model, 300 thereof are *commuters* with a town-change probability of 80% and 200 are normal nodes with a probability of changing town of 10%. The commuters are needed to guarantee a minimum network connectivity among the towns, therefore they only select a random position in the same town every fifth time and they have a pause time of only 1 second.

All the data traffic is sent out from the 200 normal nodes, the commuters are only used to redirect the packets.



(a) Shortest Path - hops



(b) Shortest Path - Euclidean

Figure 7.4: Comparison of Shortest Paths

7.2.1 Number of Ants Sent

Unidirectional Traffic

These simulations test the influence of ants on the routing of data packets in the AMRA protocol. Figure 7.5(a) shows the efficiency of AMRA and GFG/GPSR with regard to hop counts compared to the path with the fewest possible hops, if only unidirectional traffic is generated.

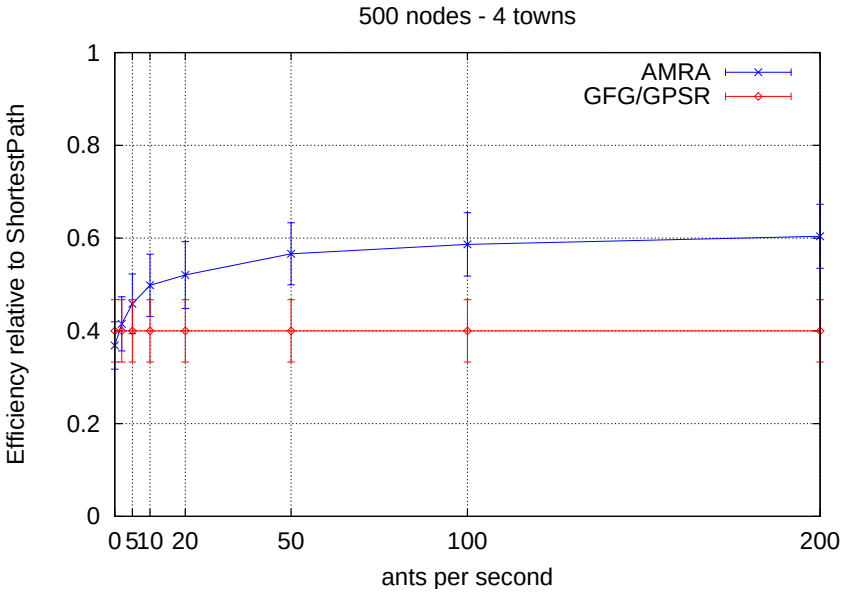
An efficiency of 0.5 means that it took the packets on average double the amount of hops compared to the shortest possible path in that particular network situation. If it takes a packet 10 hops with Shortest Path algorithm, 17 hops with AMRA and 25 with GFG/GPSR, the efficiency of AMRA relative to Shortest Path would be 0.59 and GFG/GPSR's would be 0.4. The values of AMRA and GFG/GPSR can be compared directly: If the efficiency of AMRA is 50% better than the GFG/GPSR's, it means that it took GFG/GPSR on average 50% more hops than AMRA to send the packets from source to destination.

For every total amount of ants sent per second, 10 independent simulations are carried out (10 different random seeds) with different node movements. The error bars in the diagrams show the double-sided 90% confidence interval of the 10 different simulations with the same amount of ants.

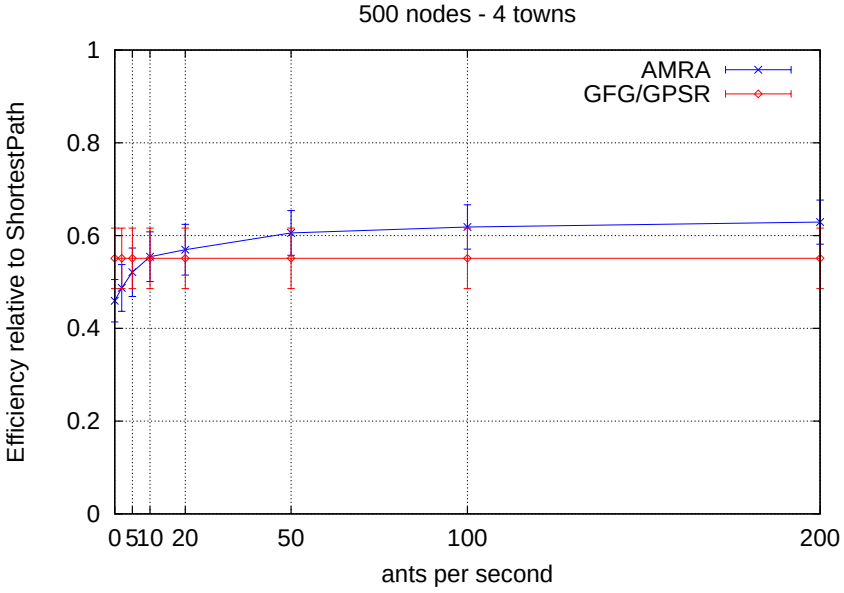
As a general trend, one can recognize that the more ants are sent through the network, the more the AMRA-protocol hop-count performance in the situation of unidirectional traffic proves to be better. The effect flattens out above 50 ants per second. In this simulation, 50 ants per second means that on average every node sends an ant to a random Logical Router in the network every 10 seconds. For the GFG/GPSR protocol, the ants have consequently no influence at all. Because the same 10 different node movement schemes are used for the different amount of ants tested, the results of the GFG/GPSR always stay exactly the same.

Compared to GFG/GPSR, AMRA performs better if a minimum amount of ants are sent; if only little network traffic is generated (10 data packets per second from the 10 sources). With no or only a few ants, the two protocols perform approximately the same as the confidence intervals overlap.

Figure 7.5(b) shows the efficiency with regard to the Euclidean distance the packets traveled on average. If only the performance of AMRA is analyzed, the tendency is the same as in the hop-count performance. What is noticeable is that the GFG/GPSR performs better with regard to the Euclidean distance than with regard to hop counts. This is easily explained with the high amount of hops the Perimeter Mode of the GFG/GPSR produces compared with the Greedy Mode to cover the same Euclidean distance.



(a) Hop efficiency



(b) Distance efficiency

Figure 7.5: 10 sources with unidirectional traffic and ants support

Bidirectional Traffic

With bidirectional traffic, the simulations are run the same as with the unidirectional traffic, the only difference being the 10 sources do not send the traffic to any other node. They build 5 pairs of communication partners that send the traffic among themselves.

As the results show in figure 7.6(a) the nodes cannot benefit from the sent ants. The performance of AMRA is approximately the same over all the simulations. This behavior is not astonishing because the packets of the two communicating nodes put a pheromone trail on the path they take, which helps route the packets sent in the opposite direction. The communicating nodes help route back other data packets with their own data traffic.

The effect of the ants on the Euclidean-distance performance is similar to the one in the unidirectional traffic. The difference between AMRA and GFG/GPSR is much smaller than it is with regard to hop counts, as shown in figure 7.6(b).

As a general conclusion, ants do not help to achieve a better routing performance in bidirectional traffic, but on the other hand they are not harmful to the overall performance and needed for unidirectional traffic.

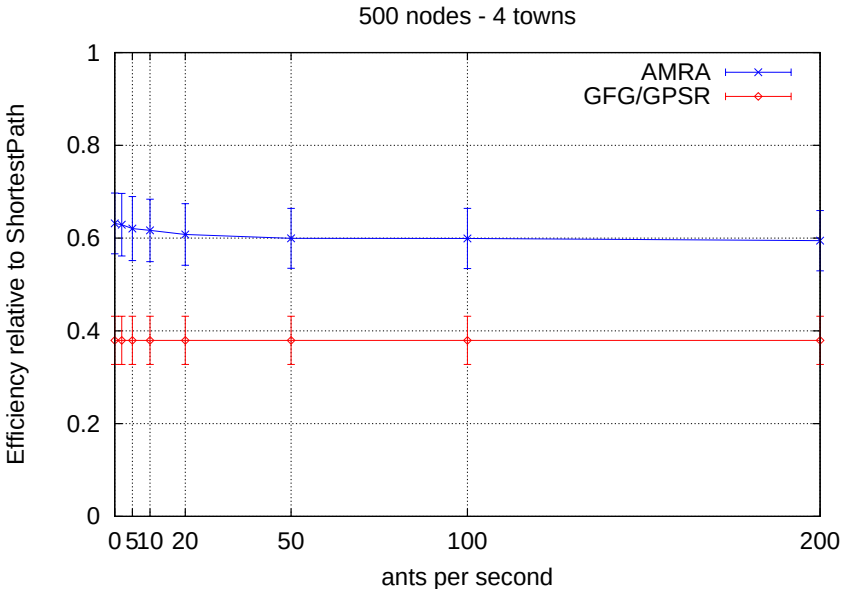
No Ants at all

To gain an advantage from the pheromone trail that data packet deposit for each other in bidirectional traffic, it is not necessary that two specific nodes communicate directly with each other. The nodes also update its pheromone tables with network traffic where they are not directly involved but have passed their footprint. If node a communicates unidirectionally with another node b positioned in a different Logical Router, and at the same time node c in the same Logical Router as b communicates unidirectionally with a node d close to a, a bidirectional-like situation originates (figure 7.7(a)). One another, the sending nodes optimize the path of the packets as shown in figure 7.7(b).

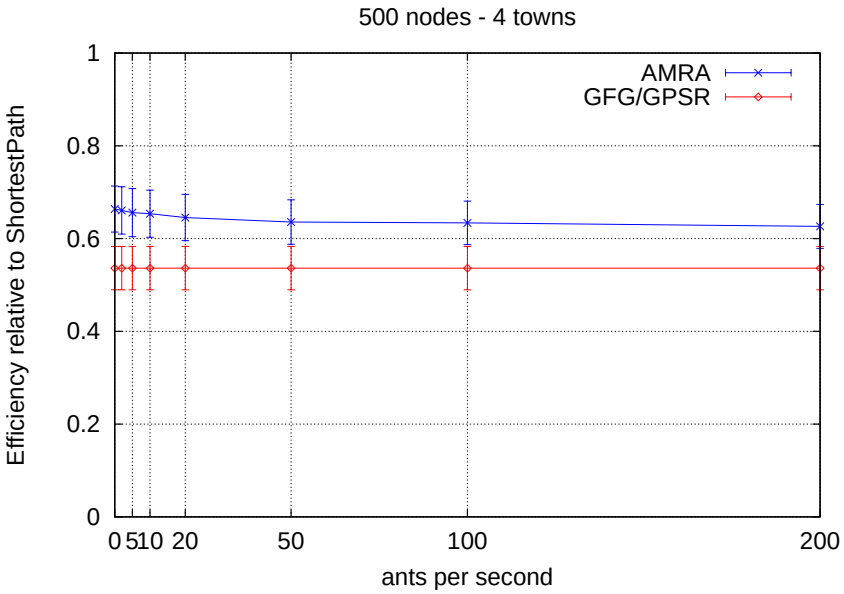
The more traffic in the network, the more bidirectional-like situations are provoked. In networks with a lot of traffic, even if only unidirectional traffic between nodes is generated, the network should almost perform as it does real bidirectional traffic.

Figures 7.8 show the results of simulations where no ants at all are sent and the number of sources sending unidirectional traffic is raised from 1 to 200. As expected the AMRA protocol performs better with regard to hop counts (figure 7.8(a)) the more traffic there is in the network. The values below 20 are not very trustworthy because there are big differences between the individual simulations and therefore the confidence intervals are large. With a higher number of sources this problem disappears and the results are trustworthy. In the Euclidean measurements shown in figure 7.8(b), once more the characteristics of the graph for AMRA is the same as in the hop-count measurements, whereas the GFG/GPSR performs better in the Euclidean sense than with regard to the hop counts.

As a conclusion to these simulations, in a network with a lot of traffic or in networks where most of the traffic is bidirectional, ants are not needed to obtain a performance gain through the use of



(a) Hop efficiency



(b) Distance efficiency

Figure 7.6: 10 sources with bidirectional traffic and ants support

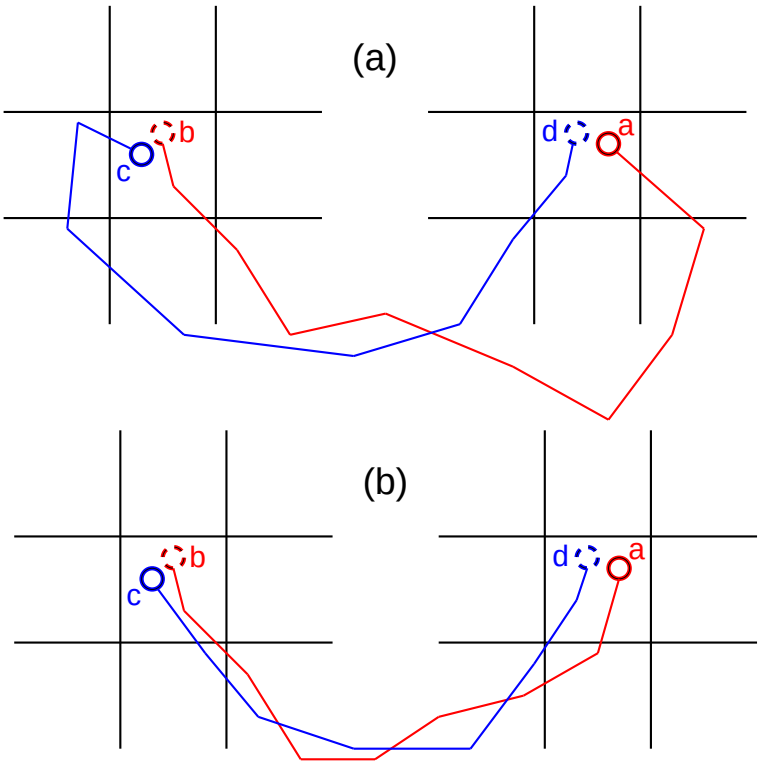
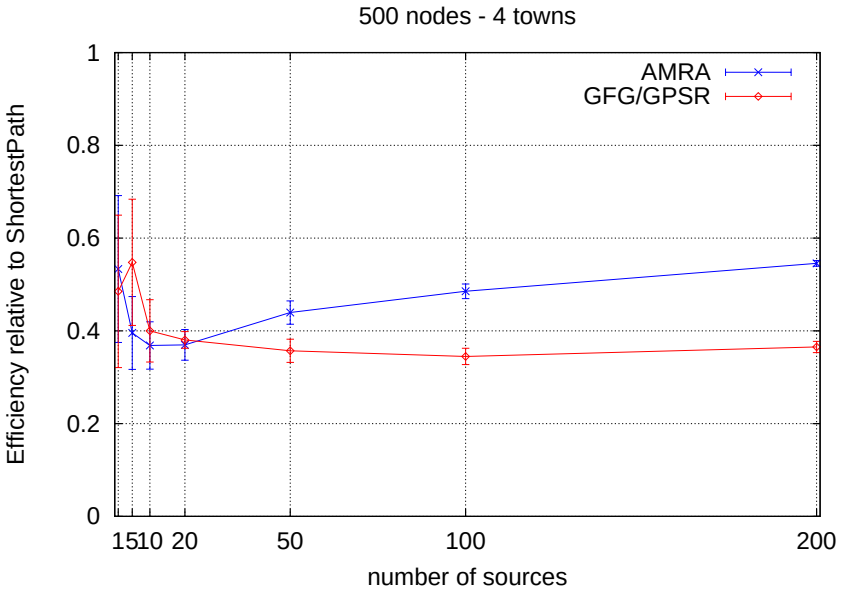
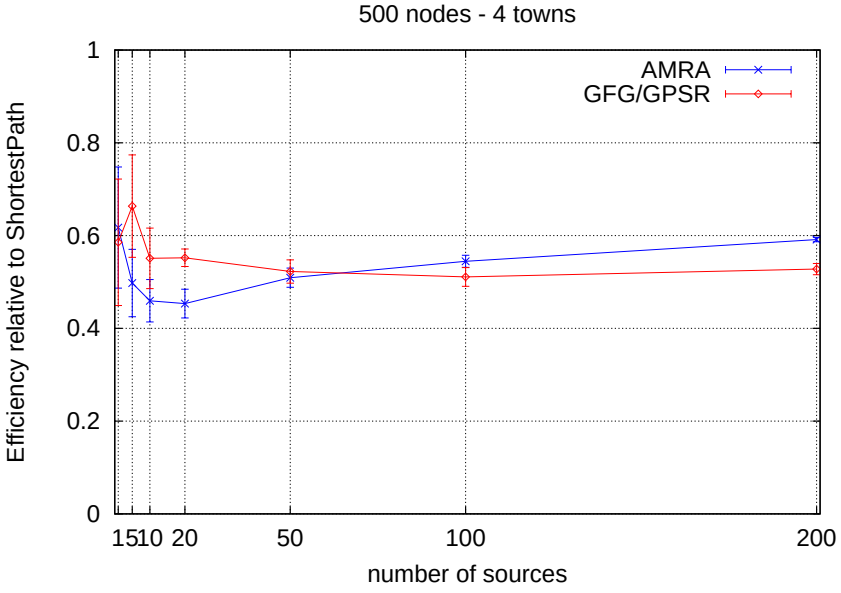


Figure 7.7: Bidirectional-like data traffic

the AMRA protocol instead of simple GFG/GPSR. This is very interesting because the overhead produced by the AMRA protocol is then reduced to the additional bytes needed for the AMRA header in the data packets. No additional ant traffic is required any more.



(a) Hop efficiency



(b) Distance efficiency

Figure 7.8: No ants sent in unidirectional traffic

7.2.2 Size of Logical Routers

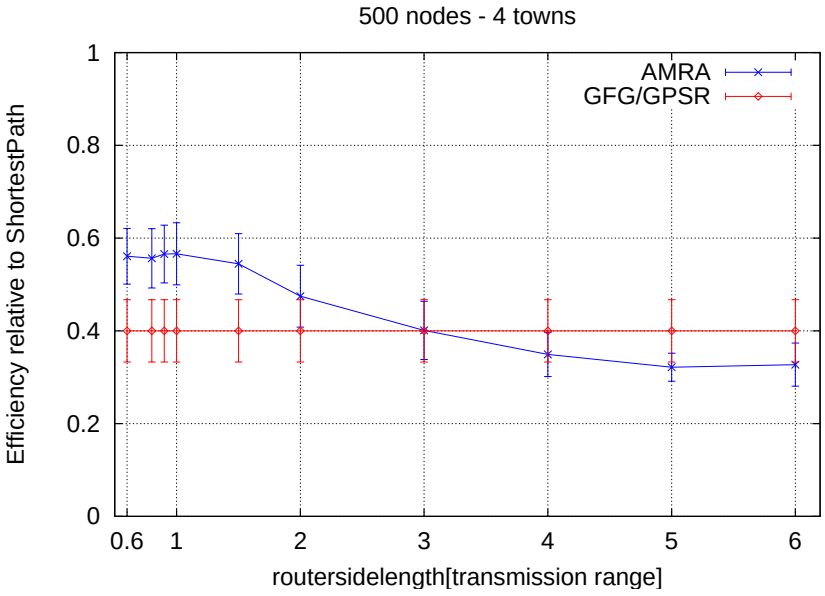
In this section the question of how the size of the Logical Routers influences the simulation results shall be answered. Within Logical Routers, the data packets are usually routed greedy towards the next Anchor Point. It is very unlikely that the underlying algorithm has to switch into backup mode if an Anchor Point for the package was set. Traffic out of the direction of the Anchor Point must be received that a packet can be routed to that direction, therefore a connection should be available. With this background it seems possible that AMRA performs better if the size of the Logical Routers gets bigger than just a square with the side length of the transmission range.

As results in figure 7.9 show, the possible benefit of a bigger router cannot compensate the loss of precision when the network is divided into bigger parts. Looking at the hop-count performance in figure 7.9(a), the efficiency is highest if the side length of the Logical Router is equal to the transmission range of the nodes. If using bigger Logical Routers, performance breaks down. If using a Logical Router size more than 3 times the side length of the transmission range, the efficiency of AMRA falls even below the efficiency of GFG/GPSR.

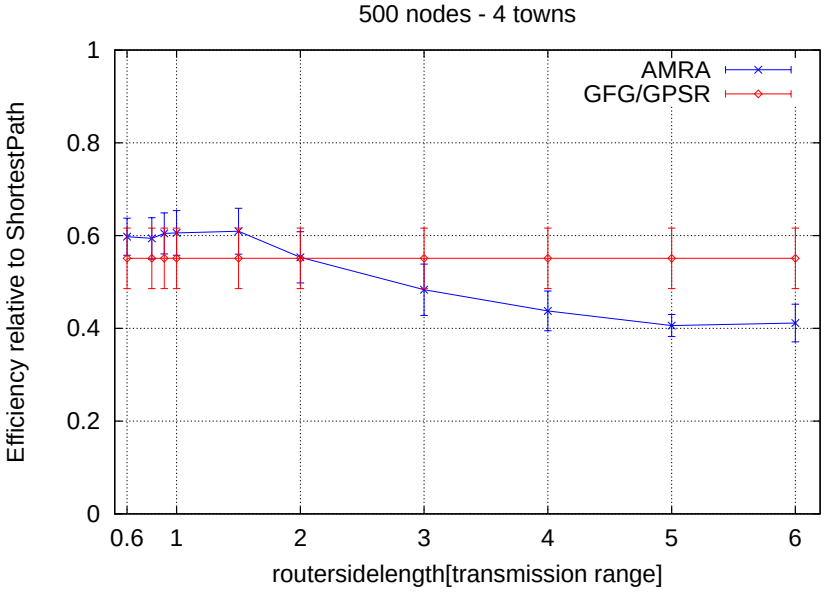
If using a smaller Logical Router side length than the transmission range, the efficiency also becomes worse, because the greedy mode cannot use the full transmission range to forward to the next node any more. The Anchor Points to which the packet should be routed are potentially within the transmission radius of the sending node and therefore the hops taken by greedy forwarding are on average shorter.

The same attitude is shown by the graph of the performance measurements in the Euclidean sense (figure 7.9(b)), only that the performance of AMRA already falls at a router side length of 2 below the performance of GFG/GPSR. This difference is caused by the fact that GFG/GPSR performs better in the Euclidean sense than it does with regard to hop counts as mentioned above.

Differences in the results between the unidirectional traffic shown in figure 7.9 and bidirectional traffic shown in figure 7.10) are within the confidence intervals and therefore the results are not discussed separately. The only mentionable singularity in the bidirectional results is the performance maximum of the AMRA protocol in the Euclidean sense at 1.5 for the Logical Router side length. Because the performance in the Euclidean sense is not as important as the performance with regard to hop counts, this little irregularity needs no further attention.

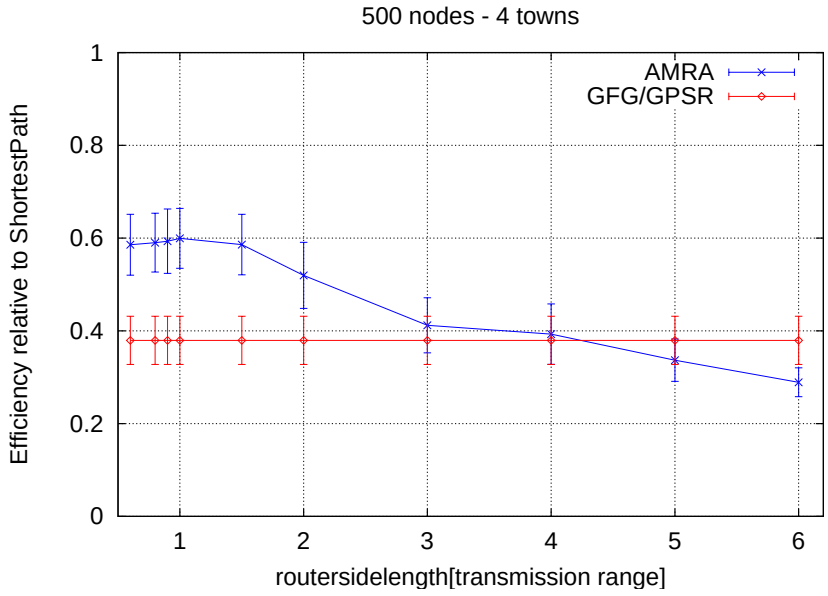


(a) Hop efficiency

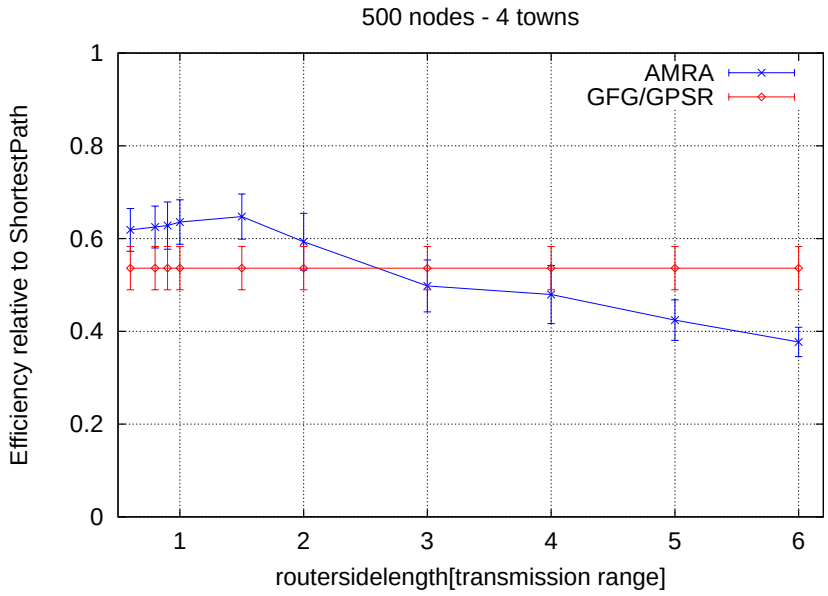


(b) Distance efficiency

Figure 7.9: 10 sources with unidirectional traffic - 50 ants per second



(a) Hop efficiency



(b) Distance efficiency

Figure 7.10: 10 sources with bidirectional traffic - 50 ants per second

7.2.3 Constant C in Pheromone Calculations

With formula 4.3 the link quality of a just received packet can be calculated. A higher value r' indicates a higher positive rating for the Logical Link the packet came from. By changing the value of the constant C, the influence a single packet has on the pheromone values is modified. A bigger value for C leads to bigger changes of the pheromone values in the routing tables, while a smaller value C reduces the influence of a single packet.

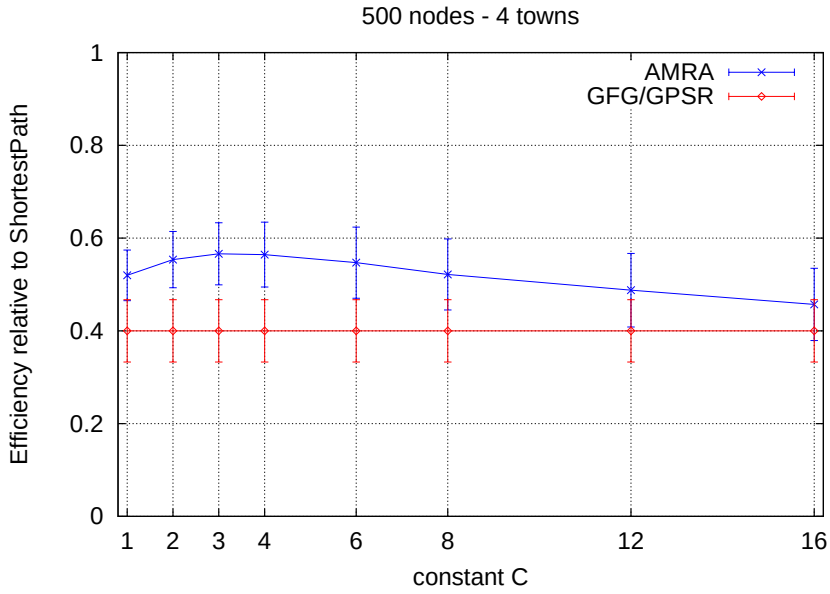
The results in figures 7.11 and 7.12 show that the best performance of AMRA is achieved if the value for the constant C is chosen out of the interval $[2, 4]$. Whether the traffic was sent unidirectionally or bidirectionally has no influence on the characteristics of the resulting graphs. The differences according to hop-count performance and performance in the Euclidean sense are also insignificant, therefore they are not discussed separately.

7.2.4 Amount of Network Traffic

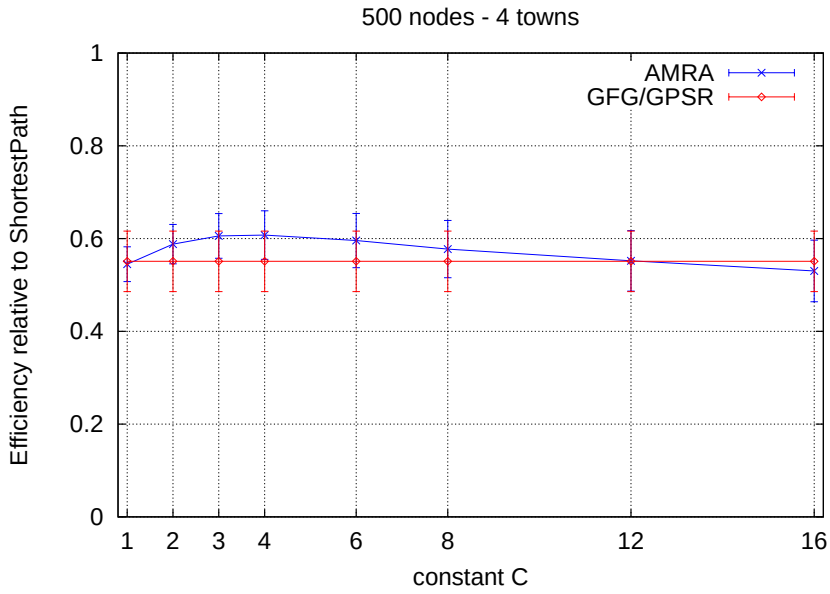
In section 7.2.1 simulations with changing amounts of sending sources without sending any ants are discussed. In this section the effect of different numbers of sending sources with additional ant traffic is described.

The drawback if no ants at all are sent through the network can be seen in figure 7.8(a). If only a few sources send data traffic, in unidirectional manner, the performance of the AMRA protocol is not improved compared to the performance of GFG/GPSR. Figure 7.13 shows the test results if additional 50 ants per second are sent randomly through the network. The effect is a rather stable hop-count performance (figure 7.13(a)) for AMRA over the whole simulation achieves a significantly higher efficiency than GFG/GPSR.

The same effect can be observed comparing the figure 7.8(b) with no ants and figure 7.13(b) with 50 ants sent per second where the covered Euclidean distance is evaluated. The ants help route the packets if the network traffic is low and do not harm the performance if more normal network traffic is sent.

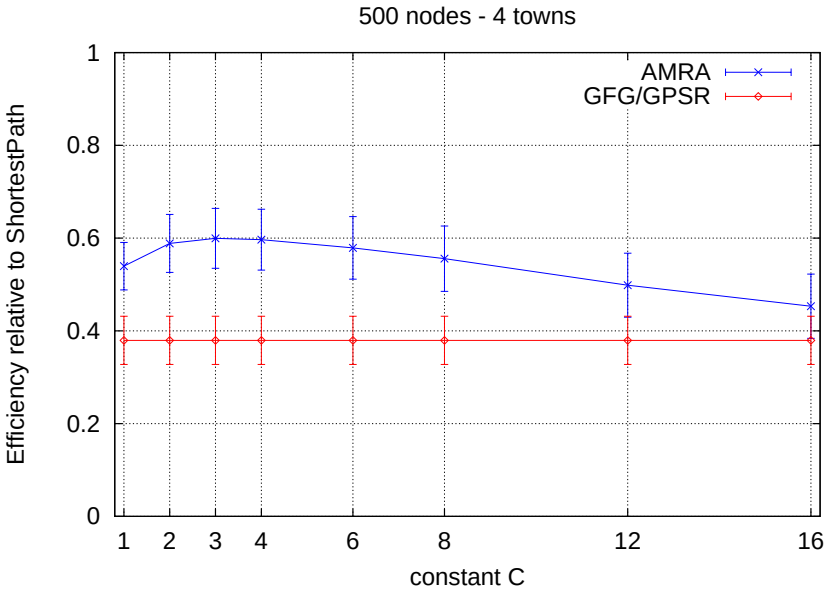


(a) Hop efficiency

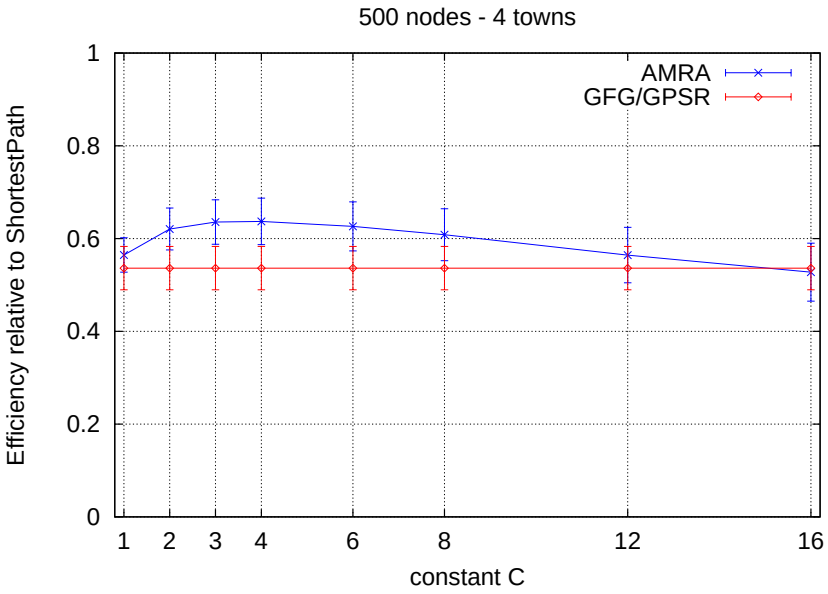


(b) Distance efficiency

Figure 7.11: 10 sources with unidirectional traffic - 50 ants per second

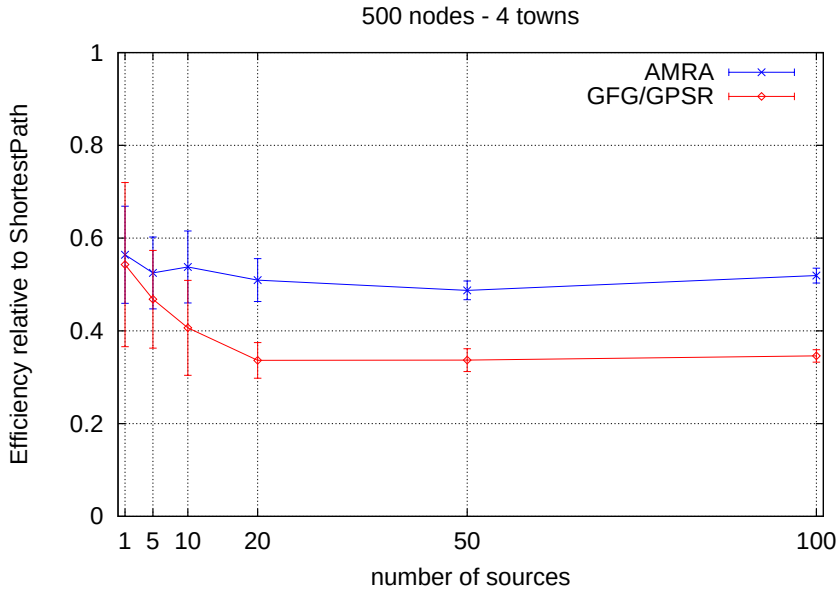


(a) Hop efficiency

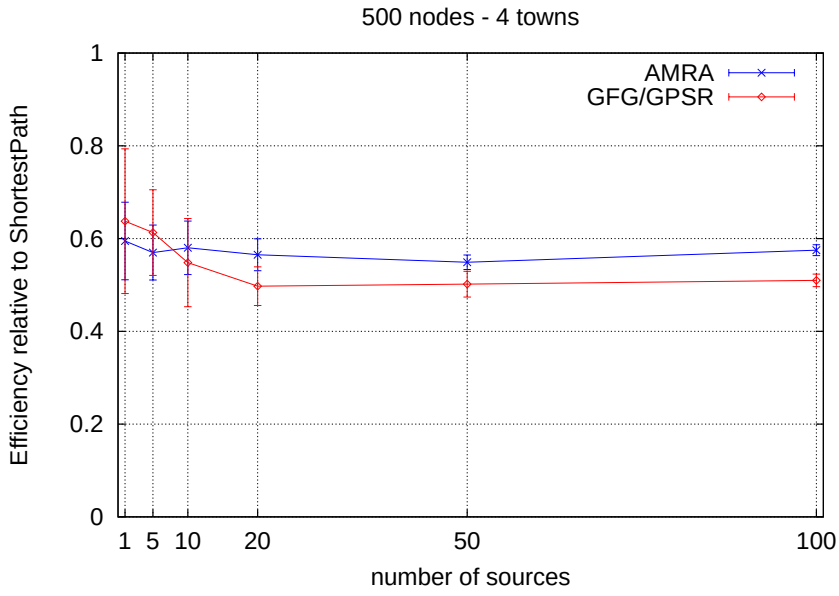


(b) Distance efficiency

Figure 7.12: 10 sources with bidirectional traffic - 50 ants per second

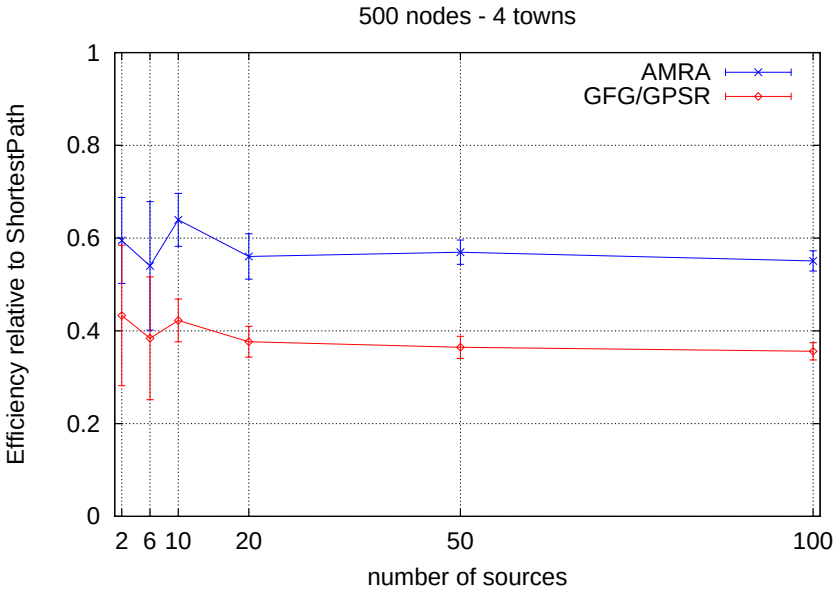


(a) Hop efficiency

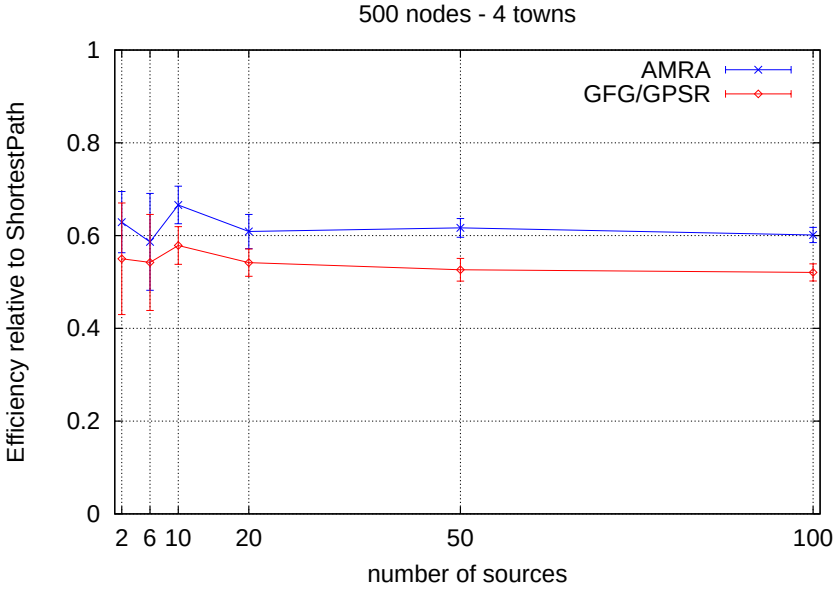


(b) Distance efficiency

Figure 7.13: unidirectional traffic - 50 ants per second



(a) Hop efficiency



(b) Distance efficiency

Figure 7.14: bidirectional traffic - 50 ants per second

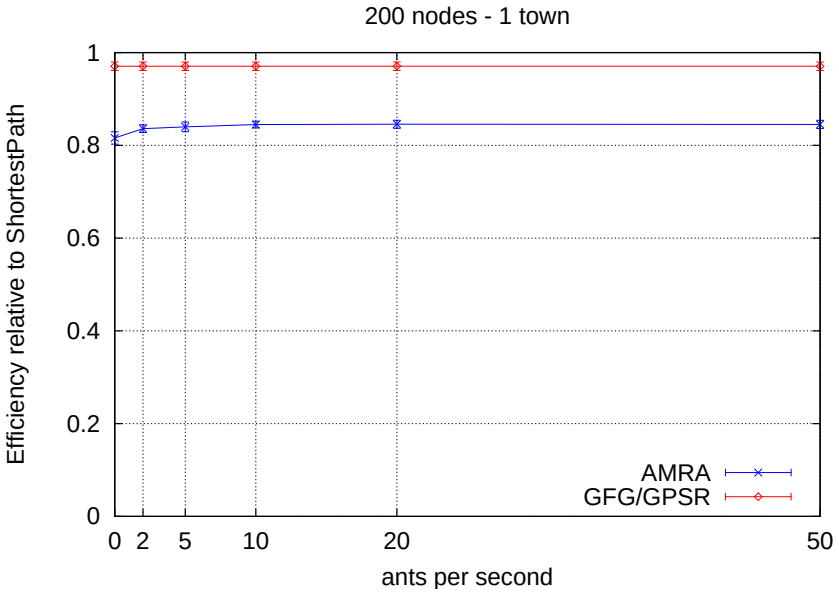
7.3 A Simple Network Scenario

The network scenario of figure 7.2 was selected to evaluate the behavior of the AMRA protocol if the routing of the data packets is easy. The GFG/GPSR protocol can route all the packets by using only the greedy mode. With these basic conditions AMRA should yield worse performance measurements. Mainly due to the Anchor Points that are not set in the straight direction towards the destination node, the paths of packets routed with AMRA have a kind of zigzag form. This detours should lower the performance in hop counts and in the Euclidean sense. Additionally, because of the movements of nodes, the routing tables might not be absolutely perfect at any time of the simulation, even though the pheromone is partly balanced out (section 4.4.4). To test if ants might have an influence on the routing in this scenario, the amount of ants sent is raised from 0 to 50. With 50 ants per second, a node would send an ant every 4 seconds.

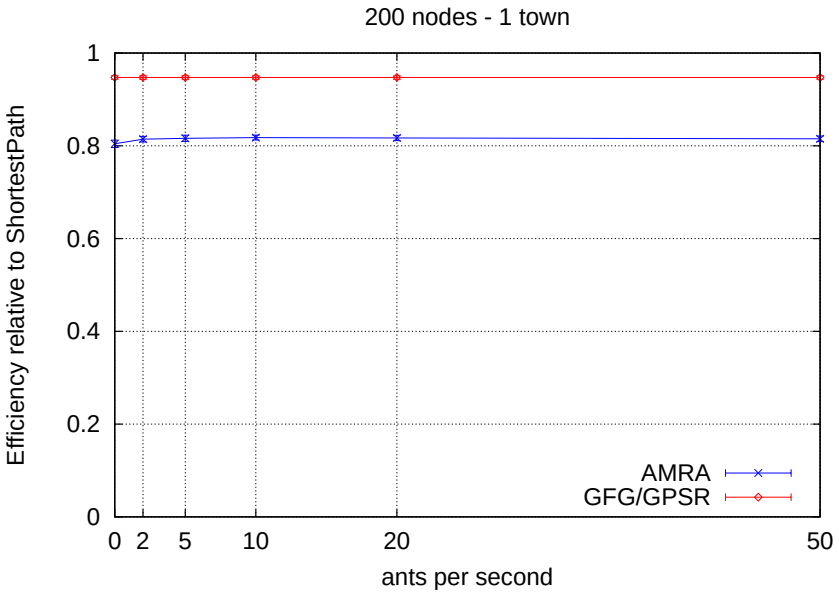
The simulations are done with 200 equal nodes moving with Random-Waypoint Mobility Model with a pause time of 120 seconds. The performance of AMRA in this simulations is not much worse than the performance of GFG/GPSR. Figures 7.15 and 7.16 show the graphs of the different simulations. GFG/GPSR has a performance efficiency of about 0.95 with regard to the hop counts, either in unidirectional (7.15(a) or in bidirectional 7.16(a) traffic. The paths taken by the Greedy Routing are therefore almost ideal. The values of the AMRA protocol are around 0.85 for bidirectional and unidirectional traffic, though the average number of hops used by AMRA compared to GFG/GPSR is around 10% higher.

In the Euclidean sense (figure 7.15(b) and figure 7.16(b)), the performance difference between AMRA and GFG/GPSR is the same as with regard to the hop counts, but differently from the simulations in the main scenario of section 7.2, the Euclidean performance is slightly lower for both protocols.

Ants do not have a major influence on the performance of the AMRA protocol in this scenario, thus the graphs do not vary a lot if more ants have been sent.

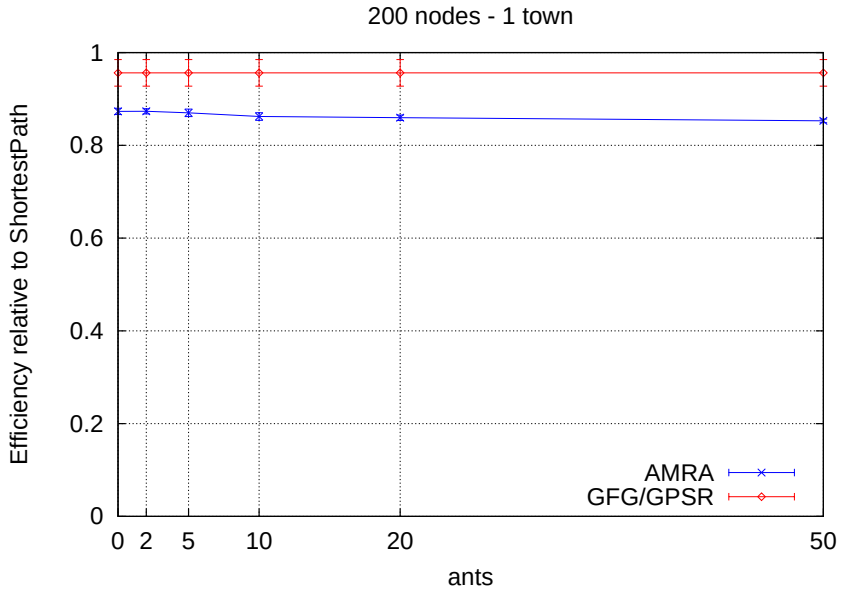


(a) Hop efficiency

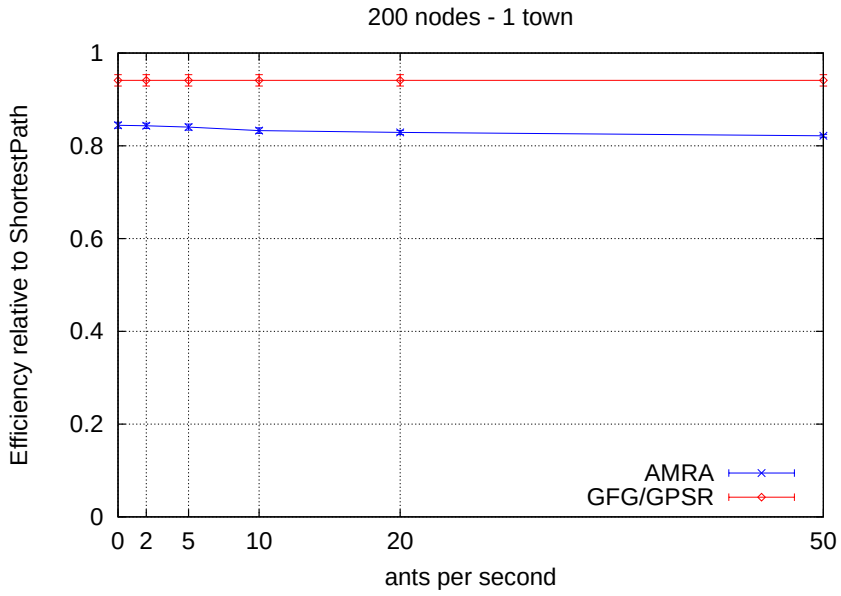


(b) Distance efficiency

Figure 7.15: 10 sources with unidirectional traffic



(a) Hop efficiency



(b) Distance efficiency

Figure 7.16: 10 sources with bidirectional traffic

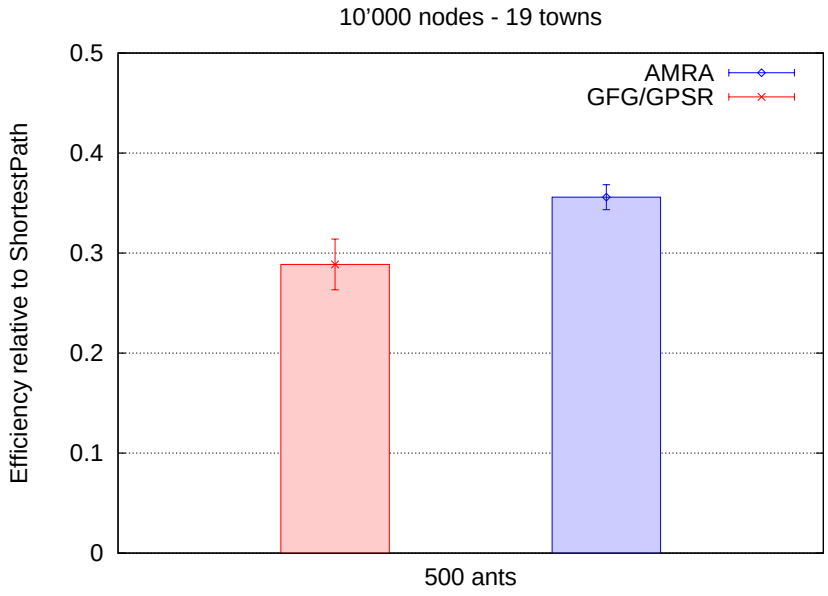
7.4 A Complex Network Scenario

To get an idea about how AMRA performs in a complex network topology with lots of towns and highways and many node free zones between them, the protocols are compared in the scenario of figure 7.3. All simulations are done with 500 ants sent per second, which means that a node sends an ant to a randomly choosen Logical Router every 20 seconds. 50 sources send unidirectional (figures 7.17) and bidirectional (figures 7.18) data traffic.

The performance of AMRA and GFG/GPSR is much worse than in the simpler scenarios compared to the best possible paths, a behavior that was expected because routing in such a complex scenario is much more difficult. The chance that the sender and receiver of a data packet are accidentally in the same town is much smaller if the nodes are spread over 19 instead of only 4 towns.

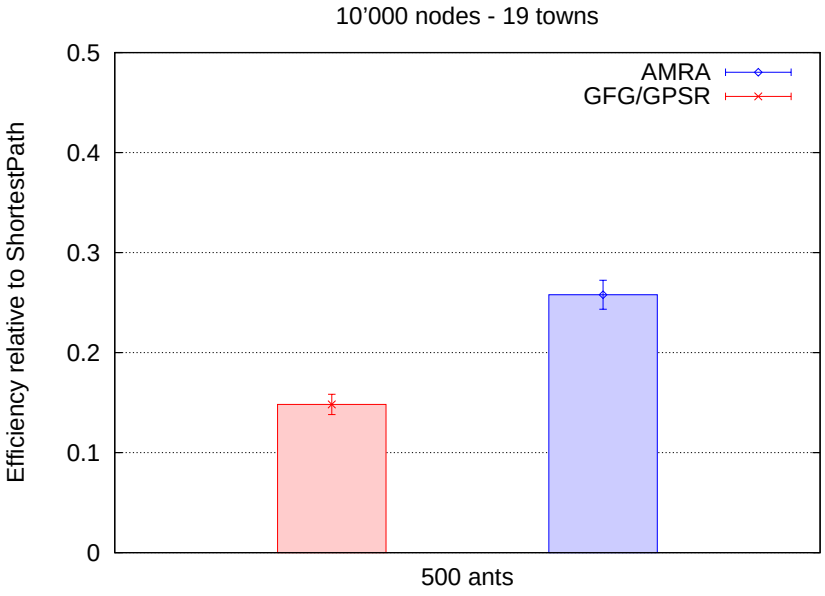
What is interesting is the fact that the difference in performance between AMRA and GFG/GPSR with regard to hop counts and in the Euclidean sense is approximately the same ratio as in the main simulation scenario of section 7.2. The AMRA protocol thus performs about 50% better than the GFG/GPSR protocol with regard to hop counts and about 30% better in the Euclidean sense if unidirectional traffic is generated. Using bidirectional traffic the advantage of AMRA rises to a performance superior by 70% with regard to hop counts and by 40% in the Euclidean sense.

(a) Hop efficiency

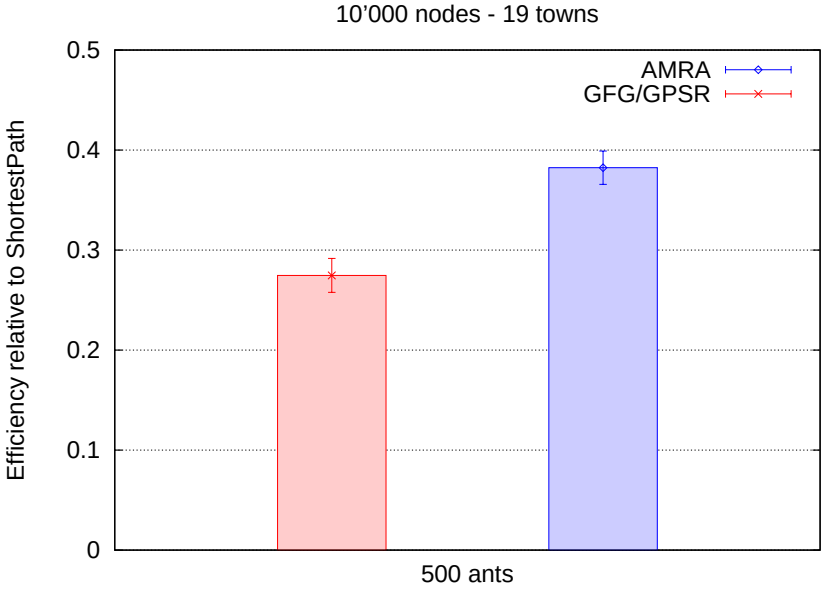


(b) Distance efficiency

Figure 7.17: 200 sources with unidirectional traffic



(a) Hop efficiency

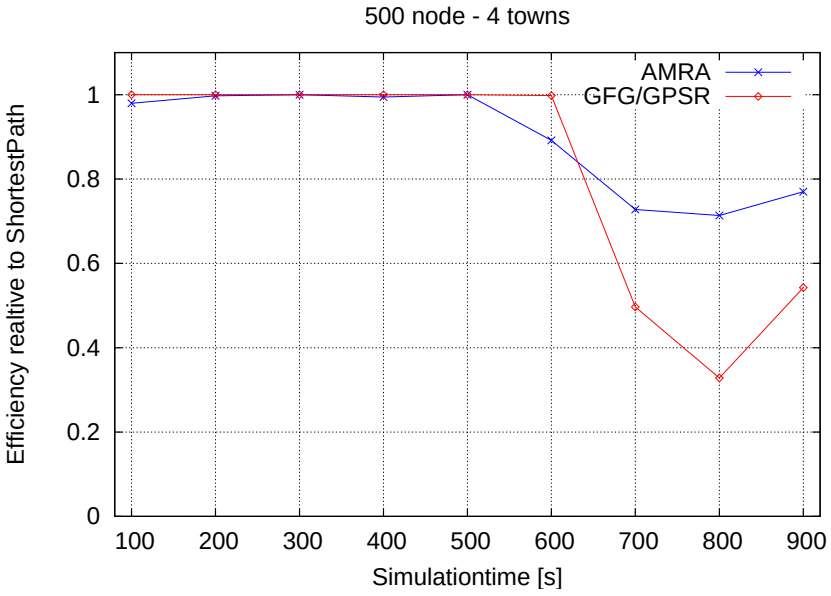


(b) Distance efficiency

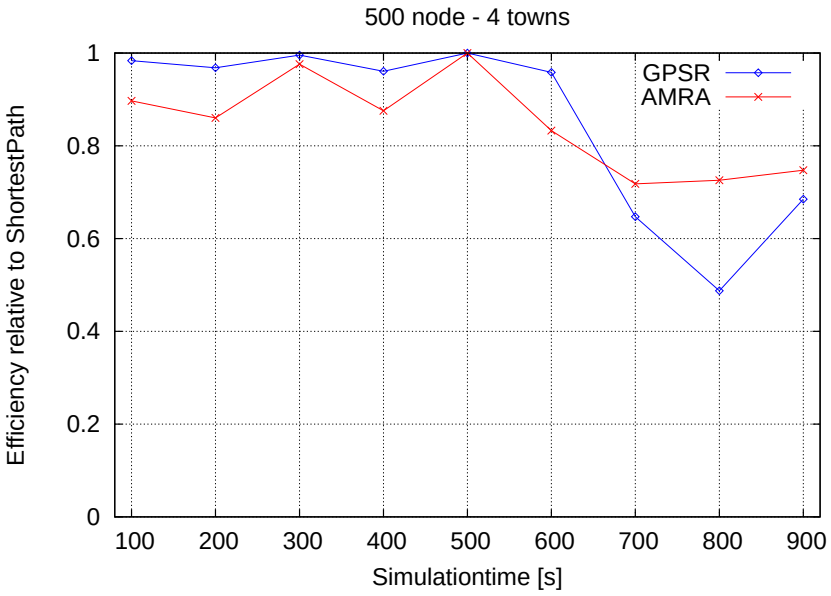
Figure 7.18: 200 sources with bidirectional traffic

7.5 Typical Example

The advantages of the AMRA protocol only appears if the data packets cannot be sent directly with greedy mode from source to destination node. Figures 7.19(a) and 7.19(b) show the development of the performance values of the routing between two nodes that move away from each other. This simulation is done in the main scenario of section 7.2 with four towns and 3 highways. During the first 500 seconds of the simulation the source and the destination node are within the same town and therefore all the traffic sent between them can be routed in greedy mode at a performance value of almost 1 both for, the AMRA and the GFG/GPSR protocol. After 500 seconds one node starts to move into another town and after 600 seconds the same node moves further into a third town, meanwhile the other node is still positioned in the origin town. This movements can be read from the graphs in figure 7.19(a). As long as the performance of AMRA (the first 500 seconds of the simulation) is almost 1, the nodes are very close to each other. When the performance becomes worse for AMRA but is still close to 1 for GFG/GPSR, one node moves away from the other one, but packets can still be routed greedy among them. As soon as the performance of GFG/GPSR breaks down at 600 seconds of the simulation, the routing between the two nodes cannot be done any more with greedy routing. From that moment on, the AMRA protocol performs better for traffic between the two nodes.



(a) Hop efficiency



(b) Distance efficiency

Figure 7.19: Two nodes moving away from each other

Chapter 8

Conclusion and Future Work

8.1 Conclusion

AMRA is able to improve the routing performance of GFG/GPSR in large-scale mobile ad-hoc networks by 50% on average. In simple topologies, where pure Greedy routing succeeds, AMRA declines in performance by 10%. These measurements are based on hop counts. Fewer hop counts usually lead to shorter end-to-end delays and produce less traffic load. In the companion thesis [31], the main simulation focus is laid on the end-to-end delays. The simulations are done in networks with at most 500 nodes, using Qualnet as simulation environment. Results of test simulations according to hop counts correspond to the results gained with the simulator of this thesis. Because in this thesis the higher performance of AMRA according to hop counts is also proved for large-scale mobile ad-hoc networks, the results of the end-to-end delays of the companion thesis can also be looked as significant for networks with $10'000$ nodes.

The additional packet-header fields needed by AMRA causes more traffic overhead. Effects of protocol overhead are unaccounted for this thesis. Additional traffic overhead is caused by ants that are sent through the network. The total overhead of the protocol depends on the amount of ants sent. As shown in the results, in networks with much traffic or in networks where only bidirecitonal traffic occurs, ants are a waste of bandwith and could be left out.

Even if no special ants are sent, AMRA is still ant-based, because data packets also lay pheromone and behave ant-like.

8.2 Future Work

- Routing information is collected by every node individually. No exchange of potentially helpful information is provided between nodes. Routing-table entries a node has might also be interesting to other nodes in the same Logical Router. Ideally, all the nodes that are in the same Logical Router use and maintain the same routing table. A drawback of to the table exchange would be the additional traffic overhead and the knowledge needed about the direct neighbors, which would cannibalize the idea of the next item. This idea of exchanging the routing tables was originally proposed in [1].

- In this work only GFG/GPSR is tested as an underlying algorithm. The principle of AMRA should work with any position-based routing protocol that uses a greedy algorithm as the main routing strategy. A further condition is that the used algorithm does not maintain its own routing table, because this could mislead the routing of AMRA.

  An interesting combination would be AMRA over BLR [32][33]. BLR is a beacon-less routing algorithm that does not need any information about neighboring nodes. The AMRA protocol is not dependent on knowledge of neighboring nodes either. Thus, a combination of the two algorithms would lend itself well.

- In this thesis, the same routing algorithm is used to forward data packets and ants. A different algorithm for ants could improve the performance. For ants the number of hops they need is not critical, because the routing tables are adapted according to the covered Euclidean distance of the ants. Therefore an algorithm like face routing [9] could be of use to find good paths with ants even though it does not provide a greedy mode.

- Special ants as the FANTs and BANTs in the ARA protocol (section 3.2) are not available in AMRA. Such a mechanism could possibly improve the overall routing efficiency of AMRA. In this thesis a node has no possibility of actively enforcing the obtainment of information about a certain zone in the network. With forward and backward ants or similar approaches, a node can selectively fix its routing table.

Bibliography

[1] M. Heissenbüttel and T. Braun, "Ants-based routing in large-scale mobile ad-hoc networks," in *Proceedings of Kommunikation in verteilten Systemen (KiVS '03)*, Leipzig, Germany, Feb. 2003, pp. 91–99.

[2] S. Giordano and M. Hamdi, "Mobility management: The virtual home region," EPFL, Lausanne, Switzerland, Tech. Rep. SSC/1999/037, Oct. 1999.

[3] J. Li, J. Jannotti, D. S. J. De Couto, D. R. Karger, and R. Morris, "A scalable location service for geographic ad-hoc routing," in *Proceedings of the 6th Annual ACM/IEEE International Conference on Mobile Computing and Networking (MOBICOM '00)*, Boston, USA, Aug. 2000, pp. 120–130.

[4] Z. J. Haas and B. Liang, "Ad hoc mobility management with uniform quorum systems," *IEEE/ACM Transactions on Networking*, vol. 7, no. 2, pp. 228–240, Apr. 1999.

[5] S. Basagni, I. Chlamtac, V. R. Syrotiuk, and B. A. Woodward, "A distance routing effect algorithm for mobility (DREAM)," in *Proceedings of the 4th annual ACM/IEEE International Conference on Mobile Computing and Networking (MOBICOM '98)*, Dallas, Texas, USA, Oct. 1998, pp. 76–84.

[6] G. Finn, "Routing and addressing problems in large metropolitan-scale internetworks," Information Sciences Institute, University of Southern California, USA, Tech. Rep. ISI/RR-87-180, Mar. 1987.

[7] H. Takagi and L. Kleinrock, "Optimal transmission ranges for randomly distributed packet radio terminals," *IEEE Transactions on Communications*, vol. 32, no. 3, pp. 246–257, Mar. 1984.

[8] T.-C. Hou and V. Li, "Transmission range control in multihop packet radio networks," *IEEE Transactions on Communications*, vol. 34, no. 1, pp. 38–44, Jan. 1986.

[9] E. Kranakis, H. Singh, and J. Urrutia, "Compass routing on geometric networks," in *Proceedings of the 11th Canadian Conference on Computational Geometry (CCCG '99)*, Vancouver, Canada, Aug. 1999, pp. 51–54.

[10] I. Stojmenovic and X. Lin, "Power-aware localized routing in wireless networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 12, no. 11, pp. 1122–1133, Nov. 2001.

[11] K. Gabriel and R. Sokal, “A new statistical appreach to geographic variation analysis,” in *Systematic Zoology 18*, 1969, pp. 259–278.

[12] G. Toussaint, “The relative neighborhood graph of a finite planar set,” in *Pattern Recognition 12*, April 1980, pp. 261–268.

[13] P. Bose, P. Morin, I. Stojmenovic, and J. Urrutia, “Routing with guaranteed delivery in ad hoc wireless networks,” in *Proceedings of the 3th International ACM Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications (DIALM ’99)*, Seattle, USA, Aug. 1999, pp. 48 – 55.

[14] B. Karp and H. T. Kung, “GPSR: Greedy perimeter stateless routing for wireless networks,” in *Proceedings of the 6th Annual ACM/IEEE International Conference on Mobile Computing and Networking (MOBICOM ’00)*, Boston, USA, Aug. 2000, pp. 243–254.

[15] F. Kuhn, R. Wattenhofer, Y. Zhang, and A. Zollinger, “Geometric ad-hoc routing: Of theory and practice,” in *Proceedings of the 22nd ACM Symposium on the Principles of Distributed Computing (PODC ’03)*, Boston, USA, July 2003, pp. 63–72.

[16] M. Dorigo and G. D. Caro, “The ant colony optimization meta-heuristic,” in *New Ideas in Optimization*, London, 1999, pp. 11–32.

[17] E. Bonabeau, M. Dorigo, and G. Theraulaz, “Swarm intelligence: from natural to artificial intelligence,” 1999.

[18] ——, “Inspiration for optimization from social insect behaviour,” *Nature*, vol. 406, no. 6791, pp. 39–42, July 2000.

[19] G. Di Caro and M. Dorigo, “AntNet: Distributed stigmergetic control for communications networks,” *Journal of Artificial Intelligence Research*, vol. 9, pp. 317–365, Dec. 1998.

[20] M. Güneş, U. Sorges, and I. Bouazizi, “ARA - the ant-colony based routing algorithm for MANETs,” in *Proceedings of the 2002 ICPP Workshop on Ad Hoc Networks (IWAHN ’02)*, Vancouver, Canada, Aug. 2002, pp. 79–85.

[21] M. Roth and S. Wicker, “Termite: Emergent ad-hoc networking,” in *Proceedings of the 2nd Mediterranean Workshop on Ad-Hoc Net- works (Med-Hoc-Net’2003)*, Mahdia, Tunesia, June 2003.

[22] ——, “Termite: Ad-hoc networking with stigmergy,” in *Proceedings of IEEE Global Telecommunications Conference (GLOBECOM’03)*, San Francisco, USA, Dec. 2003.

[23] W. Navidi and T. Camp, “Stationary distributions for the random waypoint mobility model,” *IEEE Transactions on Mobile Computing*, vol. 3, no. 1, pp. 99–108, Mar. 2004.

[24] C. Bettstetter and C. Wagner, “The spatial node distribution of the random waypoint mobility model,” in *Proceedings of the First German Workshop on Mobile Ad-Hoc Networks (WMAN)*, 2002, pp. 41–58, gI Lecture Notes in Informatics.

[25] T. Camp, J. Boleng, and V. Davies, “A survey of mobility models for ad hoc network research,” *Wireless Communications and Mobile Computing (WCMC)*, vol. 2, no. 5, pp. 483–502, 2002, special issue on Mobile Ad Hoc Networking.

[26] E. M. Royer, P. M. Melliar-Smith, and L. E. Moser, “An analysis of the optimum node density for ad hoc mobile networks,” in *Proceedings of the IEEE International Conference on Communications (ICC 2001)*, 2001, pp. 857–861.

[27] L. Blazevic, S. Giordano, and J.-Y. Le Boudec, “Self organized terminode routing,” *Cluster Computing Journal*, vol. 5, no. 2, pp. 205–218, Apr. 2002.

[28] X. Hong, M. Gerla, G. Pei, and C. Chiang, “A group mobility model for ad hoc wireless networks,” in *Proceedings of the 2nd ACM International Workshop on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM)*, 1999, pp. 53–60.

[29] F. Kuhn, R. Wattenhofer, and A. Zollinger, “Asymptotically optimal geometric mobile ad-hoc routing,” in *Proceedings of the 6th International ACM Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications (DIALM ’02)*, Atlanta, USA, Sept. 2002, pp. 24–33.

[30] F. K. R. Wattenhofer and A. Zollinger, “Worst-case optimal and average-case efficient geometric ad-hoc routing,” in *Proceedings of the 4th ACM International Symposium on Mobile and Ad Hoc Networking and Computing (MobiHoc ’03)*, Maryland, USA, June 2003, pp. 267 – 278.

[31] D. Joerg, “Ants-based routing in mobile ad-hoc networks,” Master’s thesis, University of Bern, Bern, Switzerland, Dec. 2004.

[32] M. Heissenbüttel, T. Braun, T. Bernoulli, and M. Wälchli, “BLR: Beacon-less routing algorithm for mobile ad-hoc networks,” *Elsevier’s Computer Communications Journal (Special Issue)*, vol. 27, no. 11, pp. 1076–1086, July 2004.

[33] T. Bernoulli, “Beacon-less routing in mobile ad-hoc networks,” Master’s thesis, University of Bern, Bern, Switzerland, Nov. 2004.