Real-time Audio Data Transmission over MANETs[1]

Gianmatteo Costanza

September 14, 2006

[1]`SA-2005-36`, *Semester Thesis* for Minor in Distributed Systems at the Computer Engineering and Networks Laboratory (TIK), Swiss Federal Institute of Technology (ETH) Zurich, Switzerland, under the supervision of *Prof. Bernhard Plattner, Dr. Martin May and Rainer Baumann*

Contents

Part I

Introduction

Abstract

The objective of this thesis is to assess routing protocols in their ability to cope with real-time applications in a mobile ad-hoc network. Scenarios of interest are transmissions of audio streams like webradio or voice transmissions through a gateway.
Investigation by theory brings us to the conclusion that proactive protocols like Field-Based Routing are good candidates for real-time applications in MANETs. However, cursory simulations seem not to support this finding.

Structure of the Thesis

In a first step we give a short overview of the investigated protocols, then we examine the notion of real-time and the requirements and criteria that emanate from them and model the network and communication aspects in an abstract way. In a third step we use the previously elaborated models to describe real-time criteria for every single protocol in order to finally compare each others performance in the next chapter. A rudimentary simulation of two communication models gives us further indication for assessing the protocols' adequacy. At the end we discuss the results and put them in context.

Acknowledgment

I would like to express my gratitude to all those who have helped and supported me to complete this thesis. In particular, I want to thank my supervisor Rainer Baumann who has been always available for providing inspiration and guidance whenever I asked for it. Also, I want to address my thanks to Dr. Martin May and Prof. Bernhard Plattner and all other staff members of the Computer Engineering and Networks Laboratory (TIK) who helped me advance this thesis.

Chapter 1

Routing Protocols Overview

In this section we will give a short overview of the protocols we will investigate. As proactive protocols we will have a look at DSDV (Destination-Sequenced Distance-Vector) and FBR (Field Based Routing). In the class of reactive protocols we will examine DSR (Dynamic Source Routing) and AODV (Ad-hoc On-demand Distance Vector).

1.1 Proactive Protocols

1.1.1 Destination-Sequenced Distance-Vector

DSDV [4] is a hop-by-hop distance vector routing protocol requiring each node to periodically broadcast routing updates. The key advantage of DSDV over traditional distance vector protocols is that it guarantees loop-freedom. Each DSDV node maintains a routing table listing the *next hop* for each reachable destination. DSDV tags each route with a sequence number and considers a route more favorable with a greater sequence number, or if the two routes have equal sequence numbers but has a lower metric. Each node in the network advertises a monotonically increasing even sequence number for itself.
When a node B decides that its route to a destination D has broken, it advertises the route to D with an infinite metric and a sequence number one greater than its sequence number for the route that has broken (making an odd sequence number). This causes any node A routing packets through B to incorporate the infinite-metric route into its routing table until node A hears a route to D with a higher sequence number.
Each node is required to advertise (periodically and incrementally), to each of its current neighbours, its own routing table information by broadcasting or multicasting. The entries in this table may change fairly dynamically over time, so the advertisement must be made often enough to ensure that every node can almost always locate every other node of the collection.

1.1.2 Field Based Routing

Field based routing [1] is inspired by the physical phenomenon of an electrostatic field: a service node is modelled as a (positive) point charge, and service request packets as (negative) test charges which are attracted by the service instances. This new approach suggested in [1] maps the physical model to a mobile ad hoc network in a way where each network element calculates a potential value and routes service requests towards the neighbor with the highest potential, hence towards a service instance.
The main goal of this protocol is to provide robustness rather then optimal routing. Also, it is modeled for anycast-communication where participant nodes address service nodes rather than node instances. In this scenario, it makes sense to route packets towards regions with the highest concentration of service providers, along the strongest ascent and thus avoiding loops while guaranteeing packet delivery.
The authors of this approach suggest to maintain proactively the state of service providers by periodically broadcasting advertisements. This way, every node can calculate in direction of which neighboring node the potential is the strongest. Point-to-point communication is modeled as service nodes that happen to have just a single instance, thus pointing on the most direct way to the destination node.

1.2 Reactive Protocols

1.2.1 Dynamic Source Routing

The Dynamic Source Routing protocol [3] divides the delivery of a packet into two phases: route discovery, which is done on a on-demand basis, and packet transmission.
Route discovery is done by flooding with a local broadcast, requesting the receiving intermediate nodes to add themselves into the *route request packet* and thus forming a route description to the destination node (target). Once the destination is reached, the discovery phase ends and a confirming packet (route reply) is sent back to the source node (initiator) by reversing the order of the newly discovered route.
Packet transmission is done by enclosing the complete route information into its header and make intermediate nodes forward it along the prescribed path.
An optimization is to eavesdrop forwarded packets in order to build a local cache with routing information. This cache would be maintained (periodically refreshed) and consulted before generating any route requests.
The key advantage of this protocol is that intermediate nodes do not need to maintain up-to-date routing information in order to route packets to the destination. This eliminates the need for periodic route advertisement present

in other (proactive) protocols.

1.2.2 Ad-hoc On-demand Distance Vector

AODV [6] is essentially a combination of both DSR and DSDV. It borrows the basic on-demand mechanism of Route Discovery and Route Maintenance from DSR, plus the use of hop-by-hop routing, sequence numbers, and periodic beacons from DSDV.

When a node S needs a route to some destination D, it broadcasts a *route request* message to its neighbors, including the last known sequence number for that destination. The *route request* is flooded in a controlled manner through the network until it reaches a node that has a route to the destination. Each node that forwards the *route request* creates a reverse route for itself back to node S. When the *route request* reaches a node with a route to D, that node generates a *route reply* that contains the number of hops necessary to reach D and the sequence number for D most recently seen by the node generating the *reply.* Each node that participates in forwarding this *reply* back toward the originator of the *route request* (node S), creates a forward route to D. The state created in each node along the path from S to D is hop-by-hop state; that is, each node remembers only the next hop and not the entire route, as would be done in source routing.

In order to maintain routes, AODV normally requires that each node periodically transmit a *hello* message. Failure to receive three consecutive *hello* messages from a neighbor is taken as an indication that the link to the neighbor in question is down. Alternatively, the AODV specification briefly suggests that a node may use physical layer or link layer methods to detect link breakages to nodes that it considers neighbors. When a link goes down, any upstream node that has recently forwarded packets to a destination using that link is notified via an *unsolicited route reply* containing an infinite metric for that destination. Upon receipt of such a *route reply*, a node must acquire a new route to the destination using Route Discovery as described above.

Other protocols

Several additional protocols like *Optimized Link State Routing*, *Topology Broadcast based on Reverse-Path Forwarding* and *Temporally-Ordered Routing Algorithm* have been investigated for this thesis. Due to a lack of time they have been excluded from thorough analysis. However, their basic functioning and purpose are explained in the appendix (see sections 9.1, 9.2 and 9.3).

Chapter 2

Real-time Requirements

In this thesis we want to focus primarily on real-world applications where audio streams are to be delivered in a real-time fashion. But what does this mean exactly? There is a lot of literature about *Quality of Service (QoS)* investigating and analyzing methods for ensuring traffic contracts, queueing and reservation systems, especially for high-bandwidth networks, that all do not target the center of our interest.

2.1 Implications of Real-time Guarantees

If we direct our attention to a real-world application in a MANET environment, like radio streams or voice calls, we realize that other features are also essential for a good user experience. Ensuring bandwidth and timely and ordered packet delivery is not enough. A quick establishment of communication and adaptation to the participant's mobility is as important as a gentle usage of scarce network resources.
Although features like delay, jitter, bandwidth and loss tolerance are of main interest to solid real-time transmissions, QoS traditionally makes no statement about creation and repair time of traffic paths, nor is overhead a main concern.

2.2 Hypothesis of feature distinction

As a consequence of the above explained inadequacy of QoS criteria for capturing quality and performance of real-time audio transmissions, we stipulate that real-time criteria do overlap with QoS but include additional, separate features not considered in QoS. As shown in figure 2.1, we establish that setup or creation time, repair time and overhead are required features specific to real-time applications.

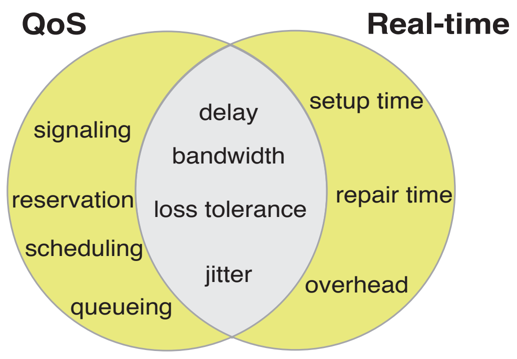

Figure 2.1: **QoS and Real-time features**

2.3 Main criteria for comparison

In order to compare the different existing routing protocols, it is useful to identify a set of meaningful properties with respect to the chosen context (chapter). We will concentrate on real-time-specific features and consider common features (Qos) only incidentally. Thus, we define the following distinguishing criteria:

2.3.1 Creation time

Definition 1 *Estimated average time for initiating a new point-to-point connection between two arbitrary peers in the MANET.*

2.3.2 Repair time

Definition 2 *Estimated average time for re-establishing a point-to-point connection between two arbitrary peers in the MANET in the case that an arbitrary rupture in the routing path occurs.*

2.3.3 Overhead

The overhead is generally measured as the quotient between the effective and the necessary resources used in order to perform a task. There are two kinds of overhead to be observed; packet and network overhead.

$$overhead = \frac{effective}{optimum}$$

Definition 3 *In the case of network overhead we measure the quotient between sent and minimum required packets. At a packet level the quotient is between metadata and payload.*

$$overhead_{network} = \frac{packetssent}{packetsrequired}$$

$$overhead_{packet} = \frac{metadata}{payload}$$

The packet overhead derives from the fact that routing-specific metainformation may be transported within each packet. Network overhead may be caused by route advertisement and other kind of information exchange between nodes.

2.4 Network Model

Networks have important properties that influence their behavior dramatically. In real life they are influenced by aspectes like topology, the shapes and texture of the surrounding buildings and objects, the intensitiy of its usage and a lot more elements.
For our theoretical approach, we need to express such properties in a more descriptive and parametric way, thus building a model upon which our reasoning can take place. Due to its natural complexity, a whole set of assumptions is made in order to ease the following analysis.

Characteristics of a network

n: number of mobile nodes

p_h: probability of successful transmission from one node to another (one hop)

p_m: probability of link loss between two neighbouring nodes due to mobility

C: connectivity, degree of density

Network size

The number of nodes present in a MANET should relate to a real-life situation. MANETs are probable to appear in conference situations, class rooms or other limited locations. Hence, we should not expect thousands of nodes, but rather a couple of dozens. If nothing else is specified, $n = 50$ is assumed.

Connectivity

Connectivity C is a statistical distribution of the neighbour count of a node. This way, C describes how dense a network graph is. Its expectation value (node's average neighbour count) is $\mathrm{E}[C] = c$. Networks with sparse connectivity $c \approx 1$ or less are likely to be partitioned, while dense connectivity $c \gg 1$ may suffering from heavy interference. If nothing else is specified, a gaussian normal distribution is assumed $C \sim \mathcal{N}(c, \sigma^2)$.

Definition 4 *A network model is a tuple (n, p_h, p_m, C)*

Average path length

In general, the distance of nodes between sender and recipient is a function of n, C and potentially other network parameters. Random graphs with given expected degrees are shown in [11] to have an average path length l of order

$$l(n, C) = \log n / \log \tilde{d}, \tilde{d} = \sum w_i^2 / \sum w_i$$

where $\tilde{\mathrm{d}}$ is hugely simplified in our case because we assume a constant degree of density c for each node.

The optimal expected average path length (shortest path) in a network with randomly distributes nodes and Connectivity C thus is

$$l_{opt} = \frac{\log n}{\log c} = \log(n - c) \tag{2.1}$$

Depending on the protocol's ability to find an optimal path from source to destination, the path length l for a transmission may be suboptimally longer.

Mobility

Mobility M is modeled by introducing p_m as the probability that two neighbouring nodes are getting departed so far as not to be able to receive the packet properly.

Possibly, p_m can be perceived as time aware if a node in motion is likely to continue travelling through the MANET. However, we assume memorylessness and state an exponential distribution, easing up the following analysis without loss of generality.

$$M \sim Exp(\lambda) \tag{2.2}$$

Single hop transmission

The probability p_h models successful transmission between *connected* nodes. Due to interference effects, it decreases with increased connectivity C and network utilization U. For the sake of simplicity, we assume that the network disposes of sufficient bandwidth such that U does not influence p_h particularly.

$$p_h(c,k) = e^{-kc}, k \in (0, \infty)$$

If a node has no neighbours able to interfere, then it can send a packet successfully (though nobody would be in the range to receive that nice packet). The damping constant k indicates how fast the situation worsens when an increased number of nodes joins the neighbourhood. For our convenience, we assume k such as to approximate a real-world situation with the probability of ten nodes transmitting successfully with $p = 0,99$.

$$0.99 \approx e^{-k10} \Rightarrow k = 0.001$$

The probability p_h converges asymptotically to zero for a connectivity striving to infinity.

$$p_h(c) = e^{-0.001c} \tag{2.3}$$

A low probability p_h could be a considerable cost factor for protocols that fail in finding an optimal path l.

Successful multi-hop packet transmission

A transmission from sender S to destination D is successful depending on the amount of hops it has to pass. For k hops the probability p_t is

$$p_t(hops = k) = (p_h \cdot \overline{p_m})^k = (p_h \cdot (1 - p_m))^k$$

A simple transmission attempt would hence not suffice for a decent success probability. Actual operating devices may cope with that fact by retrying several times, though they may implement different policies for retries. Throughout this thesis, we assume that a sufficient number of retrial will be attempted such as to statistically succeed in 99 percent of the cases.

$$retries = \lceil \frac{\log(1-0.99)}{\log(1-p_t)} \rceil = \lceil \log(p_t - 0.99) \rceil$$

For calculating the effective transmission costs (using the number of packets as metric) we need to multiply the payload by the retry factor formulated above, and then we need to consider possible repair costs to the network that happen during the transmission due to mobility. Let us just presume that we know a repair function is necessary after a certain interval of n packets.

$$packets = payload \cdot retries$$

$$\begin{aligned} cost &= packets + \frac{packets}{n_{interval}} cost_{repair} + \frac{\frac{packets}{n_{interval}} cost_{repair}}{n_{interval}} cost_{repair} + \ldots \\ &= packets + packets \frac{cost_{repair}}{n_{interval}} + packets \frac{cost_{repair}}{n_{interval}} \frac{cost_{repair}}{n_{interval}} + \ldots \\ &= \sum_{i=0}^{\infty} packets (\frac{cost_{repair}}{n_{interval}})^i = \frac{packets \cdot n_{interval}}{n_{interval} - cost_{repair}} \end{aligned}$$

Especially for proactive protocols, network maintenance during transmission may added in. The effective transmission cost is modeled as such:

Corollary 1

$$cost_{transmission} = cost_{create} + \frac{packets \cdot n_{interval}}{n_{interval} - cost_{repair}} + cost_{network\ maintenance}$$

Caches

Some protocols may use caches for for storing information that could accelerate the execution of some tasks. A cache forms a Bernouilli process with the probability of a cache hit p_{cache}.

$$Cache \sim Bernoulli(p_{cache})$$

The quality of a cache depends on its storage capacity, the size and connectivity of the network and the degree of mobility. The number of nodes n_{sniff} that a single node can be made aware of, assuming it is sniffing source and destination information for a transmission he is involved with, can be constrained as followed, depending on the actual network topology; the upper bound is the case that the observing node is the source for c distinct paths where each path has no common nodes with the other paths. Given the average path length l_{opt}, the upper bound is

$$n_{sniff_{max}} = c \log(n-c) - c + 1$$

The other extreme case is that the observing node is in the middle of overlapping paths which distinguish among themselves by just one node. For this case, the lower bound of cachable nodes is

$$n_{sniff_{min}} = \log(n-c) + c - 1$$

Hence, the number of nodes an observing node can be aware of by sniffing its traffic must be somewhere between those two extremal cases. For the sake of simplicity we suggest to use the arithmetic mean.

$$n_{sniff} = \frac{1}{2}(n_{sniff_{max}} + n_{sniff_{min}}) \approx \frac{c}{2} \log(n-c)$$

The chance to have a cache hit is $\frac{n_{sniff}}{n}$. Now we must cope with the fact that the cache entry can be invalidated due to changes in the network (mobility). We stipulate that the the entry validity has the probability $\overline{p_m}$. Throughout this thesis, we assume that cache memory is unlimited. The hit rate for a cache lookup, when used, is

$$p_{cache} = \frac{c}{2n} \log(n-c)\overline{p_m}$$

2.5 Network topologies

We now posses the means to express a concrete configuration of a network in a formal way. In order to illuminate a protocol's behavior, we formulate three prototypical network configurations:

1. stable
2. volatile
3. congested

2.5.1 Stable network

The network has a stable topology. Only a few nodes enter or leave the network. Mobility is present but at a low pace.

Corollary 2 *A* stable network *SN is is the tuple (n, p_h, p_m=0.001, c=3)*

2.5.2 Volatile network

The network is characterized by nodes frequently joining and leaving the network and a high mobility of the participating nodes which may be attached to automotive devices. The node's power saving policy may force it to fall into sleep mode as often as possible while periodically reattaching itself to the network.

Corollary 3 *A* volatile network *VN is is the tuple* $(n, p_h, p_m = 0.05, c{=}3)$

2.5.3 Congested network

The network is characterized by a moderate mobility in a dense environment; a node is connected to many others and the network graph is very compact.

Corollary 4 *A* congested network *CN is the tuple* $(n, p_h, p_m{=}0.01, c{=}10)$

2.6 Communication Model

A session between two nodes can manifest itself in many shapes and in general, a connection between two nodes lasts for more than one single packet transmission. Hence, we need to model communication and characterize it with the following parameters:

Communication Parameters

g: number of packets being transmitted from source to destination

b: bidirectionality {0,1}

Definition 5 *A communication model is a tuple (g, b).*

2.7 Communication modes

As we did before for the network model, we can express a concrete communication type in a formal way. We can model now two basic scenarions of interest described in chapter declaring two instances of the model:

- phone calls
- radio broadcast

2.7.1 Unidirectional Audio Broadcast

Corollary 5 *An unidirectional audio communication AC is modelled by ($g \in \{1600, 57600\}$, $b = 0$, p_l=0.01).*

The prototypical stream has a bandwidth of 64 KBit/s and a packet size of 1500 Bytes (typical MTU) and is listened to for 5 minutes (one song) or 3 hours (continuous consumption). The signal is encoded with no redundancy, thus susceptible to packet losses.

$$g = \frac{64000\frac{Bit}{s} \cdot \{5 \cdot 60, 3 \cdot 60 \cdot 60\}s}{1500 \cdot 8Bit} = \{1600, 57600\} \tag{2.4}$$

2.7.2 Bidirectional Voice Stream

Corollary 6 *Whether the voice call may leave the MANET through a gateway or terminate within it, a bidirectional voice communication VC is ($g \in \{96, 288, 1152\}$, $b = 1$, p_l=0.3)*

The rationale behind g is the example of a telephone call with a duration of 1, 3 or 5 minutes with a bandwidth usage of 9,6 KBit/s (GSM quality) and a packet size of 1500 Bytes (typical MTU). The voice signal is encoded with one third of redundancy.

$$g = 2\left(\frac{9600\frac{Bit}{s} \cdot \{1, 3, 12\} \cdot 60s}{1500 \cdot 8Bit}\right) = \{96, 288, 1152\} \tag{2.5}$$

2.8 Evaluation Scheme

Summing up, our evaluation efforts span a three-dimensional space composed by the *real time criteria*, *network model* and *communication model.*

The network model describes the infrastructure, the communication model describes its usage. Both combined give us a complete idea of how audio transmission takes place and what parameters have a repercussion on its real-time properties (chapter 2.3). The latter shall be expressed in terms of the enlisted parameters for each of the investigated routing protocols (chapter 1). This task is accomplished in the next chapter.

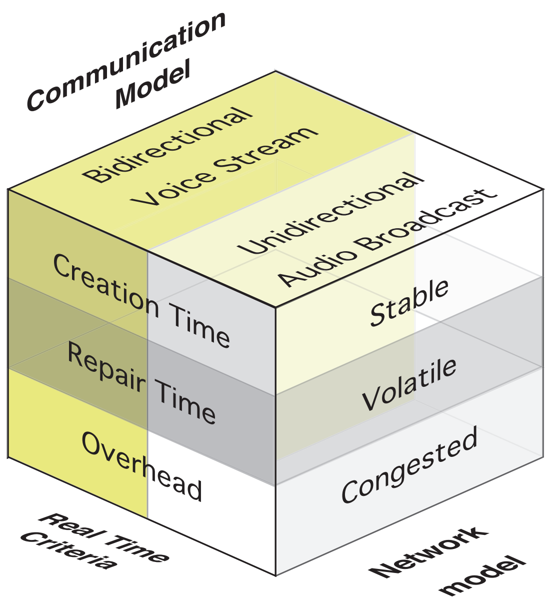

Figure 2.2: **Evaluation scheme**

Real Time criteria
Creation time
Repair time
Overhead

Network model
[SN] (n, p_h, p_m=0.001, c=3)
[VN] (n, p_h, $p_m = 0.05$, c=3)
[CN] (n, p_h, p_m=0.01, c=10)

Communication model
[AC1] ($g = 1600$, $b = 0$, p_l=0.01)
[AC2] ($g = 57600$, $b = 0$, p_l=0.01)
[VC1] ($g = 96$, $b = 1$, p_l=0.3)
[VC2] ($g = 288$, $b = 1$, p_l=0.3)
[VC3] ($g = 1152$, $b = 1$, p_l=0.3)

Figure 2.3: **Summary**

Chapter 3

Routing Protocols Properties

After having introduced the protocols of interest with a brief overview and after having established the criteria for evaluating these different protocols, we are now ready for the actual analysis.
The main objective of this chapter is to find out how each protocol behaves with respect to the three real-time criteria *creation time*, *repair time* and *overhead*. Hence, we will subdivide the analysis of each protocol accordingly.

Before starting analyse them individually, we shall list here all properties that are used throughout the next sections.

3.1 Global Properties

In the following we enumerate modeling properties that are common to all protocols like *single hop transmission probability* p_h, the *cache hit probabilities* p_{cache}, the number of *transmitted packets* g for audio and voice communication model and other statistical estimates.

$$
\begin{aligned}
p_{h_{stable}} &= e^{-0.001 \cdot c} = e^{-0.003} = 0.997 \\
p_{h_{volatile}} &= e^{-0.001 \cdot c} = e^{-0.003} = 0.997 \\
p_{h_{congested}} &= e^{-0.001 \cdot c} = e^{-0.01} = 0.99
\end{aligned}
$$

Figure 3.1: Successful single hop transmission probability p_h

$$
\begin{aligned}
p_{cache_{stable}} &= \frac{c}{2n}\log(n-c)\overline{p_m} = \frac{3}{100}\log(47)0.999 = 0.115389 \\
p_{cache_{volatile}} &= \frac{c}{2n}\log(n-c)\overline{p_m} = \frac{3}{100}\log(47)0.95 = 0.114927 \\
p_{cache_{congested}} &= \frac{c}{2n}\log(n-c)\overline{p_m} = \frac{10}{100}\log(40)0.99 = 0.36519907
\end{aligned}
$$

Figure 3.2: Cache hit probabilities p_{cache}

$$
\begin{aligned}
g_{AC_1} &= 1600 \\
g_{AC_2} &= 57600 \\
g_{VC_1} &= 96 \\
g_{VC_2} &= 288 \\
g_{VC_3} &= 1152
\end{aligned}
$$

Figure 3.3: Number of transmitted packets g for audio and voice communication model AC and VC

$$
\begin{aligned}
E[X] = 1 = n_{interval} \cdot p_m \Rightarrow n_{repair} &= \frac{1}{p_m} \\
n_{interval_{stable}} &= 1000 \\
n_{interval_{volatile}} &= 20 \\
n_{interval_{congested}} &= 100
\end{aligned}
$$

Figure 3.4: Estimated number of packets before route repair necessary due to mobility

3.2 DSDV

As previously illustrated (section 1.1.1), DSDV is a proactive protocol that periodically updates its nodes' routing tables.

3.2.1 Creation time

In the case of a proactive routing protocol, the creation of a route is one strength. Assuming every table entry is up-to-date, the route path is established immediately, making the costs of this operation weigh as much as the path length l. In case the route is invalid due to the nodes' mobility, an alternative route may be looked up, provided no network scission has occurred.
Hence, we stipulate that creation times costs as much as the expected path length

$$cost_{create} = l$$

$$\begin{aligned} cost_{create_{stable}} &\approx 3.85 \\ cost_{create_{volatile}} &\approx 3.85 \\ cost_{create_{congested}} &\approx 3.69 \end{aligned}$$

3.2.2 Repair time

Depending where on a previously working route the rupture has occured, an alternative route has to be found. In the average mean case, a rupture is expected to occur on the middle hop of a path. Hence, a repair operation needs to be started from there to the destination node. The cost for that is

$$cost_{repair} = \frac{1}{2}l$$

$$\begin{aligned} cost_{repair_{stable}} &\approx 1.93 \\ cost_{repair_{volatile}} &\approx 1.93 \\ cost_{repair_{congested}} &\approx 1.84 \end{aligned}$$

3.2.3 Overhead

Network Overhead

Depending on the update frequency of the routing tables, this protocol will produce considerable network overhead. In the optimal case, only the changes due to mobility need to be advertised.
Assuming an interval of 15 s (see [12], table I, page 3) we need to oppose that to the different bandwidth usages of our communication model. For the sake simplicity, we assume that only one single communication stream is occurring in the whole network. This way, the reasoning below leads to

a pessimistic view on the efficiency of the protocol in terms of overhead because other traffic, that would benefit from up-to-date traffic is simply disregarded.

In the case of audio streams (section 2.7.1) where a bandwidth usage of $64\frac{kBit}{s}$ is stated, equivalent to $5.\overline{3}\frac{packet}{s}$, 80 packets are sent during one update interval.

$$15s \cdot 5.\overline{3}\frac{packet}{s} = 80packet$$

Considering the different mobility probabilities p_m, the network overhead for audio communication is

$$overhead_{network_{AC}\,stable} = \frac{1}{80 * p_{m_{stable}}} = 12.5$$
$$overhead_{network_{AC}\,volatile} = \frac{1}{80 * p_{m_{volatile}}} = 1(0.25)$$
$$overhead_{network_{AC}\,congested} = \frac{1}{80 * p_{m_{congested}}} = 1.25$$

The case of a volatile is remarkable, for the overhead is lower than 1, meaning super-optimal. In fact, in this particular scenario the changes are occurring four times faster than packets are sent, and advertising every network change nonetheless wouldn't be optimal at all. The "placid" pace of the update intervals happen to behave in a very benevolent way.
In the case of voice streams (section 2.7.2) the numbers change obviously. Bandwith is at $9.6\frac{kBit}{s}$ is stated, equivalent to $0.8\frac{packet}{s}$, thus 12 packets are sent during one update interval.

$$15s \cdot 0.8\frac{packet}{s} = 12packet$$

Considering the different mobility probabilities p_m, the network overhead for voice communication is

$$overhead_{network_{VC}\,stable} = \frac{1}{12 * p_{m_{stable}}} = 83.\overline{3}$$
$$overhead_{network_{VC}\,volatile} = \frac{1}{12 * p_{m_{volatile}}} = 1.\overline{6}$$
$$overhead_{network_{VC}\,congested} = \frac{1}{12 * p_{m_{congested}}} = 8.\overline{3}$$

In general, we can note that for *stable* networks, this protocol produces the most overhead. Taking the average of above two communication scenarios, the overhead is expected to be

$$\begin{aligned} overhead_{networkstable} &= 47.92 \\ overhead_{networkvolatile} &= 1(0.96) \\ overhead_{networkcongested} &= 4.79 \end{aligned}$$

Packet Overhead

The packet overhead for regular packet delivery is minimal; it has to carry sender and destination, all routing information resides in the nodes.

Assuming an entry would require 4 Bytes (standard IP address) and considering a packet size of 1500 Bytes (communication model, 2.7.1 and 2.7.2), the effective overhead is supposed to be

$$overhead_{packet} = \frac{1500Bytes + 2 \cdot 4Bytes}{1500Bytes} = 1.005\overline{3}$$

The transmission cost calculations for the case studies AC and VC are shown in the correspondent annex 7.2.

3.3 Field Based Routing

As previously illustrated (section 1.1.2), FBR is a proactive protocol that periodically updates its nodes' routing tables. While aiming for better efficiency in service discovery it addresses routing by calculating *potentials* throughout the network.

3.3.1 Creation time

In the case of communication between specific nodes (which is equivalent to a service node with a single instance), packets are routed along the shortest path, as proven in [2] (section 3.4.1.). Hence we can use l_{opt} in our calculations without hesitation.

$$cost_{create} = l_{opt}$$

$$\begin{aligned} cost_{create_{stable}} &\approx 3.85 \\ cost_{create_{volatile}} &\approx 3.85 \\ cost_{create_{congested}} &\approx 3.69 \end{aligned}$$

3.3.2 Repair time

A particular strength of this protocol are scenarios where communication occurs between a client and a service with multiple instances. The sudden failure to reach one service instance is automatically and transparently

recovered by routing the packet to another instance of that service. This makes the routing path longer but still optimal for the current configuration of the network.
Failure occurrences below that number would have *no* creation costs at all. If more failures occur, we have no wait until service availability is advertised again later on as no explicit route discovery is provided in this protocol. In order to model this aspect, we would have to fix the number of service instances $n_{services}$ available and have a theoretical notion of when a service would be available again. Instead, lets just assume that enough service instances are available all the time and even *volatile* or *congested* network are benign enough to not partition the graph. Hence, repair costs do not occur at all.

$$cost_{repair} = 0$$

3.3.3 Overhead

Network Overhead

As described in [2] (chapter 5.2) the update interval is set to 5s.

In the case of audio streams (section 2.7.1) where a bandwidth usage of $64\frac{kBit}{s}$ is stated, equivalent to $5.\overline{3}\frac{packet}{s}$, $26.\overline{6}$ packets are sent during one update interval.

$$5s \cdot 5.\overline{3}\frac{packet}{s} = 26.\overline{6}packet$$

Considering the different mobility probabilities p_m, the network overhead for audio communication is

$$overhead_{network_{AC}\,stable} = \frac{1}{26.\overline{6} * p_{m_{stable}}} = 37.5$$
$$overhead_{network_{AC}\,volatile} = \frac{1}{26.\overline{6} * p_{m_{volatile}}} = 1(0.75)$$
$$overhead_{network_{AC}\,congested} = \frac{1}{26.\overline{6} * p_{m_{congested}}} = 3.75$$

As in 3.2.3, the overhead is lower than 1, meaning super-optimal. In this particular scenario the changes are occurring faster than packets are sent, and advertising every network change nonetheless wouldn't be optimal at all. The "placid" pace of the update intervals happen to behave in a very benevolent way.

In the case of voice streams (section 2.7.2) the numbers change obviously. Bandwith is at $9.6\frac{kBit}{s}$ is stated, equivalent to $0.8\frac{packet}{s}$, thus 4 packets are sent during one update interval.

$$5s \cdot 0.8\frac{packet}{s} = 4packet$$

Considering the different mobility probabilities p_m, the network overhead for voice communication is

$$\begin{aligned}
overhead_{network_{VC}stable} &= \frac{1}{4 * p_{m_{stable}}} = 250 \\
overhead_{network_{VC}volatile} &= \frac{1}{4 * p_{m_{volatile}}} = 5 \\
overhead_{network_{VC}congested} &= \frac{1}{4 * p_{m_{congested}}} = 25
\end{aligned}$$

In general, we can note that for *stable* networks, this protocol produces the most overhead. Taking the average of above two communication scenarios, the overhead is expected to be

$$\begin{aligned}
overhead_{network stable} &= 143.75 \\
overhead_{network volatile} &= 2.875 \\
overhead_{network congested} &= 14.375
\end{aligned}$$

Packet Overhead

The packet overhead for regular packet delivery is minimal; it has to carry information about the destination, and a sender only for bi-directional communication (2.7.2). All routing information resides in the nodes.

Assuming an entry would require 4 Bytes (standard IP address) and considering a packet size of 1500 Bytes, the effective overhead is supposed to be

$$\begin{aligned}
overhead_{packet_{ac}} &= \frac{1500Bytes + 1 \cdot 4Bytes}{1500Bytes} = 1.002\overline{6} \\
overhead_{packet_{vc}} &= \frac{1500Bytes + 2 \cdot 4Bytes}{1500Bytes} = 1.005\overline{3}
\end{aligned}$$

The transmission cost calculations for the case studies AC and VC are shown in the corresponding annex 7.2.

3.4 DSR

As previously illustrated (section 1.2.1), DSR is a reactive protocol that knows two modes of operation: route discovery and the actual packet transmission. Let us investigate the first one more closely.

3.4.1 Creation time

If a node has not yet any knowledge about the route to a destination (learnt by sniffing the surrounding traffic) and has no up-to-date entry in the routing cache from former communication attempts, a route discovery has to be initiated. In order to find the target, we *flood* the network and once reached the target, a *route reply* is sent along the newly discovered route to the initiating source. The fastest packet will reach the destination in l hops, such that a round trip takes this amount of time:

$$hops = hops_{flood} + hops_{reply} = 2l = 2\log(n-c)$$

However, every single transmission must take transmission failure and mobility into account. Assuming l to be $\log(n-c)$ (2.1), the probability p_d to discover a route in $hops$ time is

$$\begin{aligned} p_d = & \quad (p_h \cdot \overline{p_m})^{hops} = (e^{-0.001c}(1-p_m))^{2\log(n-c)} \\ p_{d_{stable}} = & \quad (e^{-0.001\cdot 3}(1-0.001))^{2\log(50-3)} = 0.969665 \\ p_{d_{volatile}} = & \quad (e^{-0.001\cdot 3}(1-0.1))^{2\log(50-3)} = 0.658313 \\ p_{d_{congested}} = & \quad (e^{-0.001\cdot 10}(1-0.01))^{2\log(50-10)} = 0.86249 \end{aligned}$$

As we have to observe, a simple round would not suffice for a decent success probability. Actual operating devices may cope with that fact by retrying several times, though they may implement different policies for retrial. We assume that a sufficient number of retrial will be attempted such as to statistically succeed in 0.99 of the cases.

$$\begin{aligned} retry = & \quad \log(\tfrac{1-p_d}{0.01}) \\ retry_{stable} = & \quad 1.1 \\ retry_{volatile} = & \quad 3.53 \\ retry_{congested} = & \quad 2.62 \end{aligned}$$

Combining these thoughts, we have a clearer idea of how much the creation phase costs.

$$cost_{create} = hops \cdot retry = \log(n-c) \cdot retry \tag{3.1}$$

$$\begin{aligned} cost_{create_{stable}} \approx & \quad 4.27 \\ cost_{create_{volatile}} \approx & \quad 13.6 \\ cost_{create_{congested}} \approx & \quad 9.67 \end{aligned}$$

3.4.2 Repair time

In case of sudden rupture of a formerly working route, an alternative has to be found for packet delivery. As a means of last resort, a route discovery as described above has to be initiated. However, DSR has been designed with some optimisations in mind; in response to a single route discovery (as well as through routing information from other packets overheard), a node can learn and cache multiple routes to any destination.

The repair mechanism itself does not differ from an ordinary route discovery, but any intermediate node may have knowledge in the cache and can thus send a *route reply* immediately.

$$cost_{repair} = P[cache\ miss] \cdot cost_{create} + P[cache\ hit] \cdot cost_{lookup}$$

$$P[cache\ miss] = \overline{p_{cache}}^{l} = (1 - p_{cache})^{\log(n-c)}$$

$$P[cache\ miss]_{stable} = 0.624$$

$$P[cache\ miss]_{volatile} = 0.625$$

$$P[cache\ miss]_{congested} = 0.187$$

$$P[cache\ hit] = \overline{P[cache\ miss]}$$

$$cost_{lookup} = E[nodes\ queried|cache\ hit] = \frac{1}{2}l$$

Thus, the average repairing cost for a route is

$$\begin{aligned} cost_{repair_{stable}} &\approx 3.39 \\ cost_{repair_{volatile}} &\approx 9.385 \\ cost_{repair_{congested}} &\approx 3.301 \end{aligned}$$

3.4.3 Overhead

Network Overhead

Network overhead may occur while discovering or repairing a route and while delivering a packet. In a perfect network, a node would notifiy its presence only once to any other node and thus avoid any supplemental traffic in the network. When delivering a packet, an optimal route (minimal numbers of intermediary nodes) would be chosen.
The behaviour of DSR is obviously not optimal for route discovery. A flooding of the network would cause $(n-1)(c-1)$ packets to be sent through the network whereas only the intermediary $l-1$ nodes between source and target should be bothered in the optimal case.

$$overhead_{network} = \frac{(n-1)(c-1)}{2l}$$

For the three different network types, the overhead is expected to be

$$\begin{aligned}
overhead_{network\,stable} &= 12.73\\
overhead_{network\,volatile} &= 12.73\\
overhead_{network\,congested} &= 59.77
\end{aligned}$$

Regarding the ability to find and use an optimal route for packet delivery, this protocol should be able to asymptotically assess the best possible route by enriching its route cache and continually calculate the shortest path.

Packet Overhead

The packet overhead for regular packet delivery is linear to its route length; every packet is provided with a list of nodes along whom the packet is forwarded to its destination. In the average, metadata in the length of l entries are inserted into a packet's header. In the optimum case, every node (the network) would know how to forward the packet and no such information would be encapsulated.

$$overhead_{packet} = \frac{payload + h \cdot entry}{payload}$$

Assuming an entry would require 4 Bytes (standard IP address) and considering a packet size of 1500 Bytes (communication model, 2.7.1 and 2.7.2), the effective overhead is supposed to be

$$overhead_{packet} = \frac{1500Bytes + 25 \cdot 4Bytes}{1500Bytes} = 1.0\overline{6}$$

The transmission cost calculations for the case studies AC and VC are shown in the correspondent annex 7.1.

3.5 AODV

As previously illustrated (section 1.2.2), AODV is a proactive protocol that periodically updates its nodes' routing tables.

3.5.1 Creation time

As previously stated in section 1.2.2, AODV is a mixture of DSR and DSDV. With respect to creation time it behaves exactly like DSR (section 3.4.1).

$$\begin{aligned}
cost_{create_{stable}} &\approx \quad 4.27\\
cost_{create_{volatile}} &\approx \quad 13.6\\
cost_{create_{congested}} &\approx \quad 9.67
\end{aligned}$$

3.5.2 Repair time

Also, in term of repair characteristics, AODV shares the same qualities of DSR. Previous thoughts (section 3.4.2) apply here as well.

$$\begin{aligned} cost_{repair_{stable}} &\approx 3.39 \\ cost_{repair_{volatile}} &\approx 9.385 \\ cost_{repair_{congested}} &\approx 3.301 \end{aligned}$$

3.5.3 Overhead

Network Overhead

Given the default update rate for route maintenance of once per second (see [7], table 2, page 6), we can calculate the specific network overhead for audio and voice communication.

In the case of audio streams (section 2.7.1) where a bandwidth usage of $64\frac{kBit}{s}$ is stated, equivalent to $5.\overline{3}\frac{packet}{s}$, $5.\overline{3}$ packets are sent during one update interval.

$$1s \cdot 5.\overline{3}\frac{packet}{s} = 5.\overline{3}packet$$

Considering the different mobility probabilities p_m, the network overhead for audio communication is

$$\begin{aligned} overhead_{network_{AC}\,stable} &= \frac{1}{5.\overline{3} * p_{m_{stable}}} = 187.5 \\ overhead_{network_{AC}\,volatile} &= \frac{1}{5.\overline{3} * p_{m_{volatile}}} = 3.75 \\ overhead_{network_{AC}\,congested} &= \frac{1}{5.\overline{3} * p_{m_{congested}}} = 18.75 \end{aligned}$$

In the case of voice streams (section 2.7.2) with bandwith at $9.6\frac{kBit}{s}$ (equivalent to $0.8\frac{packet}{s}$) 0.8 packets are sent during one update interval.

$$1s \cdot 0.8\frac{packet}{s} = 0.8packet$$

Considering the different mobility probabilities p_m, the network overhead for voice communication is

$$\begin{aligned} overhead_{network_{VC}\,stable} &= \frac{1}{0.8 * p_{m_{stable}}} = 1250 \\ overhead_{network_{VC}\,volatile} &= \frac{1}{0.8 * p_{m_{volatile}}} = 25 \\ overhead_{network_{VC}\,congested} &= \frac{1}{0.8 * p_{m_{congested}}} = 125 \end{aligned}$$

In general, we can note that for *stable* networks, this protocol produces the most overhead. Taking the average of above two communication scenarios, the overhead is expected to be

$$\begin{aligned} overhead_{networkstable} &= 718.75 \\ overhead_{networkvolatile} &= 14.375 \\ overhead_{networkcongested} &= 71.875 \end{aligned}$$

Packet Overhead

As for DSDV, the packet overhead for regular packet delivery is minimal; it has to carry sender and destination, all routing information resides in the nodes.

$$overhead_{packet} = \frac{1500Bytes + 2 \cdot 4Bytes}{1500Bytes} = 1.005\overline{3}$$

3.6 Summary

3.6.1 Creation Time

	DSR		DSDV		AODV		FBR	
$cost_{create_{stable}}$	4.27	2	3.85	1	4.27	2	3.85	1
$cost_{create_{volatile}}$	13.6	2	3.85	1	13.6	2	3.85	1
$cost_{create_{congested}}$	9.67	2	3.67	1	9.67	2	3.69	1
rank		2		1		2		1

3.6.2 Repair Time

	DSR		DSDV		AODV		FBR	
$cost_{repair_{stable}}$	3.39	4	1.93	2	3.39	4	0	1
$cost_{repair_{volatile}}$	9.39	4	1.93	2	9.39	4	0	1
$cost_{repair_{congested}}$	3.3	4	1.84	2	3.3	4	0	1
rank		4		2		4		1

3.6.3 Network Overhead

	DSR		DSDV		AODV		FBR	
$network_{stable}$	12.73	2	12.5	1	718.75	4	25	3
$network_{volatile}$	12.73	2	1	1	14.375	4	5	3
$network_{congested}$	59.77	3	1.25	1	71.875	4	25	2
rank		2		1		4		3

3.6.4 Packet Overhead

	DSR	DSDV	AODV	FBR
	$1.0\overline{6}$	$1.005\overline{3}$	$1.005\overline{3}$	$1.002\overline{6}$
rank	4	2	2	1

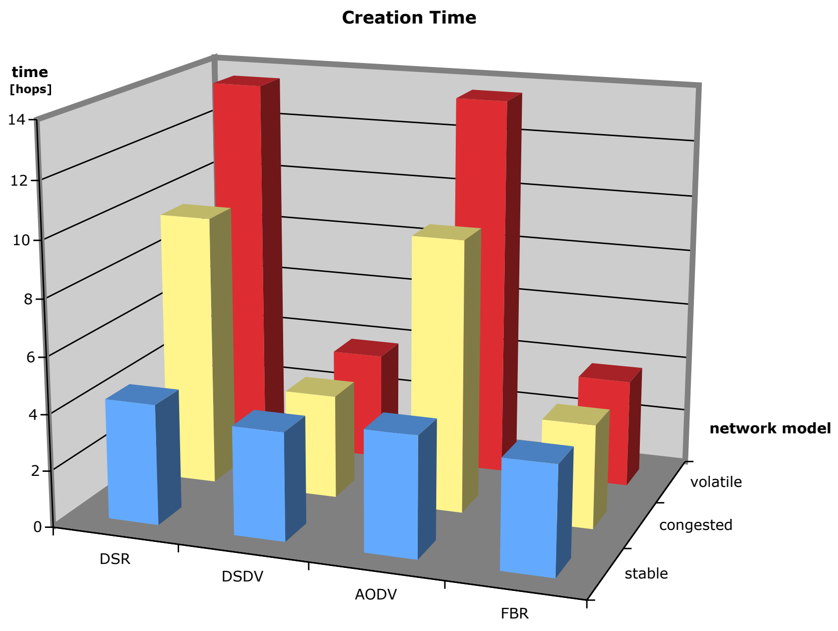

Figure 3.5: **Creation time**

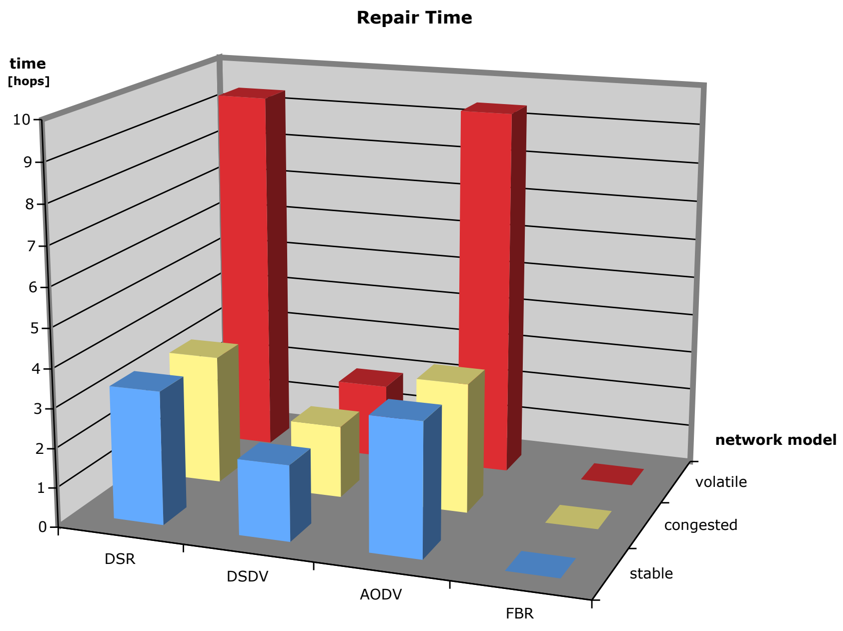

Figure 3.6: **Repair time**

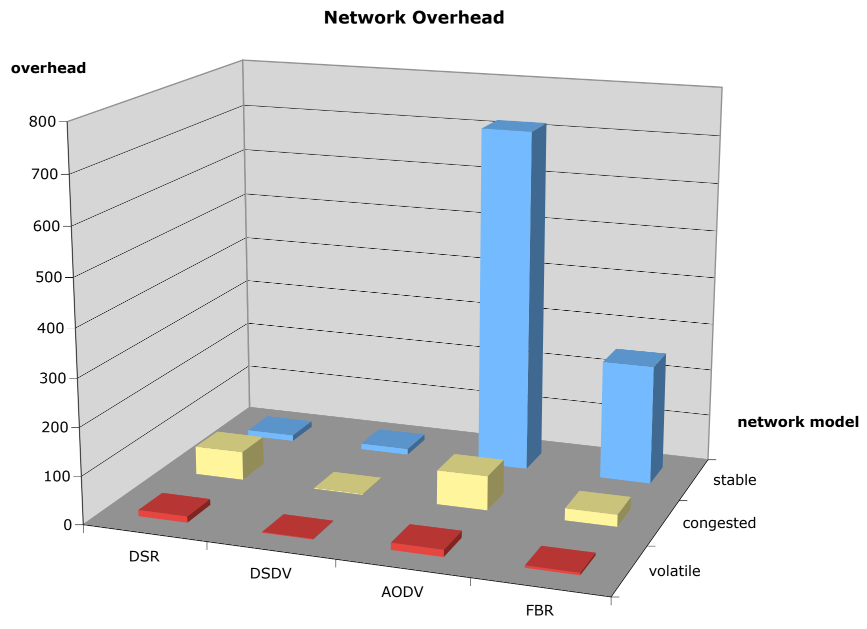

Figure 3.7: **Network overhead**

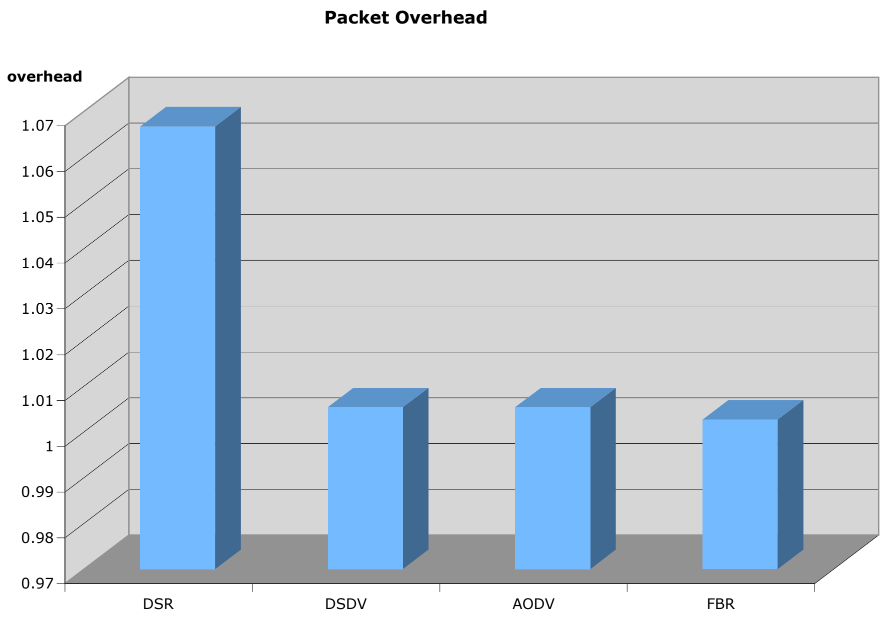

Figure 3.8: **Packet overhead**

Chapter 4

Comparison of Routing Protocols

For each context we analyse how the different protocols behave. Our reasoning is based on the calculations from chapter 7.

4.1 Unidirectional Audio Broadcast

First of all, let us compare the transmission costs of all protocols for audio communication (AC).

	DSR		DSDV		AODV		FBR	
$cost_{AC1_{stable}}$	1610	1	1627	2	1907	4	1664	3
$cost_{AC1_{volatile}}$	3029	4	1795	2	2075	3	1664	1
$cost_{AC1_{congested}}$	1665	3	1654	1	1934	4	1664	2
$cost_{AC2_{stable}}$	57801	1	58435	2	68515	4	59764	3
$cost_{AC2_{volatile}}$	108544	4	64459	2	74539	3	59764	1
$cost_{AC2_{congested}}$	59581	2	59407	1	69487	4	59764	3
average rank		3		1		4		2

As we can see, DSDV seems to perform best in case of unidirectional audio communication while FBR is remarkably well, especially in *volatile* environments.

4.2 Bidirectional Voice Streams

The transmission costs in the case of voice communication (VC) are as follows.

	DSR		DSDV		AODV		FBR	
$cost_{VC1_{stable}}$	101	1	109	2	221	4	124	3
$cost_{VC1_{volatile}}$	195	3	119	1	231	4	124	2
$cost_{VC1_{congested}}$	109	1	110	2	222	4	124	3
$cost_{VC2_{stable}}$	294	1	317	2	653	4	364	3
$cost_{VC2_{volatile}}$	557	3	347	1	683	4	364	2
$cost_{VC2_{congested}}$	308	1	322	2	658	4	364	3
$cost_{VC3_{stable}}$	1161	1	1255	2	2599	4	1444	3
$cost_{VC4_{volatile}}$	2185	3	1375	1	2719	4	1444	2
$cost_{VC3_{congested}}$	1202	1	1274	2	2618	4	1444	3
average rank		1		1		4		3

DSR and DSDV assume here the pole position. While DSR has difficulties with *volatile* networks and excels in the other scenarios, DSDV seems to cope best in the contrary case. FBR performs not too bad in comparison whereas AODV is clearly on the last position.

Chapter 5

Simulations with Glomosim

Intent and Scope of Simulations

In order to evaluate and better discuss our theoretical findings from the previous chapters, we made simulations using *Glomosim* [13], a scalable simulation environment for wireless and wired network systems developed by UCLA.

For that purpose we picked two communication models, AC1 and VC2, and made them run in *stable*, *volatile* and *congested* networks for the three in *Glomosim* available protocols: DSR, AODV and FBR.

5.1 Network Model Mapping

In chapter 2 we defined a network model as the tuple (n, p_h, p_m, C) with three different topologies (section 2.5). For establishing comparability, we had to map these three topologies into the *Glomosim* framework. The exact configuration is found in the appendix (8.1).

5.2 Results

For the two scenarios AC1 and VC2, 5 minutes of webradio and 3 minutes of a phone call (see appendix 8.2), 20 runs were performed for three protocols and three network topologies with different random seeds.

In the case of webradio simulation, we can see in the box plot for volatile networks (figure 5.3) that AODV clearly beats the contenders with nearly 9000 packets delivered of the 9600 sent while maintaining a narrow spread. DSR takes a middle position while FBR is at the last rank.

The result changes completely for congested networks (figure 5.4) where FBR and DSR perform flawlessly and AODV struggles with packet delivery.

Protocol	DSR	AODV	FBR
Stable avg (spread σ)	9599.2 (0.7)	9598.9 (0.61)	9599.9 (0.61)
Volatile avg (spread σ)	529.5 (54.92)	8679.0 (34.3)	3776.4 (39.32)
Congested avg (spread σ)	9598.3 (1.14)	6826.2 (56.11)	

Figure 5.1: **Summary Phone Call simulation**

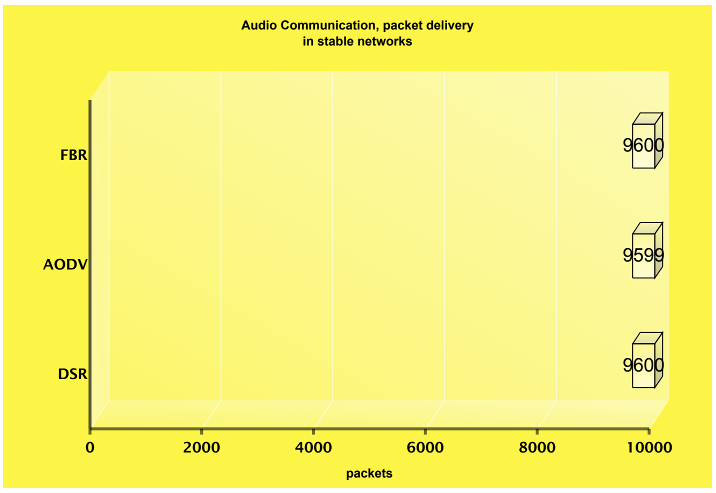

Figure 5.2: **Webradio, stable network**

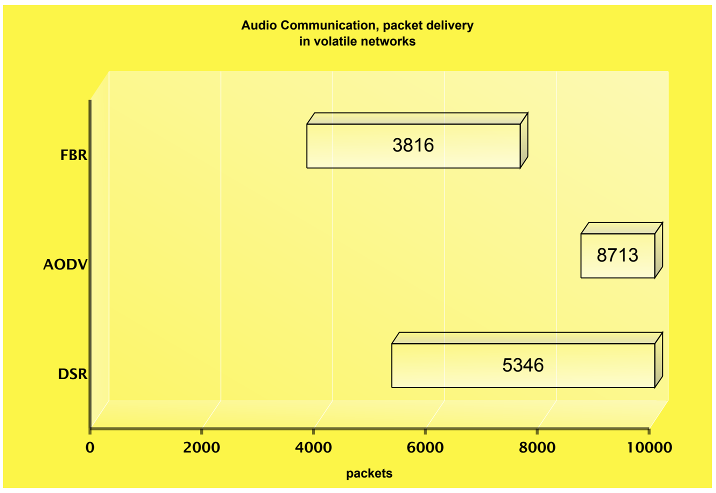

Figure 5.3: **Webradio, volatile network**

In the case of the phone call simulation, we can see in the box plot for volatile and congested networks (figure 5.7 and 5.8) that AODV is the clear dominator before DSR and FBR.
In contrast to the other scenario, FBR shows surprising difficulties in congested networks where it considerably falls behind DSR.

Generally speaking, stable networks (figures 5.2 and 5.6) are no challenge to any of the observed protocols. As we can clearly see from the box plots, AODV is the dominator with the exeption of webradio in in a congested network, and DSR behaves better than FBR.

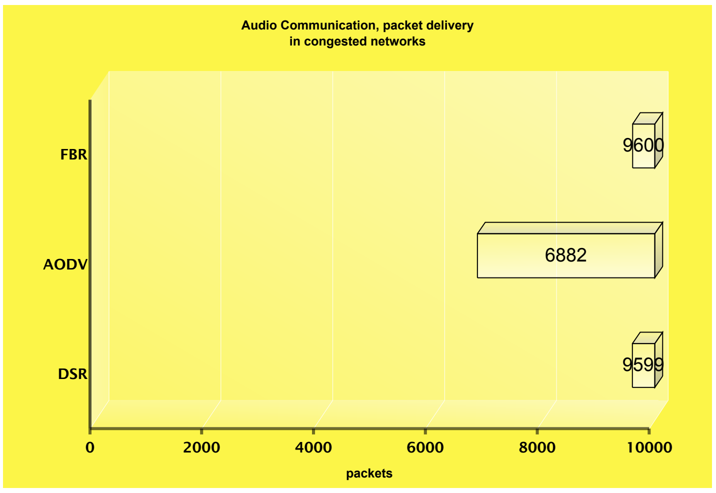

Figure 5.4: **Webradio, congested network**

Protocol	DSR	AODV	FBR
Stable avg (spread σ)	50 (0)	49 (0)	50 (9)
Volatile avg (spread σ)	22.6 (3.6)	43.6 (2.62)	19 (3.26)
Congested avg (spread σ)	18.2 (3.81)	36.1 (3.02)	6.9 (2.33)

Figure 5.5: **Summary Phone Call simulation**

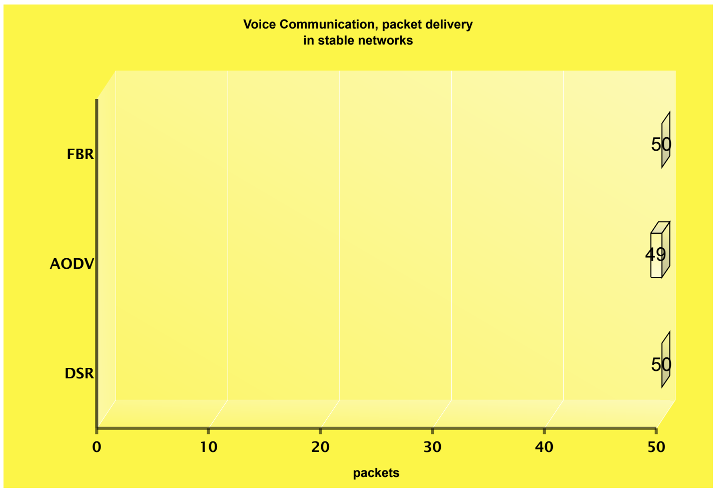

Figure 5.6: **Phone call, stable network**

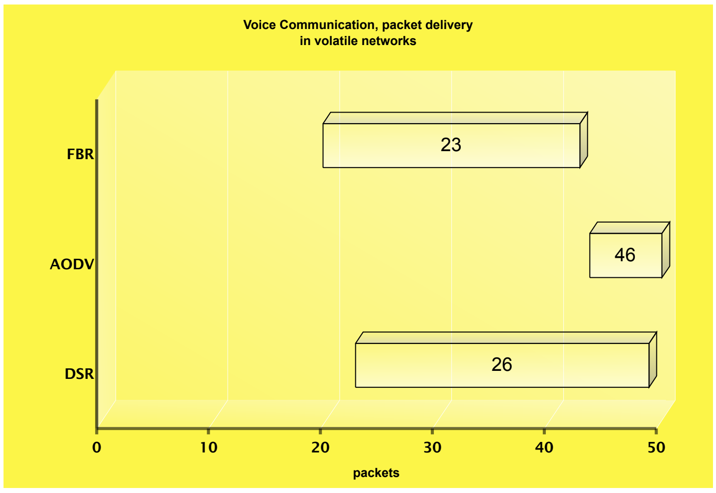

Figure 5.7: **Phone call, volatile network**

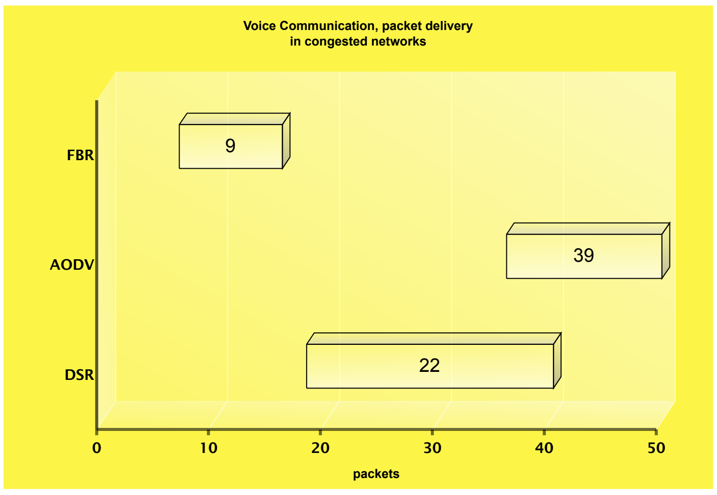

Figure 5.8: **Phone call, congested network**

Chapter 6

Discussion

In this chapter we take a look on our findings and compare our theoretical expectations with the results of cursory simulations. Furthermore, we reflect about the method and strategy used during this thesis and outline further possible work on this topic.

First of all, let's recapitulate our approach: by creating network and communication models, we analyzed the real-time relevant criteria of *creation* and *repair time*, *network* and *packet overhead* (chapter 2.3). By doing so, we created a multi-dimensional space, and in order to ease the analysis of it, we concentrated our thought on particular scenarios and made assumptions about parameters. The scenarios are unidirectional audio stream on one side and bidirectional voice streams on the other. Important parameters like the number of participating nodes, the degree of connectivity or mobility were stipulated in different models (chapter 2.4).

Although always keeping a certain degree of abstraction, we calculated through all these scenarios and established rankings for the inspected protocols. As we have discovered, FBR and DSDV perform better in terms of creation and repair time, thanks to their proactive nature. In terms of overhead, DSDV is the most gentle for the network by far before DSR. FBR is the most efficient with regard to packet overhead as it can exploit the fact that unidirectional communication does not need full metadata as DSDV and AODV do.
Summing up, Field Based Routing seems to perform best in real-time relevant applications at the expense of a certain network overhead.

If we look now at the outcome of our simulations, the situation is turned upside down. AODV demonstrates by far a more robust packet delivery rate than FBR or DSR. However, we need to consider that in the *Glomosim* distribution, DSR and AODV are standard protocols while FBR has been autonomously implemented at TIK [15]. It's admissible to say that its immature implementation could prevent the FBR protocol from showing its true potential. And one of its strengths, the service oriented architecture, is not exploited in these simulations as they use only one single gateway node. Also, these simulations do not test our three real-time criteria *overhead*, *creation-* and *repair time*. More elaborate simulations centered to that purpose would be needed before we can have empirical data that is useful for making statements about real-time adequacy of these protocols. Due to lack of time we could not include such thorough simulations in this thesis. However, the simulations done give us a rudimentary overview of the performance.

In general, we have to stress out at this point that the chosen top-down approach of this thesis is very complex and extremely tedious. We tried to handle complexity by reducing the dimension of free parameters to the point we could reason with scalar values. However, the choices made and the parameters assumed are highly debatable. Also, the modeling of certain aspects like *cache*, *mobility* or *connectivity* at the same time makes its assessment very difficult. Thus we have to state that our model is not as robust as we wish it to be.

Hence, if the approach of this thesis would be continued in following work, more particular analytical attention would be necessary to every single aspect of the network and communication behavior modeled. The objective to gain more general statements by applying the same analytical scheme on these different protocols dearly has to face the problem of describing those protocols in their behavior for path creation and repair, and state maintenance as well.
This modeling effort highly depends on the actual implementation of the protocol. A lot of important parameters are not always fixed and are an implementation choice that is made with respect to separate optimizations. Indeed, one of the most incisive parameter is the update frequency of routing information in proactive protocol. By changing it, the performance could vary a lot. However, we have tried to cope with that fact by using different network models: *stable*, *volatile* and *congested*.

Summing up, the Field-Based Routing and Destination-Sequenced Distance-Vector protocol seem promising for real-time application and deserve more investigation. Although our simulations do not support this statement, more thorough simulations should give clearer evidence of it.

Part II

Appendix

Chapter 7

Transmission cost calculations

7.1 DSR

scenario	*cost*	*transmission overhead*
$cost_{AC_{1_{stable}}}$	1610	1.006071202
$cost_{AC_{1_{volatile}}}$	3029	1.892702539
$cost_{AC_{1_{congested}}}$	1665	1.040254552
$cost_{AC_{2_{stable}}}$	57801	1.003474989
$cost_{AC_{2_{volatile}}}$	108544	1.884441043
$cost_{AC_{2_{congested}}}$	59581	1.034379386
$cost_{VC_{1_{stable}}}$	101	1.047907316
$cost_{VC_{1_{volatile}}}$	195	2.025830646
$cost_{VC_{1_{congested}}}$	109	1.134928656
$cost_{VC_{2_{stable}}}$	294	1.018236313
$cost_{VC_{2_{volatile}}}$	557	1.931413549
$cost_{VC_{2_{congested}}}$	308	1.067783901
$cost_{VC_{3_{stable}}}$	1161	1.007109687
$cost_{VC_{3_{volatile}}}$	2185	1.896007138
$cost_{VC_{3_{congested}}}$	1202	1.042604618

7.2 DSDV

scenario	*cost*	*transmission overhead*
$cost_{AC_{1_{stable}}}$	1627	1.016835129
$cost_{AC_{1_{volatile}}}$	1795	1.121411552
$cost_{AC_{1_{congested}}}$	1654	1.033596535
$cost_{AC_{2_{stable}}}$	57735	1.01449563
$cost_{AC_{2_{volatile}}}$	63759	1.119072052
$cost_{AC_{2_{congested}}}$	59407	1.031355029
$cost_{VC_{1_{stable}}}$	109	1.125367824
$cost_{VC_{1_{volatile}}}$	119	1.229944247
$cost_{VC_{1_{congested}}}$	110	1.140550147
$cost_{VC_{2_{stable}}}$	317	1.098630688
$cost_{VC_{2_{volatile}}}$	347	1.203207111
$cost_{VC_{2_{congested}}}$	322	1.114932928
$cost_{VC_{3_{stable}}}$	1255	1.088604262
$cost_{VC_{3_{volatile}}}$	1375	1.193180685
$cost_{VC_{3_{congested}}}$	1274	1.105326471

7.3 AODV

scenario	*cost*	*transmission overhead*
$cost_{AC_{1_{stable}}}$	1907	1.191835129
$cost_{AC_{1_{volatile}}}$	2075	1.296411552
$cost_{AC_{1_{congested}}}$	1934	1.208596535
$cost_{AC_{2_{stable}}}$	68515	1.18949563
$cost_{AC_{2_{volatile}}}$	74539	1.294072052
$cost_{AC_{2_{congested}}}$	69487	1.206355029
$cost_{VC_{1_{stable}}}$	221	2.292034491
$cost_{VC_{1_{volatile}}}$	231	2.396610914
$cost_{VC_{1_{congested}}}$	222	2.307216813
$cost_{VC_{2_{stable}}}$	653	2.265297355
$cost_{VC_{2_{volatile}}}$	683	2.369873778
$cost_{VC_{2_{congested}}}$	658	2.281599595
$cost_{VC_{3_{stable}}}$	2599	2.255270929
$cost_{VC_{3_{volatile}}}$	2719	2.359847351
$cost_{VC_{3_{congested}}}$	2618	2.271993138

7.4 FBR

scenario	*cost*	*transmission overhead*
$cost_{AC_{1_{stable}}}$	1664	1.039906342
$cost_{AC_{1_{volatile}}}$	1664	1.039906342
$cost_{AC_{1_{congested}}}$	1664	1.03980555
$cost_{AC_{2_{stable}}}$	59764	1.037566843
$cost_{AC_{2_{volatile}}}$	59764	1.037566843
$cost_{AC_{2_{congested}}}$	59764	1.037564043
$cost_{VC_{1_{stable}}}$	124	1.290105704
$cost_{VC_{1_{volatile}}}$	124	1.290105704
$cost_{VC_{1_{congested}}}$	124	1.288425828
$cost_{VC_{2_{stable}}}$	364	1.263368568
$cost_{VC_{2_{volatile}}}$	364	1.263368568
$cost_{VC_{2_{congested}}}$	364	1.262808609
$cost_{VC_{3_{stable}}}$	1444	1.253342142
$cost_{VC_{3_{volatile}}}$	1444	1.253342142
$cost_{VC_{3_{congested}}}$	1444	1.253202152

Chapter 8

Glomosim

Here only the most essential parameters are indicated. The full configuration files and scripts are provided on a separate CD-ROM.

8.1 Configuration parameters used

Followingly, the parameters mapping the network topologies into the *Glomosim* framework are shown for *stable*, *volatile* and *congested* network models.

For all topologies, NUMBER-OF-NODES is set at 50.

The 20 random seeds used are: `234, 34, 9845, 485, 8347, 3, 123, 2345, 368, 99234, 367, 251, 997, 1973, 57, 38, 7, 29, 16, 49`

Stable network

TERRAIN-DIMENSIONS	(2000 , 2000)
MOBILITY	NONE

Volatile network

TERRAIN-DIMENSIONS	(2000 , 2000)
MOBILITY	RANDOM-WAYPOINT
MOBILITY-WP-MAX-SPEED	10

Congested network

TERRAIN-DIMENSIONS	(100 , 100)
MOBILITY	RANDOM-WAYPOINT
MOBILITY-WP-MAX-SPEED	5

Additionally, a lot of noise traffic is introduced from other nodes, causing collisions and bandwidth bottlenecks.

8.2 Communication model

8.2.1 Simulation AC1: Unidirectional Audio Broadcast

This scenario represents 5 minutes of webradio transmission from the internet. The client nodes periodically sends a *keep alive* packet to the source.
The required bandwidth is 64KBit/s. Hence, 5 minutes of transmission amount to 19'200'000 Bit. At a packet size of 2000 Bit, the number of packet being transmittet is 9600, sent in interval of 1/32 s.
`Configuration file: PROJECT/bin/ac1.{dsr,aodv,fbr}.[congested].in`

8.2.2 Simulation VC2: Bidirectional Audio Transmission

This scenario represents 3 minutes of voice transmission from the MANET to a peer node in the internet, through a gateway.
The required bandwidth is 9.6KBit/s. Hence, 3 minutes of transmission amount to 1'728'000 Bit. At a packet size of 2000 Bit, the number of packet being transmittet is 864 in each direction, sent in interval of 4.8 s.
`Configuration file: PROJECT/bin/vc2.{dsr,aodv,fbr}.[congested].in`

Chapter 9

Additional inquired protocols

9.1 Optimized Link State Routing

The protocol [8] is an optimization of the classical link state algorithm tailored to the requirements of a mobile wireless LAN. The key concept used in the protocol is that of multipoint relays (MPRs). MPRs are selected nodes which forward broadcast messages during the flooding process. This technique substantially reduces the message overhead as compared to a classical flooding mechanism, where every node retransmits each message when it receives the first copy of the message. In OLSR, link state information is generated only by nodes elected as MPRs. Thus, a second optimization is achieved by minimizing the number of control messages flooded in the network. As a third optimization, an MPR node may chose to report only links between itself and its MPR selectors. Hence, as contrary to the classic link state algorithm, partial link state information is distributed in the network. This information is then used for route calculation. OLSR provides optimal routes (in terms of number of hops). The protocol is supposedly particularly suitable for large and dense networks as the technique of MPRs works well in this context.

9.2 Topology Broadcast based on Reverse-Path Forwarding

Topology Dissemination Based on Reverse-Path Forwarding [9] is a proactive, link-state routing protocol designed for mobile ad-hoc networks, which provides hop-by-hop routing along shortest paths to each destination. Each node running TBRPF computes a source tree (providing paths to all reachable nodes) based on partial topology information stored in its topology table, using a modification of Dijkstra's algorithm. To minimize overhead, each node reports only *part* of its source tree to neighbors. TBRPF uses a combination of periodic and differential updates to keep all neighbors informed of the reported part of its source tree. Each node also has the option to report additional topology information (up to the full topology), to provide improved robustness in highly mobile networks. TBRPF performs neighbor discovery using ”differential” HELLO messages which report only *changes* in the status of neighbors. This results in HELLO messages that are much smaller than those of other link-state routing protocols such as OSPF.

9.3 Temporally-Ordered Routing Algorithm

TORA is a distributed routing protocol [5] based on a link reversal algorithm. It is designed to discover routes on demand, provide multiple routes to a destination, establish routes quickly, and minimize communication overhead by localizing algorithmic reaction to topological changes when possible. Route optimality (shortest-path routing) is considered of secondary importance, and longer routes are often used to avoid the overhead of discovering newer routes.

The actions taken by TORA can be described in terms of water flowing downhill towards a destination node through a network of tubes that models the routing state of the real network. The tubes represent links between nodes in the network, the junctions of tubes represent the nodes, and the water in the tubes represents the packets flowing towards the destination. Each node has a height with respect to the destination that is computed by the routing protocol. If a tube between nodes A and B becomes blocked such that water can no longer flow through it, the height of A is set to a height greater than that of any of its remaining neighbors, such that water will now flow back out of A (and towards the other nodes that had been routing packets to the destination via A).

At each node in the network, a logically separate copy of TORA is run for each destination. When a node needs a route to a particular destination, it broadcasts a QUERY packet containing the address of the destination for which it requires a route. This packet propagates through the network until it reaches either the destination, or an intermediate node having a route to the destination. The recipient of the QUERY then broadcasts an UPDATE packet listing its height with respect to the destination. As this packet propagates through the network, each node that receives the UPDATE sets its height to a value greater than the height of the neighbor from which the UPDATE was received. This has the effect of creating a series of directed links from the original sender of the QUERY to the node that initially generated the UPDATE.

When a node discovers that a route to a destination is no longer valid, it adjusts its height so that it is a local maximum with respect to its neighbors and transmits an UPDATE packet. If the node has no neighbors of finite height with respect to this destination, then the node instead attempts to discover a new route as described above. When a node detects a network partition, it generates a CLEAR packet that resets routing state and removes invalid routes from the network. TORA is layered on top of IMEP, the Internet MANET Encapsulation Protocol [5], which is required to provide reliable, in-order delivery of all routing control messages from a node to each of its neighbors, plus notification to the routing protocol whenever a link to one of its neighbors is created or broken. To reduce overhead, IMEP attempts to aggregate many TORA and IMEP control messages (which IMEP refers to as objects) together into a single packet (as an object block) before transmission. Each block carries a sequence number and a response list of other nodes from which an ACK has not yet been received, and only those nodes ACK the block when receiving it; IMEP retransmits each block with some period, and continues to retransmit it if needed for some maximum total period, after which time, the link to each unacknowledged node is declared down and TORA is notified. IMEP can also provide network layer address resolution, but we did not use this service, as we used ARP [19] with all four routing protocols. For link status sensing and maintaining a list of a nodes neighbors, each IMEP node periodically transmits a BEACON (or BEACON-equivalent) packet, which is answered by each node hearing it with a HELLO (or HELLO-equivalent) packet.

Bibliography

[1] Vincent Lenders, Martin May, Bernhard Plattner, *Towards a New Communication Paradigm for Mobile Ad Hoc Networks*, IEEE International Workshop on Heterogeneous Multi-Hop Wireless and Mobile Networks, Washington DC, November, 2005.

[2] V. Lenders, M. May, and B. Plattner, *Service Discovery in Mobile Ad Hoc Networks: A Field Theoretic Approach*, in Proceedings of the IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM), Taormina, Italy, June 2005.

[3] David B. Johnson, David A. Maltz, Yih-Chun Hu, *The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks (DSR)*, IETF MANET Working Group, INTERNET-DRAFT, 19 July 2004, `http://www.ietf.org/internet-drafts/draft-ietf-manet-dsr-10.txt`

[4] Charles E. Perkins and Pravin Bhagwat. *Highly dynamic Destination-Sequenced Distance-Vector routing (DSDV) for mobile computers.* In Proceedings of the SIGCOMM 94 Conference on Communications Architectures, Protocols and Applications, pages 234244, August 1994. A revised version of the paper is available from `http://www.cs.umd.edu/projects/mcml/papers/Sigcomm94.ps`

[5] Vincent D. Park and M. Scott Corson. *Temporally-Ordered Routing Algorithm (TORA) version 1: Functional specification*
Internet-Draft, draft-ietf-manet-tora-spec-00.txt, November 1997.

[6] Charles Perkins. *Ad Hoc On Demand Distance Vector (AODV) routing*
Internet-Draft, draft-ietf-manet-aodv-00.txt, November

[7] Charles E. Perkins and Elizabeth M. Royer. *Ad hoc On-Demand Distance Vector Routing*
Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications, New Orleans, LA, February 1999, pp. 90-100.

[8] T. Clausen, P. Jacquet, *Optimized Link State Routing Protocol* Internet Draft rfc 3626,
http://hipercom.inria.fr/olsr/rfc3626.txt October 2003

[9] R. Ogier, F. Templin, M. Lewis, *Topology Dissemination Based on Reverse-Path Forwarding*, http://tools.ietf.org/html/rfc3684, February 2004

[10] jitter01 Yishay Mansour, Boaz Patt-Shamir, *Jitter Control in QoS Networks* - IEEE/ACM Transactions on Networking, 2001
urlhttp://ieeexplore.ieee.org/iel4/5965/15971/00743428.pdf

[11] Fan Chung, Linyuan Lu, *The average distances in random graphs with given expected degrees* - PNAS 99 (2002) 15879-15882
urlmath.ucsd.edu/ fan/wp/avef.pdf

[12] Josh Broch, David A. Maltz, David B. Johnson, Yih-Chun Hu, Jorjeta Jetcheva *A Performance Comparison of Multi-Hop Wireless Ad Hoc Network Routing Protocols*
Computer Science Department, Carnegie Mellon University, http://www.monarch.cs.cmu.edu/,1998

[13] X. Zeng, R. Bagrodia, M. Gerla, *Glomosim: A library for parallel simulation of large-scale wireless networks*, in: Workshop on Parallel and Distributed Simulation, 1998, pp. 154161.

[14] LaTeX, written by L. B. Lamport,
http://en.wikipedia.org/wiki/LaTeX

[15] Computer Engineering and Networks Laboratory (TIK), Swiss Federal Institute of Technology (ETH) Zurich, Switzerland