Performance Comparison Of MANET Routing Protocols In Different Network Sizes

Computer Science Project

David Oliver Jörg

Institute of Computer Science and Applied Mathematics

Computer Networks and Distributed Systems (RVS)

University of Berne, Switzerland

2003

head:
Prof. Dr. Torsten Braun

assisted by:
Marc Heissenbüttel

Abstract

A Mobile Ad-Hoc Network (MANET) is a collection of wireless mobile nodes forming a temporary network without using any centralized access point, infrastructure, or centralized administration. To establish a data transmission between two nodes, typically multiple hops are required due to the limited transmission range. Mobility of the different nodes makes the situation even more complicated. Multiple routing protocols especially for these conditions have been developed during the last years, to find optimized routes from a source to some destination. The scope of this project was to test routing performance of four different routing protocols (AODV, DSR, LAR 1 and ZRP) in variable network sizes up to thousand nodes. We have used QualNet Simulator from Scalable Networks to perform the simulations. This paper describes the different routing protocols, the simulator and the experiment setup and finally presents and discusses the results.

Contents

1 Introduction **4**
1.1 Mobile Ad-Hoc Networks (MANET) . . . 4

2 Ad Hoc Network Routing Protocols **5**
2.1 Ad Hoc On-Demand Distance Vector Routing (AODV) . . . 5
2.1.1 Path Discovery Process . . . 5
2.1.2 Maintaining Routes . . . 6
2.2 Dynamic Source Routing (DSR) . . . 6
2.2.1 Route Discovery . . . 7
2.2.2 Route Maintenance . . . 8
2.3 Location-Aided Routing (LAR) . . . 9
2.3.1 LAR Request Zone Types . . . 10
2.4 Zone Routing Protocol (ZRP) . . . 12
2.4.1 Intrazone Routing Protocol (IARP) . . . 13
2.4.2 Interzone Routing Protocol (IERP) . . . 13
2.4.3 Bordercast Resolution Protocol (BRP) . . . 13
2.4.4 ZRP Example . . . 14

3 Methodology **15**
3.1 Simulation Environment . . . 16
3.1.1 QualNet . . . 16
3.2 Mobility Model . . . 21
3.2.1 Random Waypoint Mobility Model . . . 22
3.3 Application Model . . . 23
3.4 Metrics . . . 23

4 Simulation Results **23**
4.1 Overview . . . 23
4.2 Problems . . . 24
4.3 Packet Delivery Ratio . . . 26
4.4 Routing Overhead . . . 26
4.5 Average End-To-End Delay . . . 27
4.6 Average Hop Count . . . 27

5 Related work **27**
5.1 Broch, Maltz, Johnson, Hu and Jetcheva . . . 27
5.2 Mingliang, Tay and Long . . . 28
5.3 Das, Perkins, Royer and Marina . . . 28

6 Conclusions **29**

List of Figures

1 Infrastructured and ad-hoc networks. 4
2 AODV Path Discovery Process. 6
3 AODV Route Maintenance by using Link Failure Notification Message 7
4 DSR Route Discovery Process. 8
5 LAR Expected Zone. 10
6 LAR - Different Request Zones. 11
7 LAR Scheme 1 - Request Zone. 11
8 ZRP - Routing Zone of Node A, ρ = 2 12
9 A Sample Network with ρ = 2. 14
10 A ZRP Routing Example. 15
11 Node placement with 1000 nodes, rectangular simulation area and 1 source node 15
12 QualNet Event Handling. 17
13 QualNet Simulator Batch-Mode Execution. 17
14 Traveling pattern of a mobile node using Random Waypoint Mobility [1]. 22
15 Comparison of the four protocols as a function of network size. 24

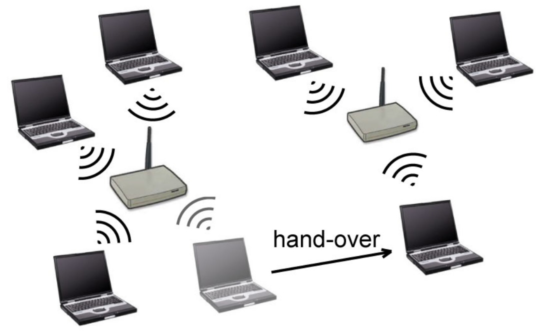
1 Introduction

1.1 Mobile Ad-Hoc Networks (MANET)

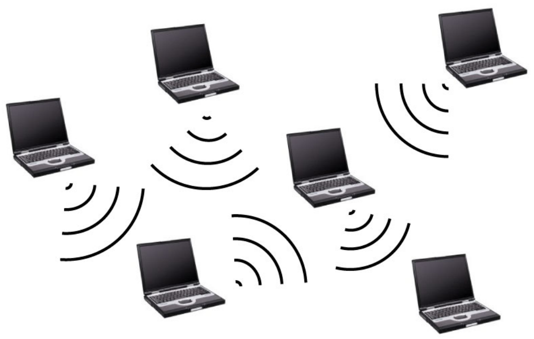
In the last couple of years, the use of wireless networks has become more and more popular. There exist three types of mobile wireless networks: *infrastructured networks*, *ad-hoc networks* and *hybrid networks* which combine infrastructured and ad-hoc aspects.

An infrastructured network (Figure 1(a)) consists of wireless mobile nodes and one or more bridges, which connect the wireless network to the wired network. These bridges are called *base stations*. A mobile node within the network searches for the nearest base station (e.g. the one with the best signal strength), connects to it and communicates with it. The important fact is that all communication is taking place between the wireless node and the base station but not between different wireless nodes. While the mobile node is traveling around and all of a sudden gets out of range of the current base station, a *handover* to a new base station will let the mobile node communicate seamlessly with the new base station.

In contrary to infrastructured networks, an ad-hoc network lacks any infrastructure. There are no base stations, no fixed routers and no centralized administration. All nodes may move randomly and are connecting dynamically to each other. Therefore all nodes are operating as routers and need to be capable to discover and maintain routes to every other node in the network and to propagate packets accordingly. Mobile ad-hoc networks may be used in areas with little or no communication infrastructure: think of emergency searches, rescue operations, or places where people wish to quickly share information, like meetings etc.



(a) An infrastructured network with two base stations.



(b) A mobile ad-hoc network.

Figure 1: Infrastructured and ad-hoc networks.

A hybrid network combines both aspects described before. It makes use of any available base stations while it also supports infrastructureless communication.

2 Ad Hoc Network Routing Protocols

2.1 Ad Hoc On-Demand Distance Vector Routing (AODV)

The Ad Hoc On-Demand Distance Vector routing protocol (AODV) is an improvement of the Destination-Sequenced Distance Vector routing protocol (DSDV)[1]. DSDV has its efficiency in creating smaller ad-hoc networks. Since it requires periodic advertisement and global dissemination of connectivity information for correct operation, it leads to frequent system-wide broadcasts. Therefore the size of DSDV ad-hoc networks is strongly limited. When using DSDV, every mobile node also needs to maintain a complete list of routes for each destination within the mobile network.

The advantage of AODV is that it tries to minimize the number of required broadcasts. It creates the routes on a on-demand basis, as opposed to maintain a complete list of routes for each destination. Therefore, the authors of AODV classify it as a *pure on-demand route acquisition system* [3].

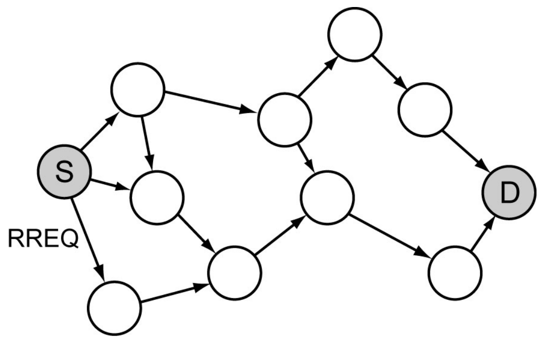
2.1.1 Path Discovery Process

When trying to send a message to a destination node without knowing an active route[2] to it, the sending node will initiate a path discovery process. A route request message (RREQ) is broadcasted to all neighbors, which continue to broadcast the message to their neighbors and so on. The forwarding process is continued until the destination node is reached or until a intermediate node knows a route to the destination that is new enough. To ensure loop-free and most recent route information, every node maintains two counters: *sequence number* and *broadcast_id*. The broadcast_id and the address of the source node uniquely identify a RREQ message. broadcast_id is incremented for every RREQ the source node initiates. An intermediate node can receive multiple copies of the same route request broadcast from various neighbors. In this case – if a node has already received a RREQ with the same source address and broadcast_id – it will discard the packet without broadcasting it furthermore. When an intermediate node forwards the RREQ message, it records the address of the neighbor from which it received the first copy of the broadcast packet. This way, the *reverse path* from all nodes back to the source is being built automatically. The RREQ packet contains two sequence numbers: the source sequence number and the last destination sequence number known to the source. The source sequence number is used to maintain “freshness” information about the reverse route to the source while the destination sequence number specifies what actuality a route to the destination must have before it is accepted by the source. [3]

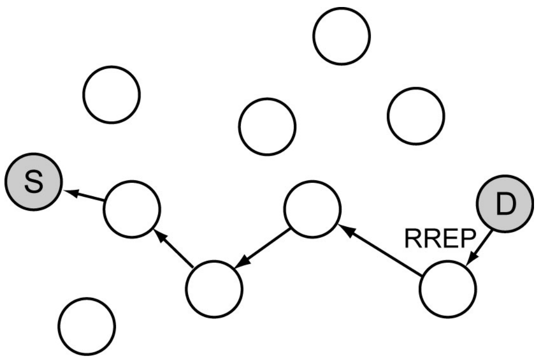
[1]When using Destination-Sequenced Distance Vector routing, every mobile node in the network maintains a routing table where it lists all possible destinations within the network and the corresponding hop counts to them. The protocol requires periodic advertisement of the routing information throughout the entire network by using *full dump* and *incremental* packets. This potentially large amount of network traffic strongly limits the use of this protocol to small ad-hoc networks. A detailed description of DSDV can be found in [2].

[2]A route is considered active if it has an entry in the routing table that is marked as valid. Active routes expire after a certain amount of time or on occurence of a link failure. Only active routes can be used to forward data packets.

When the route request broadcast reaches the destination or an intermediate node with a fresh enough route, the node responds by sending a unicast route reply packet (RREP) back to the node from which it received the RREQ. So actually the packet is sent back reverse the path built during broadcast forwarding. A route is considered fresh enough, if the intermediate node's route to the destination node has a destination sequence number which is equal or greater than the one contained in the RREQ packet. As the RREP is sent back to the source, every intermediate node along this path adds a forward route entry to its routing table. The forward route is set active for some time indicated by a route timer entry. The default value is 3000 milliseconds, as referred in the AODV RFC [4]. If the route is no longer used, it will be deleted after the specified amount of time. Since the RREP packet is always sent back the reverse path established by the routing request, AODV only supports symmetric links.



(a) Source node S initiates the path discovery process.



(b) A RREP packet is sent back to the source.

Figure 2: AODV Path Discovery Process.

2.1.2 Maintaining Routes

If the source node moves, it is able to send a new RREQ packet to find a new route to the destination. If an intermediate node along the forward path moves, its upstream neighbor notices the move and sends a *link failure notification* message to each of its active upstream neighbors to inform them of the erasure of that part of the route (see Figure 3). The link failure notification is forwarded as long as the source node is not reached. After having learned about the failure, the source node may reinitiate the route discovery protocol. Optionally a mobile node may perform local connectivity maintenance by periodically broadcasting hello messages[3].

2.2 Dynamic Source Routing (DSR)

The Dynamic Source Routing (DSR) protocol is an on-demand routing protocol based on source routing. In the source routing technique, a sender determines the exact sequence of nodes through

[3]An AODV Hello message is defined as a RREP packet with a time to live (TTL) field of 1. Furthermore the node may send a unicast RREP or an ICMP Echo Request message to the next hop for local route maintenance.

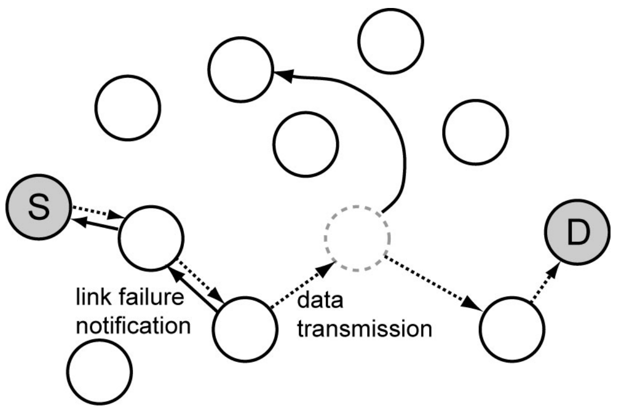


Figure 3: AODV Route Maintenance by using Link Failure Notification Message

which to propagate a packet. The list of intermediate nodes for routing is explicitly contained in the packet's header.

In DSR, every mobile node in the network needs to maintain a *route cache* where it caches source routes that it has learned. When a host wants to send a packet to some other host, it first checks its route cache for a source route to the destination. In the case a route is found, the sender uses this route to propagate the packet. Otherwise the source node initiates the route discovery process. Route discovery and route maintenance are the two major parts of the DSR protocol.

2.2.1 Route Discovery

For route discovery, the source node starts by broadcasting a *route request* packet that can be received by all neighbor nodes within its wireless transmission range. The route request contains the address of the destination host, referred to as the *target* of the route discovery [5], the source's address, a *route record* field and a unique identification number. At the end, the source host should receive a *route reply* packet containing a list of network nodes through which it should propagate the packets, supposed the route discovery process was successful.

During the route discovery process, the route record field is used to accumulate the sequence of hops already taken. First of all the sender initiates the route record as a list with a single element containing itself. The next neighbor node appends itself to the list and so on. Each route request packet also contains a unique identification number called *request_id*. request_id is a simple counter which is increased whenever a new route request packet is being sent by the source node. So every route request packet can be uniquely identified through its initiator's address and request_id. When a host receives a route request packet, it is important to process the request in the order described below. This way we can make sure that no loops will occur during the broadcasting of the packets.

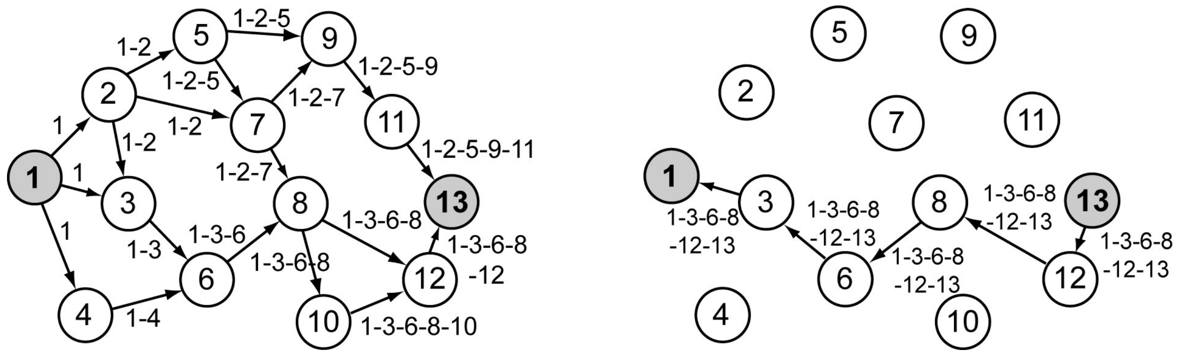
1. If the pair $\langle$ source node address, request_id $\rangle$ is found in the list of recent route requests, the packet is discarded.

2. If the host's address is already listed in the request's route record, the packet is also dis-

carded. This ensures removal of later copies of the same request that arrive by using a loop.

3. If the destination address in the route request matches the host's address, the route record field contains the route by which the request reached this host from the source node. A *route reply* packet is sent back to the source node containing a copy of this route.

4. Otherwise, add this host's address to the route record field of the route request packet and re-broadcast the packet.



(a) Building of the route record. (b) Propagation of the route reply.

Figure 4: DSR Route Discovery Process.

A route reply is sent back either if the request packet reaches the destination node itself, or if the request reaches an intermediate node which has an active route[4] to the destination in its route cache. The route record field in the request packet indicates which sequence of hops was taken. If the node generating the route reply is the destination node, it just takes the route record field of the route request and puts it into the route reply. If the responding node is an intermediate node, it appends the cached route to the route record and then generates the route reply.

Sending back route replies can be accomplished in two different manners: DSR may use symmetric links, but it is not required to. In the case of symmetric links, the node generating the route reply just uses the reverse route of the route record. When using unidirectional (asymmetric) links, the node needs to initiate its own route discovery process and piggyback the route reply on the new route request.

2.2.2 Route Maintenance

Route maintenance can be accomplished by two different processes:

- Hop-by-hop acknowledgement at the data link layer

[4]An active route towards a destination has a route cache table entry that is marked as valid. Only active routes can be used to forward data packets.

- End-to-end acknowledgements

Hop-by-hop acknowledgement at the data link layer allows an early detection and retransmission of lost or corrupt packets. If the data link layer determines a fatal transmission error (for example, because the maximum number of retransmissions is exceeded), a *route error* packet is being sent back to the sender of the packet. The route error packet contains two parts of information: The address of the node detecting the error and the host's address which it was trying to transmit the packet to. Whenever a node receives a route error packet, the hop in error is removed from the route cache and all routes containing this hop are truncated at that point.

End-to-end acknowledgement may be used, if wireless transmission between two hosts does not work equally well in both directions. As long as a route exists by which the two end hosts are able to communicate, route maintenance is possible. There may be different routes in both directions. In this case, replies or acknowledgements on the application or transport layer may be used to indicate the status of the route from one host to the other. However, with end-to-end acknowledgement it is not possible to find out the hop which has been in error.

2.3 Location-Aided Routing (LAR)

Ad hoc on-demand distance vector routing (AODV) and distance vector routing (DSR) that have been previously described are both based on different variations of flooding. The goal of Location-Aided Routing (LAR) described in [6] is to reduce the routing overhead by the use of location information. Position information will be used by LAR for restricting the flooding to a certain area [7].

In the LAR routing technique, route request and route reply packets similar to DSR and AODV are being proposed. The implementation in the simulator follows the LAR1 algorithm similar to DSR.

Location Information When using LAR, any node needs to know its physical location. This can be achieved by using the Global Positioning System (GPS). Since the position information always includes a small error, GPS is currently not capable of determining a node's exact position. However, differential GPS[5] offers accuracies within only a few meters.

Expected Zone When a source node S wants to send a packet to some destination node D and needs to find a new route, it first tries to make a reasonable guess where D could be located. Suppose node S knows that at time t_0 D's position was P and that the current time is t_1. Using this information S is able to determine the *expected zone* of D from the viewpoint of node S by time t_1. For instance if D traveled with an average speed v, the source node S expects D to be in a circle around the old position P with a radius $v(t_1 - t_0)$. The expected zone is only an estimate

[5]Differential GPS (DGPS) is a method of eliminating location errors in a GPS receiver. It makes use of a base station at precisely known coordinates, which computes the difference between the GPS-calculated coordinates and the known location. Thus, the error (which the base station determined) can be transmitted to other GPS receivers and used to correct the signal.

by S to determine possible locations of D. If D traveled with a higher speed than S expected, the destination node may be outside the expected zone at time t_1.

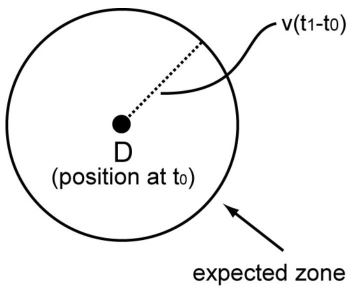


Figure 5: LAR Expected Zone.

If the source node does not know the position of D at time t_0, it will not be possible to estimate an expected zone. D could be anywhere. In this case, the entire ad-hoc network is selected as the expected zone and the routing algorithm reduces to a simple flooding.

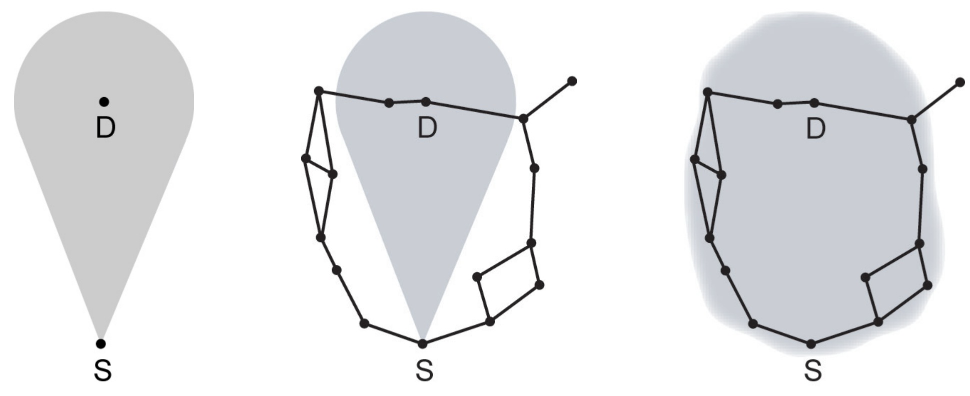
Request Zone Be S still our source node that wants to send a packet to destination node D. The *request zone* is somewhat different from the expected zone, for it defines the zone where a route request should be forwarded from. An intermediate node will forward a route request packet only, if it belongs to the request zone. This is different from the flooding protocols described before. Obviously the request zone should contain the expected zone to reach destination node D. The request zone may also include further regions:

- To create a path from S to D, both nodes must be contained in the request zone (Figure 6(a)). So if source S is not contained in the expected zone of D, additional regions need to be included. Otherwise the packet will not be forwarded from S to D.

- Under certain circumstances there may be no route from S to D, even if both nodes are contained in the requested zone (see Figure 6(b)). For instance, nodes that are near, but outside the request zone are needed to propagate the packet. Thus, after some timeout period, if no route is found from S to D, the request zone will be expanded and S will initiate a new route discovery process (Figure 6(c)). In this case, the route determination process will take longer because multiple route discoveries are needed.

2.3.1 LAR Request Zone Types

An intermediate node needs to use an algorithm to determine if it should forward a packet or not and if it is member of the request zone or not. LAR defines two different types of request zones in order to do this. LAR Scheme 1 (LAR1) was used in our simulations, it is discussed more detailed below. Further we mention LAR2 just for completeness.



(a) Request Zone. (b) Too Small Request Zone. (c) Larger Request Zone.

Figure 6: LAR - Different Request Zones.

LAR Scheme 1 (LAR1) The request zone of LAR1 is a rectangular geographic region. Remember: If source node S knows a previous location P of destination node D at time t_0, if it also knows its average speed v and the current time t_1, then the expected zone at time t_1 is a circle around P with radius $r = v(t_1 - t_0)$. The request zone now is defined as the smallest possible rectangle that includes source node S and the circular expected zone. Further should the sides of the rectangle be parallel to the x and y axes.

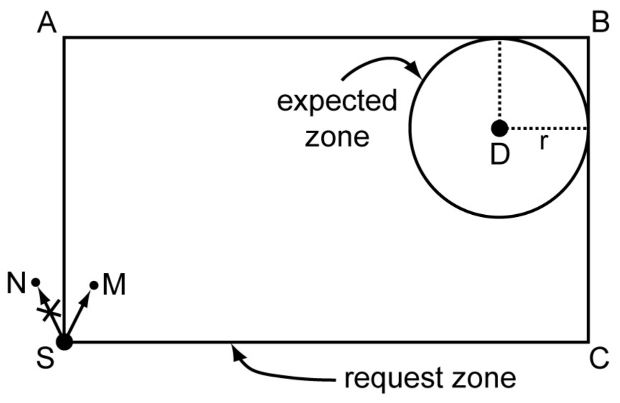


Figure 7: LAR Scheme 1 - Request Zone.

The source node is capable of determining the four corners of the rectangular request zone. This four coordinates are now included in the route request packet when initiating the route discovery process. Every node which is outside the rectangle specified by the four corners in the packet just drops the packet. As soon as the destination D receives the route request packet, it sends back a route reply packet as described in the flooding algorithms. Its reply differs by containing its current position, the actual time, and as an option its average speed. Source node S is going to use this information for a route discovery in the future.

LAR Scheme 2 (LAR2) The second LAR scheme is defined by specifying (estimated) destination coordinates (x_d, y_d) plus the distance to the destination [7]. The estimated destination and the current distance to it are included in the route request. Now, a node may only forward the route request packet if it is closer or at maximum δ farther away than the previous node. δ is a system parameter which is dependant on implementation. Every forwarding node overwrites the distance field in the packet with its own current distance to the destination. This process ensures that the packet moves towards the destination.

2.4 Zone Routing Protocol (ZRP)

In a mobile ad-hoc network, it can be assumed that most of the communication takes place between nodes close to each other. The Zone Routing Procol (ZRP) described in [8] takes advantage of this fact and divides the entire network into overlapping zones of variable size. It uses proactive protocols for finding zone neighbors (instantly sending *hello* messages) as well as reactive protocols for routing purposes between different zones (a route is only established if needed). Each node may define its own zone size, whereby the zone size is defined as number of hops to the zone perimeter. For instance, the zone size may depend on signal strength, available power, reliability of different nodes etc. While ZRP is not a very distinct protocol, it provides a framework for other protocols [9].

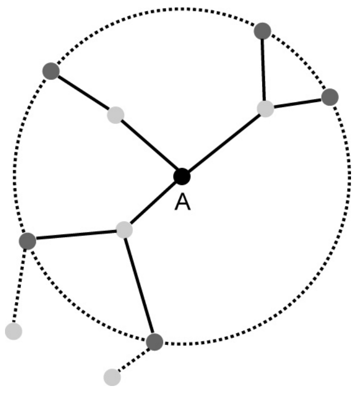


Figure 8: ZRP - Routing Zone of Node A, $\rho = 2$

First of all, a node needs to discover its neighborhood in order to be able to build a zone and determine the perimeter nodes. In Figure 8, all perimeter nodes are printed in dark gray color – they build the border of A's zone with radius $\rho = 2$. The detection process is usually accomplished by using the *Neighbor Discovery Protocol* (NDP). Every node periodically sends some *hello* messages to its neighbours. If it receives an answer, a point-to-point connection to this node exists. Nodes may be selected by different criteria, be it signal strength, radio frequency, delay etc. The discovery messages are repeated from time to time to keep the map of the neighbors updated.

The routing processes inside a zone are performed by the *Intrazone Routing Protocol* (IARP). This protocol is responsible for determing the routes to the peripheral nodes of a zone. It is generally a proactive protocol. An other type of protocol is used for the communication between different zones. It is called *Interzone Routing Protocol* (IERP) and is only responsible for routing between peripheral zones. A third protocol, the *Bordercast Resolution Protocol* (BRP) is used to optimize the routing process between perimeter nodes. Thus, it is not necessary to flood all peripheral nodes, what makes queries become more efficient. Below, the three protocols are described in more detail.

2.4.1 Intrazone Routing Protocol (IARP)

The IARP protocol is used by a node to communicate with the other interior nodes of its zone. An important goal is to support unidirectional links, but not only symmetric links. It occurs very often, that a node A may send data to a node B, but node B cannot reach node A due to interference or low transmission power for example. IARP is limited to the size of the zone ρ. The periodically broadcasted route discovery packets will be initialized with a Time To Live (TTL) field set to $\rho - 1$. Every node which forwards the packet will now decrease this field by one until the perimeter is reached. In this case, the TTL field is 0 and the packet will be discarded. This makes sure that an IARP route request will never be forwarded out of a node's zone.

As already mentioned, IARP is a proactive, table-driven protocol for the local neighborhood may change rapidly, and changes in the local topology are likely to have a bigger impact on a nodes routing behaviour than a change on the other end of the network [9]. Proactive, table-driven routing delivers a fast, efficient search of routes to local hosts. Local routes are immediately available. Therefore, every node periodically needs to update the routing information inside the zone. Additionally, local route optimization is performed. This includes the following actions:

- Removal of redundant routes
- Shortening of routes, if a node can be reached with a smaller number of hops
- Detecting of link failures and bypassing them trough multiple local hops

2.4.2 Interzone Routing Protocol (IERP)

The Interzone Routing Protocol is used to communicate between nodes of different zones. It is a reactive protocol and the route discovery process is only initiated on demand. This makes route finding slower, but the delay can be minimized by use of the Bordercast Resolution Protocol. IERP takes advantage of the fact that IARP knows the local configuration of a zone. So a query is not submitted to all local nodes, but only to a node's peripheral nodes. Furthermore, a node does not send a query back to the nodes the request came from, even if they are peripheral nodes.

2.4.3 Bordercast Resolution Protocol (BRP)

The Bordercast Resolution Protocol is rather a packet delivery service than a full featured routing protocol. It is used to send routing requests generated by IERP directly to peripheral nodes to increase efficiency. BRP takes advantage of the local map from IARP and creates a bordercast tree of it. The BRP employs special query control mechanisms to steer route requests away from areas of the network that have already been covered by the query [10]. The use of this concept makes it much faster than flooding packets from node to node.

A detailed description of the Bordercast Resolution Protocol, including its implementation is described in [10].

2.4.4 ZRP Example

In order to make the routing process and the use of the Bordercast Resolution Protocol more clear, we are considering a simple, stationary example. Figure 9 shows a graphical representation of the network, and we still suppose $\rho = 2$. Node A tries to send a packet to node V, the zone border of node A is marked with gray color.

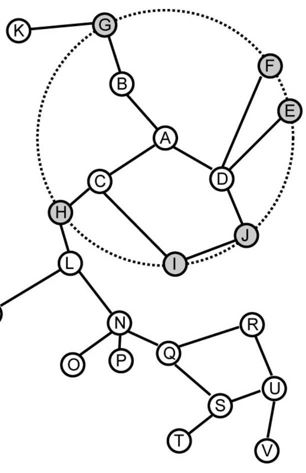
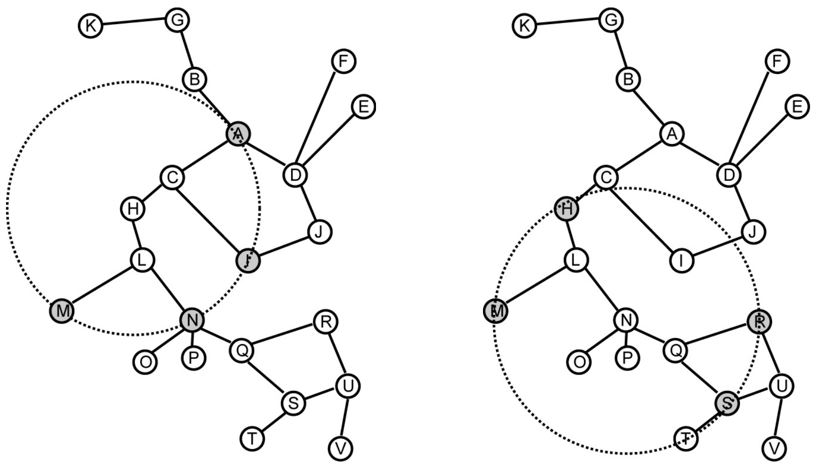


Figure 9: A Sample Network with $\rho = 2$.

First of all, node A needs to determine whether V is within its own zone or not. For this purpose, the Intrazone Routing Protocol (IARP) is used. Remember that IARP is proactive, A immediately knows that V is not within its zone and initiates a route request using the Interzone Routing Protocol (IERP). As described in section 2.4.3, IERP now uses the Bordercast Resolution Protocol (BRP) to optimize the request: The route request packet is not flooded to all nodes in A's zone, but only to the peripheral nodes E, F, G, H, I and J. From now on these nodes are searching their own routing tables for a route to the destination.

Node H does not find V in its zone either. Therefore it bordercasts the request to its peripheral nodes (see Figure 10(a)). The important thing when using BRP is that the bordercast tree of H does not contain nodes A and I: those two nodes have already been covered by the routing request. So H propagates the route request only to node M and N. As shown in Figure 10(b), N continues to bordercast the request to R and S (M and N have already received the request). Finally R and S both know V within their local zone and send back a route reply packet.



(a) Node H’s Routing Zone. (b) Node N’s Routing Zone.

Figure 10: A ZRP Routing Example.

3 Methodology

The main interest of the project was to test the ability of different routing protocols to react on network topology changes (for instance link breaks, node movement, and so on). Furthermore the focus was set on different network sizes, varying number of nodes and area sizes. Our investigations did not include the protocol’s operation under heavy load, e.g. its operation in congestion situations. Therefore only rather small packet sizes and one source node were selected.

As referenced in many other papers, we experimented with 900 seconds of simulated time over a rectangular field. However, we added 180 seconds at the beginning of the real simulation to stabilize the mobility model (see section 3.2.1). Rectangular areas were selected in order to force the use of longer routes between two nodes. Shorter routes would occur in a square field with identical node density.

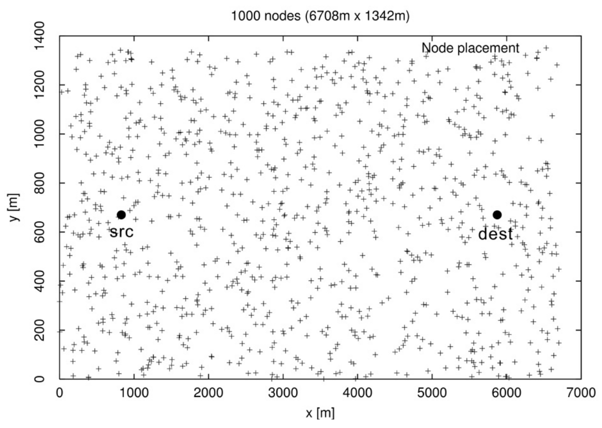


Figure 11: Node placement with 1000 nodes, rectangular simulation area and 1 source node

The simulations were performed with five different area sizes and numbers of nodes, but all with equal node density: 1500m $\times$ 300m, 50 nodes; 2121m $\times$ 424m, 100 nodes; 3000m $\times$ 600m, 200 nodes; 4743 $\times$ 949m, 500 nodes; 6708m $\times$ 1342m, 1000 nodes. In order to speed up the simulations, we only used one source and a single destination node, initially placed at $\frac{1}{8}$ respectively $\frac{7}{8}$ of the simulation field's x-axis (see Figure 11). All other nodes were randomly distributed. To make fair, direct comparisons between the different protocols, we generated a *node placement file* for each area where the exact initial position of every node was defined. Together with the random seed in the simulator's configuration file described later on, an identical movement pattern for all protocols can be guaranteed.

3.1 Simulation Environment

The simulations were performed using the QualNet Simulator v3.6 from Scalable Network Technologies, which is a commercial GloMoSim based product [11, 12]. A Sun SPARC UltraAX-MP (3 x 433 MHz, 2GB memory) built our hardware environment. While it exists a multi-threaded version of QualNet for parallel execution on multiple processors, the university version only supports a single processor.

3.1.1 QualNet

Scalable Network Technologies describes its simulator as follows: "QualNet is a discrete event simulator developed by Scalable Networks. It is extremely scalable, accommodating high-fidelity models of networks of 10's of thousands of nodes. QualNet makes good use of computational resources and models large-scale networks with heavy traffic and mobility, in reasonable simulation times."

In a discrete event simulator, the simulation is not performed in a constant time flow, but at specific points of time, when an event occurs. Such a simulator is based on an event scheduler, which contains any event that needs to be processed and stepped trough. The simulation time is increased in discrete steps to the time of the actual event whenever an event occurs. Processing an event may also produce some new events; those are inserted into the event scheduler. Every protocol in QualNet is implemented as a final state machine, which only changes its state on the occurence of an event (e.g. arrival of a packet).

The overview of a typical QualNet protocol is shown in Figure 12, in state diagram form. Every protocol starts with an initialization function, which reads external input and configures the protocol. The handling then is passed over to an event dispatcher. When an event for that layer occurs, QualNet Simulator first determines the event's protocol and hands it to the dispatcher for that protocol. The event dispatcher now checks for the type of event and calls the appropriate event handler to process it. Finally, at the end of the simulation, a finalization function is called for every protocol, to print out the collected statistics. The event to bring the simulator into finalize state is generated automatically at the end of the simulation.

QualNet Developer is a collection of five different tools:

- QualNet Simulator: The simulator itself, fast and scalable.

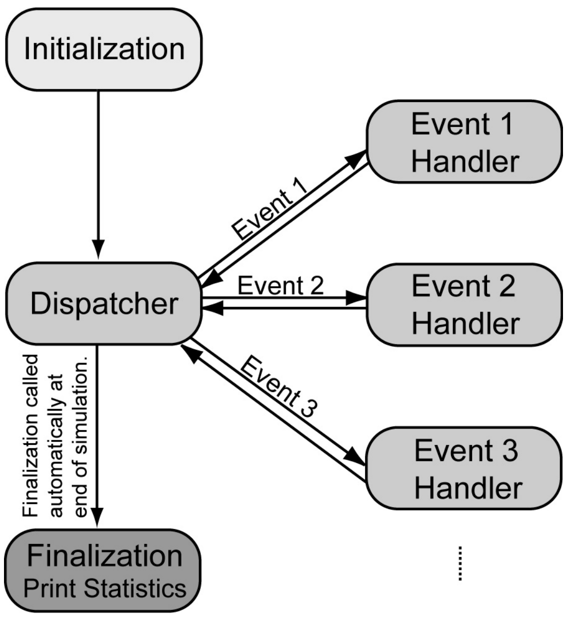


Figure 12: QualNet Event Handling.

- QualNet Animator: A graphical user interface for intuitive experiment setup and animation tool.
- QualNet Designer: A finite state machine tool for custom protocol development.
- QualNet Analyzer: A statistical graphing tool for evaluating the metrics.
- QualNet Tracer: A packet-level visualization tool for viewing the packets going up and down the protocol stack.

The simulator is fully implemented in C++ while the graphical toolkit is implemented in Java. In this project, only the simulator part was used in order to speed up the simulations. The experiments were executed using the batch mode and the according configuration files.

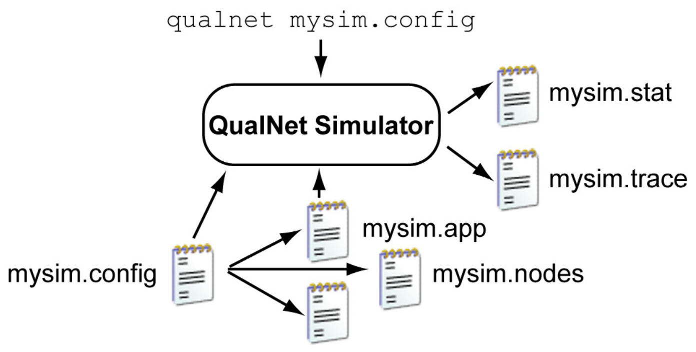


Figure 13: QualNet Simulator Batch-Mode Execution.

In the following paragraphs, the configuration files that were used and changed will be described in more detail.

mysim.config `mysim.config`[6] is the main configuration file for QualNet Simulator. It supports a huge amount of different options for every network layer: network topology, antenna and transmission parameters, frequencies, medium access control, routing models, and so on. By far the most of the parameters used in this paper were default values from QualNet. The description in this section is limited to customized values used for this project.

The first section specifies general simulation properties for a specific experiment:

```
EXPERIMENT-NAME mysim
SIMULATION-TIME   1080S
SEED   24597
COORDINATE-SYSTEM    CARTESIAN
TERRAIN-DIMENSIONS   (1500, 300)
```

- The experiment name is used for generation of the output files. Output is written to the following two files:
  - `<EXPERIMENT-NAME>.stat`, for collected statistics about the simulation.
  - `<EXPERIMENT-NAME>.trace`, for the packet-level trace file.
- Simulation time specifies the duration of the entire simulation. The following different units are supported: NS (nanoseconds), US (microseconds), MS (milliseconds), S (seconds), M (minutes), H (hours) and D (days).
- The seed value is used to initialize the random number generator. The random number generator is used in several models, e.g. for random node placement or random mobility models.
- Finally the simulation area is defined by using the Cartesian coordinate system, the terrain dimensions are indicated in meters.

Now the network topology is defined: Every node in QualNet is identified by a unique nodeId, which can range from 1 to 4’294’967’295. In the example below, a subnet with 50 nodes is generated, using 16 bits for the subnet mask (As an example N16-0 indicates 0.0.0.0/16, while N8-192.168.0.0 means 192.168/24). Node placement is initialized using a custom node placement file which is read by the simulator at startup. Additionally, we define a random waypoint mobility model (see section 3.2.1) using a pause time of 30 seconds, a minimum and maximum speed of 1 respectively 20 meters per second. Position granularity indicates how far a node may move in meters without needing to recalculate all parameters.

[6]mysim.config is just used as an example in this report, virtual any other different file name may be chosen, e.g. aodv_1000n_highmobility.config.

```
SUBNET N16-0 { 1 thru 50 }
NODE-PLACEMENT       FILE
NODE-PLACEMENT-FILE   ./mysim.nodes

MOBILITY               RANDOM-WAYPOINT
MOBILITY-WP-PAUSE      30S
MOBILITY-WP-MIN-SPEED   1
MOBILITY-WP-MAX-SPEED   20
MOBILITY-POSITION-GRANULARITY   5.0
```

Finally, a physical transmission model, a routing protocol, the application and statistical parameters need to be defined. The application layer parameters for the simulation are stored in a separate file, which is described later on. Multiple statistics exist for every different network layer and are stored in the `mysim.stat` file, which can be best viewed with QualNet Analyzer. Packet Tracing is turned off by default, because it typically requires huge amounts of disk space. However, to determine the routing overhead and other custom metrics, packet tracing must be turned on.

```
PHY-MODEL PHY802.11b
IP-FORWARDING YES
ROUTING-PROTOCOL AODV
APP-CONFIG-FILE ./mysim.app

APPLICATION-STATISTICS    YES
TCP-STATISTICS            YES
UDP-STATISTICS            YES
ROUTING-STATISTICS        YES
NETWORK-LAYER-STATISTICS  YES
QUEUE-STATISTICS          YES
MAC-LAYER-STATISTICS      YES
PHY-LAYER-STATISTICS      YES

PACKET-TRACE YES
TRACE-ALL YES
```

mysim.nodes In the `mysim.nodes` file, the placement of the nodes inside the simulation area is being specified. Each line defines a new node record. The parameters must be given in the following order:

- First the unique node identifier is indicated.
- The second parameter is for consistency with the mobility trace format, it is always set to 0.
- Next, the x, y and z coordinate of the node is defined in brackets (in meters).

- Optionally, one may also define the orientation of the node using the last two floating point parameters.

```
# NODE-PLACEMENT-FILE
# Format: nodeId 0 (x, y, z) [azimuth elevation]
1 0 (187.5, 150.0, 0.0) 0.0 0.0
2 0 (1312.5, 150.0, 0.0) 90.0 0.0
3 0 (451.446533203125, 265.402221679688, 0.0)
```

mysim.app QualNet offers different models to produce the data expected to flow across a network - they are called *traffic generators*. Models of representative applications (such as web browsing, file transfer, Telnet) can be used, but there also exist different theoretical or trace-based models. We only give a short description of each available model and limit the description of the file format to CBR, because only CBR traffic was used in the experiment setup.

- **FTP** represents the File Transfer Protocol server and client.
- **FTP/Generic** represents a more configurable model of the File Transfer Protocol server and client. The size of the items sent is not determined by network traces, but instead are specified by the user.
- **HTTP** represents a single-thread web-browser and a set of web-servers the browser will connect to. The model considers "think time" between client requests, and varying numbers of pages, items per page, and size of items, in the server responses. The client also refers variable lengths of sessions during which it makes requests of the same server for a given number of pages.
- **Telnet** represents the cleartext terminal server and client. The typing rate and sizes of server responses are taken from distributions created from network traces.
- **LOOKUP** is an unreliable query/response application that can be used to simulate applications such as DNS lookup or ping.
- **TRAFFIC-GEN** simulates a random-distribution based traffic generator.
- **TRAFFIC-TRACE** simulates a trace file-based traffic generator.
- **CBR** stands for **c**onstant-**b**it-**r**ate traffic. It is generally used to simulate multimedia traffic, or to fill in background traffic to affect the performance of other applications being analyzed.

  The file format is as follows:
  `CBR <src> <dest> <items_to_send> <item_size> <interval> <start time> <end time>`

  The example below indicates that node 1 (node with nodeID 1) will send 64 byte packets to node 2. The 0 for <items_to_send> specifies that the number of items is unlimited -

items will be sent until <end time>. Four packets will be generated per second (every 0.25 seconds), starting from 180 simulation seconds, and ending at 1080 seconds.

```
CBR 1 2 0 64 0.25S 180S 1080S
```

- **MCBR** stands for Multicast CBR (constant-bit-rate) traffic. It is similar to CBR, but sets a multicast address as the destination and rerquires multicast protocols to be running.
- **VBR** represents **v**ariable-**b**it-**r**ate traffic.
- **VoIP** is available for voice over IP applications. It requires configuration of the H323 settings.

3.2 Mobility Model

An important factor in mobile ad-hoc networks is the movement of nodes, which is characterized by speed, direction and rate of change. Mobility in the "physical world" is unpredictable, often unrepeatable, and it has a dramatic effect on the protocols developed to support node movement. Therefore, different "synthetic" types of mobility models have been proposed to simulate new protocols. Synthetic means to realistically represent node movement, but without using network traces.

In [1], seven different synthetic entity mobility models based on random directions and speeds are being discussed:

1. Random Walk Mobility Model (including its many derivates): A simple mobility model based on random directions and speeds.
2. Random Waypoint Mobility Model: A model based on random waypoints and random speeds that includes pause times between changes in destination and speed, discussed in more detail in the next section.
3. Random Direction Mobility Model: A model that forces mobile nodes to travel to the edge of the simulation area before changing direction and speed.
4. A Boundless Simulation Area Mobility Model: A model that converts a 2D rectangular simulation area into a torus-shaped simulation area.
5. Gauss-Markov Mobility Model: A model that uses one tuning parameter to vary the degree of randomness in the mobility pattern.
6. A Probabilistic Version of the Random Walk Mobility Model: A model that utilizes a set of probabilities to determine the next position of a mobile node.
7. City Section Mobility Model: A simulation area that represents streets within a city.

3.2.1 Random Waypoint Mobility Model

We used the Random Waypoint Mobility Model for our examinations, which is by far the most often used model. It was first used by Johnson and Maltz in the evaluation of Dynamic Source Routing [5], and was later refined by the same research group [13].

In this model, a mobile node moves from its current location to a randomly chosen new location within the simulation area, using a random speed uniformly distributed between $[v_{min}, v_{max}]$. v_{min} refers to the minimum speed of the simulation, v_{max} to the maximum speed. The Random Waypoint Mobility Model includes *pause times* when a new direction and speed is selected. As soon as a mobile node arrives at the new destination, it pauses for a selected time period (pause time) before starting traveling again.

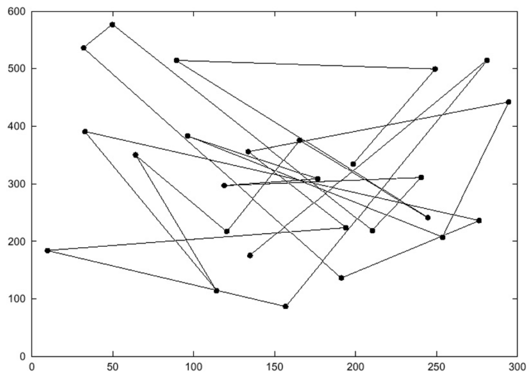


Figure 14: Traveling pattern of a mobile node using Random Waypoint Mobility [1].

This model is often simplified by using a uniformly distributed speed between $(0, v_{max}]$, and the average velocity in this case is assumed as $\frac{v_{max}}{2}$. Further more, it is expected that this average speed is maintained during the simulation process. This is not the case. The paper from Yoon, Liu and Nobel [14] examines and mathematically proves that the average speed constantly decreases and would eventually reach zero when using this model. The mobile nodes become more and more "stuck", travelling long distances at low speeds. It is clear that the results obtained by such a model will not be reliable. The simple solution suggested is to set a positive minimum speed. For instance, when setting a minimum speed of 1 m/s, a steady-state average speed settles after some time at a positive value.

As often found in other papers, but adapted to the fact above, we ran our simulation with a minimum speed of 1 m/s and a maximum speed of 20 m/s. Additionally, we chose a fixed pause time of 30 seconds. The real[7] simulation ran for 900 seconds, but we added 180 seconds in the beginning to stabilize the mobility model. So in fact the entire experiment simulation time was 1080 seconds and the first data packet was initiated after 180 seconds.

[7]"real simulation" means the period when network traffic was generated.

3.3 Application Model

To be able to make comparisons to related work (e.g. [13]), we chose the traffic source to be a constant-bit-rate (CBR) source. The sending rate was set to four packets per second, the network contained one source and one destination, and a packet size of 64 bytes was defined. As the main interest of the project was to determine if a protocol could track changes in the network topology, low packet sizes would factor out congestive effects. As described in [13], it was found that all protocols are leading to problems due to congestion when using 1024-byte packets (i.e. some nodes would drop most of the packets that they received for forwarding).

3.4 Metrics

The following four metrics have been chosen to compare the protocols:

- *Packet delivery ratio*: Packet delivery ratio is calculated by dividing the number of packets received by the destination through the number of packets originated by the application layer of the source (i.e. CBR source). It specifies the packet loss rate, which limits the maximum throughput of the network. The better the delivery ratio, the more complete and correct is the routing protocol.
- *Routing overhead*: The routing overhead describes how many routing packets for route discovery and route maintenance need to be sent in order to propagate the CBR packets. It is an important measure for the scalability of a protocol. It for instance determines, if a protocol will function in congested or low-bandwidth situations, or how much node battery power it consumes. If a protocol requires to send many routing packets, it will most likely cause congestion, collision and data delay in larger networks.
- *End-to-end delay*: End-to-end delay indicates how long it took for a packet to travel from the CBR source to the application layer of the destination. It represents the average data delay an application or a user experiences when transmitting data.
- *Hop count*: Hop count is the number of hops a packet took to reach its destination.

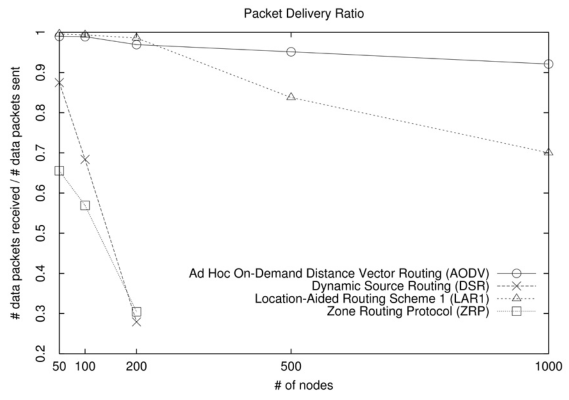
4 Simulation Results

As already mentioned in section 3, we experimented with different network sizes from 50 up to 1000 nodes. Due to severe problems with the implementation of some protocols and the speed of the simulation, we unfortunately only managed to simulate DSR and ZRP up to 200 nodes. However, conclusions can also been made for those two protocols.

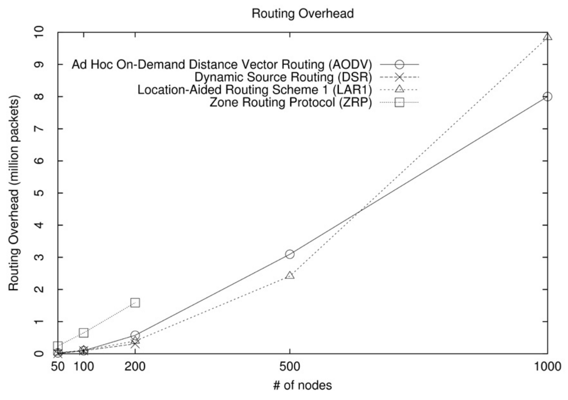
4.1 Overview

Figure 15 shows a graphical representation of the experiments results of the four different routing protocols.

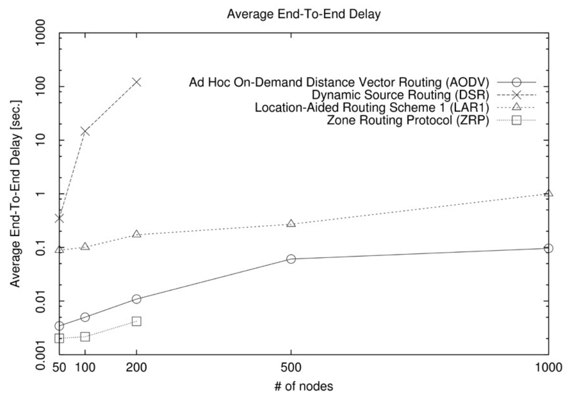
Almost all protocols perform relatively well in small networks (i.e. 50 nodes), when only few hops need to be taken to reach the destination node. Nevertheless, ZRP already at this point fails to deliver a greater percentage of the originated data packets - it only reaches a delivery ratio of 66%. As the network size grows, only AODV always manages to deliver the packets with a reliability greater than 90%. At a first glance, it can easily be stated that DSR and ZRP completely fail in larger networks: in a network of 200 nodes, the packet delivery drops below 30 percent. This fact might also be the reason for the simulation problems we experienced with those two protocols.



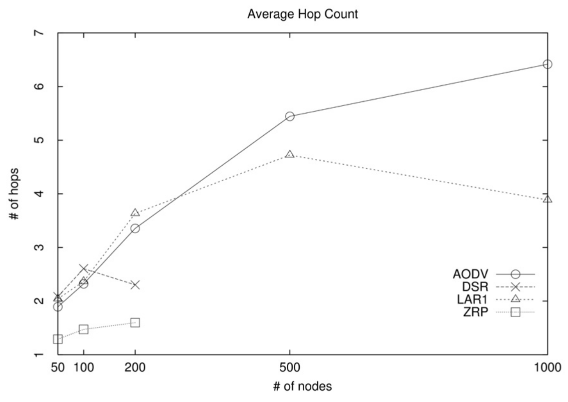
(a) Packet Delivery Ratio.



(b) Routing Overhead.



(c) Average end-to-end delay.



(d) Average hop count.

Figure 15: Comparison of the four protocols as a function of network size.

4.2 Problems

In a first section, we will give a description of more or less severe problems we experienced during our simulation. This is important because those issues may influence the interpretation of the simulation results.

- **Simulation speed and scalability**: Although Scalable Networks describes QualNet Simulator as very fast and scalable and supporting 10's of thousands of nodes, we were not able to run all simulations initially planned for this project. At first we intended to employ 30 CBR sources, as this number was often chosen in other papers. Unfortunately, we realized that the simulation is not feasible at this number of sources, because it would require huge system resources and take weeks to complete. As the university edition of QualNet only supports sequential execution on one processor, our simulator could not make use of the three processors built into our system. So we decided to redimension the project to only use a single traffic source. We also increased `MOBILITY-POSITION-GRANULARITY` by factor 5 to reduce computation expenditure when a node only moves very little (see 3.1.1). For 1000 nodes, running AODV, with one CBR source, it took QualNet about half a day to complete. To our disappointment, we were still only able to simulate the DSR and ZRP protocol up to 200 nodes. The simulator with DSR and 500 nodes crashed after more than 70 hours, most likely due to the lack of system resources.

- **Implementation errors in protocols**: Additional trouble we had in our experiments came from implementation errors in the mobile routing protocols. For instance, the simulation with DSR for 200 nodes unexpectedly crashed with a segmentation fault after some time. Or, AODV did no longer work if the "local repair" feature was turned on. We would have been interested in comparison of AODV with or without the local repair feature, which means that an intermediate node searches itself for a new route when a link break occurs. Those errors were reported to the support of Scalable Network which provided us with new updated versions of the AODV and DSR protocol.

  However, it is still not guaranteed that there are no further implementation errors in the protocols. For instance, also the Zone Routing Protocol showed a strange behaviour (see below). Reviewing all the protocols source code was out of the scope of this project. Therefore when comparing the results, the fact of possible errors must always be kept in mind.

- **No IERP packets in ZRP simulations**: As already stated in the description of the Zone Routing Protocol, this protocol makes use of three different protocols: the Network Discovery Protocol (NDP), the Intrazone Routing Protocol (IARP), and the Interzone Routing Protocol (IERP). The IERP packets are responsible for finding routes between different zones.

  When analyzing the trace files of the ZRP experiments, we found out that only NDP and IARP packets have been sent, but no IERP packets at all. This may either indicate an implementation error in the protocol, or mean that all nodes in the simulation area were included in the same zone by default. Unfortunately there was absolutely no documentation on the use of the simulator's ZRP protocol, so we were not able to set a custom zone size in the configuration files.

  Therefore, we should not rely on the results obtained for ZRP. Especially, since ZRP did not perform well in our simulations but was reported to be highly scalable (more then AODV) in other papers.

- **Only single run per scenario**: Due to the long simulation times for the larger networks and the limited scope of this project, we were only able to perform a single simulation run per scenario. This may lead to statistical errors, for instance a specific initial position and/or mobility trace is particularly good or bad for some protocol.

4.3 Packet Delivery Ratio

When looking at the packet delivery ratio (Figure 15(a)), it can easily be seen that AODV and LAR1 perform much better than DSR and ZRP. DSR and ZRP already completely fail for a network of 200 nodes, as they loose more than 70 percent of all CBR packets initiated by the source. The low delivery ratio when using DSR may be explained by the aggressive route caching built into this protocol. For a large number of nodes and higher mobility, the benefit of caching routes is completely lost - in contrary, stale routes are often chosen in DSR with higher loads. This often leads to route failures, retransmission and loss of packets. DSR reacts only slowly to route changes due to large amounts of routes in the cache.

AODV and LAR perform similarly good in networks up to 200 nodes with a delivery ratio of almost 100 percent. Here, LAR1 performs slightly better due to the additional use of location information. For larger networks however, AODV outperforms LAR1 as illustrated in Figure 15(a). For 1000 nodes, the delivery ratio in LAR1 decreases to approximately 70 percent. This might be due to the fact, that the estimated position of the destination node is less reliable in larger networks due to longer routes. For instance, the destination may have moved farther than expected and the request zone has been selected too small. As a consequence, this leads to higher packet retransmission and also higher packet loss rate.

We do not know how to explain the very bad performance of the Zone Routing Protocol. We assume most likely an implementation error in the protocol because all nodes have been assigned to the same zone.

4.4 Routing Overhead

The results concerning routing overhead (Figure 15(b)) behave as we expected: ZRP requires to send by far the most routing packets due to its proactive components, namely the frequent hello packets to update the routing table within the local zone.

On the whole, LAR1 performs a bit better than AODV by limiting the routing packets to the expected zone. The larger routing overhead for LAR1 in a network of 1000 nodes is explained by the less reliable position of the destination node in large networks and by the selection of a too small request zone. Additionally, this may also be a statistical error because LAR1 may have performed badly in this specific run.

DSR always has a lower routing load than AODV: Due to aggressive caching, DSR will most often find a route in its cache and therefore rarely initiate a route discovery process unlike AODV. But because this routes are most often not valid anymore, a lot of packets get dropped. DSR's routing overhead is dominated by route replies (unicast packets), while AODV's routing load is dominated by route requests (broadcast packets). Therefore, DSR performs very well when looking at the routing overhead.

4.5 Average End-To-End Delay

All protocols except DSR are quite effective in delivering packets when looking at the average delay from the source to the destination's application layer (see Figure 15(c)). In a network of 1000 nodes, the delay for LAR1 is still below 1 second while AODV shows even better results below 0.1 seconds. The higher delay when using location aided routing may be explained as follows: When mobility is high, more packets may travel over non-optimal routes with larger hop counts, which may be stored in a route cache. Therefore, these packets will experience longer end-to-end delay than the ones travelling over the shortest path. Best performance is reached by the Zone Routing Protocol due to the regular updates of the routing table within the zone and due to the routing optimization by the bordercast resolution protocol. DSR performs very bad, with an average delay time of about 121 seconds in a network of 200 nodes. When we take a look at extremal values, the maximum delay amounts to 429 seconds and 23 hops! However, these results compared to the packet delivery ratio in Figure 15(a) are not too astonishing. DSR often uses stale routes due to the large route cache, which leads to frequent packet retransmission and extremely high delay times.

4.6 Average Hop Count

As illustrated in Figure 15(d), the results are a bit weird: for the LAR1 and DSR protocol, the average hop count decreases as the network size grows. Moreover, if we compare the DSR graph to the average end-to-end delay, the delay heavily rises whilst the hop count decreases. We assume that these inconsistencies were produced in a particular simulation run, i.e. we expect a statistical error. The source and the destination node may have been in a relatively near position. The Zone Routing Protocol reaches the best average hop count, due to use of the Bordercast Resolution Protocol.

5 Related work

We would like to introduce some other papers about simulation results and protocol comparison for AODV and DSR, although those experiments used a far different experiment setup. Almost always the network protocols were simulated as a function of pause time (i.e. as a function of mobility), but never as a function of network size. There are no previously released paper about wireless protocol comparison in larger networks we are conscious of.

5.1 Broch, Maltz, Johnson, Hu and Jetcheva

An earlier protocol performance comparison was carried out by Broch, Maltz, Johnson, Hu and Jetcheva [13], who conducted experiments with DSDV[8], TORA[9], DSR and AODV. The simulations were quite different for they used a constant network size of 50 nodes, 10 to 30 traffic

[8]Destination-Sequenced Distance Vector Routing

[9]Temporally-Ordered Routing Algorithm

sources, seven different pause times and various movement patterns. The *ns-2* discrete event simulator [15] developed by the University of California at Berkeley and the VINT project [16] was extended to correctly model the MAC and physical-layer behaviour of the IEEE 802.11 wireless LAN standard.

DSDV performed well and delivered almost any packet in low mobility scenarios, i.e. when the node mobility rate and movement speed are low, but it failed as the mobility increased. TORA appeared as the worst performer concerning routing packet overhead. However, it still delivered more than 90% of the packets in scenarios with 10 or 20 sources. DSR showed a very good performance at all mobility rates and speeds. At last, AODV performed almost as well as DSR in all scenarios, it is also able to eliminate DSR's source routing overhead, but it was found producing more routing overhead than DSR at high rates of node mobility.

Unfortunately, our results can hardly be compared to this paper, for we only used a single traffic source and did not investigate the influence of different mobility rates and speeds. Generally, it can be said that AODV also produced a larger packet overhead than DSR, but DSR performed far worse then AODV in our investigations in terms of packet delivery ratio. As already described, DSR completely failed in larger networks.

5.2 Mingliang, Tay and Long

DSR and DSDV were simulated and compared to a newly developed Cluster-based Routing Protocol (CBRP) by Mingliang, Tay and Long [17]. The simulations were performed with pause times from 0 to 600 seconds and with 25 to 150 mobile nodes.

The focus of this presentation is set to CBRP, specially how it scales in larger networks and in situations with higher mobility. It can be seen that the packet delivery ratio of DSR falls to approximately 65% in a network of 150 nodes, which is good comparable to our results. CBRP performed much better with a delivery ratio always greater then 90 percent and a lower routing overhead than DSR in larger networks.

5.3 Das, Perkins, Royer and Marina

In February 2001 Das, Perkins, Royer and Marina [18] presented a performance comparison of the AODV and DSR protocol. The experiments were based on the *ns-2* simulator, IEEE 802.11 MAC layer, a radio model similar to Lucent's WaveLAN radio interface and random waypoint mobility with pause times from 0 to 500 seconds. In two different scenarios, 50 and 100 nodes were utilized with an area size of 1500m/300m respectively 2200m/600m.

Although the results can not directly be compared to this paper, it was also concluded that AODV outperforms DSR in more stressful situations (i.e. larger network, higher mobility). In high mobility scenarios with low pause times, DSR performed badly due to the frequent use of stale routes and slow reaction to link changes. This leaded to poor delay and delivery ratio. DSR only showed advantage in the general lower routing overhead and in low mobility and small load scenarios.

6 Conclusions

In this paper, a performance comparison of four different mobile ad-hoc routing protocols (AODV, DSR, LAR1 and ZRP) was presented as a function of network and area size. Different kinds of protocols are included in this comparison, as we have on-demand vs. hybrid routing (ZRP), hop-by-hop vs. source routing, and location aided routing. In the last few years, there were several performance examinations of such routing protocols, although the performance was almost always evaluated as a function of mobility rate and speed without considering the network size.

Scalability is a very important factor for mobile ad-hoc network protocols, as it determines if a protocol will function or fail when the number of mobile users increases. However, there are still very few papers published in this subject area. We assume this might be due to the huge amount of system resources and processing power such a large scale simulation requires. We used QualNet simulator, which is commercial and said to be faster than *ns-2* for instance. However, the simulation speed was still slow and we were only able to perform a single run per scenario in the context of this project. Therefore, those results should be validated through multiple, additional simulation runs in a future work. Also, different initial node position patterns, more sources, additional metrics (such as path length difference from shortest) could be used.

As a result of our studies, it can be said that DSR performs very poor in larger networks, as it shows extreme high delays and delivers less than 30% of all packets in a network of 200 nodes. The performance of AODV was very good in all network sizes, even though the routing overhead is higher than in DSR. LAR1 is even better than AODV up to 200 nodes in terms of delivery ratio and routing overhead, but the delivery ratio then decreases to 70 percent. It must be added that the comparison between LAR1 and the other protocols is not so fair, as LAR1 additionally uses geographic information. This data could even be used more strongly for routing purposes, as in the GPSR [19] or BLR [20] protocol for instance. A comparison of LAR1 with those protocols would be more convenient. Unfortunately, we cannot take a conclusion for ZRP due to the missing IERP packets. Those results will need to be validated in a future experiment.

References

[1] T. Camp, J. Boleng, and V. Davies, "A survey of mobility models for ad hoc network research," *Wireless Communications and Mobile Computing (WCMC): Special issue on Mobile Ad Hoc Networking: Research, Trends and Applications*, vol. 2, no. 5, pp. 483–502, 2002.

[2] C. Perkins and P. Bhagwat, "Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers," in *Proceedings of the ACM SIGCOMM'94 Conference on Communications Architectures, Protocols and Applications*, London, UK, August 1994, pp. 234–244.

[3] C. E. Perkins and E. M. Royer, "Ad-hoc on demand distance vector routing," in *Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications (WMCSA'99)*, vol. 3, New Orleans, LA, USA, February 1999, pp. 90–100.

[4] C. Perkins, E. Belding-Royer, and S. Das, "RFC 3561: Ad hoc on-demand distance vector (AODV) routing," July 2003, category: experimental. [Online]. Available: ftp://ftp.isi.edu/in-notes/rfc3561.txt

[5] D. B. Johnson and D. A. Maltz, "Dynamic source routing in ad hoc wireless networks," in *Mobile Computing*, T. Imielinski and H. Korth, Eds. Kluwer Academic Publishers, 1996, vol. 353, ch. 5, pp. 153–181.

[6] Y.-B. Ko and N. H. Vaidya, "Location-aided routing (LAR) in mobile ad hoc networks," in *Mobile Computing and Networking (MOBICOM'98)*, Dallas, TX, USA, 1998, pp. 66–75.

[7] M. Mauve, J. Widmer, and H. Hartenstein, "A survey on position-based routing in mobile ad-hoc networks," *IEEE Network Magazine*, vol. 15, no. 6, pp. 30–39, November 2001.

[8] Z. Haas, M. Pearlman, and P. Samar, "The zone routing protocol (ZRP) for ad hoc networks," July 2002, IETF Internet Draft, draft-ietf-manet-zone-zrp-04.txt.

[9] J. Schaumann, "Analysis of the zone routing protocol," December 2002. [Online]. Available: http://www.netmeister.org/misc/zrp/zrp.pdf

[10] Z. J. Haas, M. R. Pearlman, and P. Samar, "The bordercast resolution protocol (BRP) for ad hoc networks," Internet Engineering Task Force (IETF), July 2002, IETF Internet Draft, draft-ietf-manet-zone-brp-02.txt. [Online]. Available: http://www.ietf.org/proceedings/02nov/I-D/draft-ietf-manet-zone-brp-02.txt

[11] Scalable Network Techologies, "Qualnet simulator," Software Package, 2003. [Online]. Available: http://www.qualnet.com

[12] UCLA Parallel Computing Laboratory, University of California, "GloMoSim Scalable Mobile Network Simulator," Software Package, December 2000. [Online]. Available: http://pcl.cs.ucla.edu/projects/glomosim/

[13] J. Broch, D. A. Maltz, D. B. Johnson, Y.-C. Hu, and J. Jetcheva, “A performance comparison of multi-hop wireless ad hoc network routing protocols,” in *Proceeedings of the 4th Annual ACM/IEEE International Conference on Mobile Computing and Networking (MOBICOM’98)*, October 1998, pp. 85–97.

[14] J. Yoon, M. Liu, and B. Noble, “Random waypoint considered harmful,” in *Proceedings of INFOCOM*, 2003.

[15] Information Sciences Institute, “NS-2 network simulator,” Software Package, 2003. [Online]. Available: http://www.isi.edu/nsnam/ns/

[16] “The VINT Project,” USC/ISI, Xerox PARC, LBNL, and UC Berkeley, 1997. [Online]. Available: http://www.isi.edu/nsnam/vint/

[17] J. Mingliang, Y. Tay, and P. Long, “A cluster-based routing protocol for mobile ad hoc networks,” 1999/2002. [Online]. Available: http://www.comp.nus.edu.sg/~tayyc/cbrp/hon.ppt

[18] S. R. Das, C. E. Perkins, E. M. Royer, and M. K. Marina, “Performance comparison of two on-demand routing protocols for ad hoc networks,” *IEEE Personal Communications Magazine, special issue on Mobile Ad Hoc Networks*, vol. 8, no. 1, pp. 16–29, February 2001.

[19] 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)*, Boston, MA, USA, August 2000, pp. 243–254.

[20] M. Heissenbüttel and T. Braun, “BLR: A beacon-less routing algorithm for mobile ad-hoc networks,” Institute of Computer Science and Applied Mathematics (IAM), Tech. Rep., March 2003. [Online]. Available: http://www.iam.unibe.ch/~rvs/publications/TR-IAM-03-001.pdf