Georgiy Senenkov

PROTOCOLS COMPARISON IN AD HOC NETWORKS

Master's thesis

Mobile computing

17.9.2003

University of Jyväskylä

Department of Mathematical Information Technology

Author: Georgiy Senenkov

Contact Information: E-mail: gsenen@cc.jyu.fi

Title: Protocols Comparison in Ad Hoc Networks

Työn nimi: Protokollien Vertailu Ad Hoc Verkoissa

Work: Master's Thesis

Number of Pages: 76

Study Line: Mobile Computing

Department: University of Jyväskylä, Department of Mathematical Information Technology

Keywords: Ad hoc networks, routing protocols, performance evaluation, simulation

Avainsanat: Ad hoc verkot, reititysprotokolla, tehokkuuden arviointi, simulointi.

Abstract: Ad hoc networks are networks, which are characterized by multi-hop wireless connectivity, unpredictably and frequently changing network topology. They demand efficient routing protocols. The purpose of this work is the performance comparison of two on-demand and one table-driven routing protocols for ad hoc networks – Ad Hoc On-Demand Distance Vector Routing (AODV), Dynamic Source Routing (DSR) and Destination-Sequenced Distance Vector (DSDV) respectively.

Tiivistelmä: Ad hoc verkoille on ominaista se, että yhteys muodostetaan usean hypyn kautta ja verkon topologia voi muuttua jatkuvasti. Verkot vaativat tehokkaita reititysprotokollia. Tämän työn tarkoituksena on verrata kahden ”on-demand” ja yhden ”table-driven” protokollan tehokkuutta keskenään. Verrattavat protokollat ovat “Ad Hoc On-Demand Distance Vector Routing” (AODV), “Dynamic Source Routing” (DSR) ja “Destination-Sequenced Distance Vector” (DSDV).

Preface

This Master's Thesis work was done at the Department of Mathematical Information Technology, and Agora Center, University of Jyväskylä, Finland. I would like to thank my supervisors Erkki Laitinen and Matthieu Weber for their help, support, comments and advices during my master thesis.

Finally I would like to thank Vagan Terziyan and Helen Kaykova for their help and support during my studying and living in Finland.

University of Jyväskylä, 25.11.2002

Abbreviations

ACK	Acknowledgment
ACTD	Advanced Concept Technology Demonstration
ANP	Accelerated Null message Protocol
AODV	Ad Hoc On-Demand Distance Vector Routing
API	Application Programming Interface
ARP	Address Resolution Protocol
ATM	Asynchronous Transfer Mode
AWE	Advanced Warfighting Experiment
BER	Bit Error Rate
CBR	Continuous Bit-Rate
CSMA	Carrier Sense Multiple Access
DARPA	Defense Advanced Research Projects Agency
DBF	Distributed Bellman-Ford
DoD	Department of Defense
DS	Direct Sequence
DS	Direct-Sequence
DSDV	Destination-Sequenced Distance Vector
DSR	Dynamic Source Routing

DSSS	Direct Sequence Spread Spectrum
ELB	Extending the Littoral Battlespace
FAMA	Fixed-Assigned Multiple Access
FH	Frequency Hopping
FIFO	“First-In, First-Out”
FSN	Finite State Machine
FTP	File Transfer Protocol
GloMo	Global Mobile Information Systems
GloMoSim	Global Mobile information system Simulator
HTTP	Hypertext Transfer Protocol
IETF	Internet Engineering Task Force
IP	Internet Protocol
JTA	Joint Technical Architecture
LANs	Local Area Networks
LAR	Location-Aided Routing
LOS	Line of Sight
LPI/D	Low Probability of Interception/ Detection
MAC	Medium Access Control
MACA	Medium Access Collision Avoidance
MACAW	Media Access Protocol for Wireless LAN's

MANET	Mobile Ad Hoc Networking Working Group
MMWN	Multimedia Mobile Wireless Network
ModSAF	Modular Semi-Automated Forces
NPDU	Network Protocol Data Unit
NTDR	Near-Term Digital Radio
ODMRP	On-Demand Multicast Routing Protocol
OPNET	OPtimized Network Engineering Tool
OSPF	Open Shortest Path First
OTH	Over-The-Horizon
PARSEC	PARallel Simulation Environment for Complex systems
RERR	Route Error Packet
RIP	Routing Information Protocol
RIP	Routing Information Protocol
RREP	Route Reply Packet
RREQ	Route Request Packets
RTP	Real Time Protocol
RTS/CTS	Request-to-Send/Clear-to-Send

SIRCIM	Simulation of Indoor Radio Channel Impulse response Models
SURAN	Survivable Radio Network
TCP	Transmission Control Protocol
TDMA	Time-Division Multiple Access
TI	Tactical Internet
TIREM	Terrain Integrated Rough Earth Model
UDP	User Datagram Protocol
WG	Working Group
WINGs	Wireless Internet Gateways
WRP	Wireless Routing Protocol

Contents

1 INTRODUCTION 1
1.1 MOTIVATION 2
1.2 THE PRESENT 4
1.2.1 Tactical Internet 5
1.2.2 ELB 6
1.2.3 G1oMo 7
1.2.4 IETF MANET Working Group 9
1.3 THESIS LAYOUT 9
2 ROUTING PROTOCOLS 11
2.1 DSDV PROTOCOL 12
2.1.1 Route Advertisement 12
2.1.2 Topology Changes 13
2.1.3 Route Selection Criteria 14
2.1.4 Routing Information 16
2.2 AODV PROTOCOL 18
2.2.1 Route Establishment 18
2.2.2 Expanding Ring Search 20
2.2.3 Route Reply 20
2.2.4 Route Error Messages 21
2.2.5 Local Repair 22
2.2.6 Route Table Management 23
2.3 DSR PROTOCOL 24
2.3.1 Basic DSR Route Discovery 26
2.3.2 Additional Route Discovery Features 28
2.3.3 Basic DSR Route Maintenance 30
2.3.4 Additional Route Maintenance Features 32
2.3.5 Routing Information 33
CONCLUSION 35
3 SIMULATORS 36
3.1 GLOMOSIM 36
3.2 NS 40
3.3 QUALNET 43

3.4 OPNET 45
CONCLUSION 46
4 SIMULATION RESULTS 48
4.1 PHYSICAL AND DATA LINK LAYER MODEL 48
4.2 MEDIUM ACCESS CONTROL 50
4.3 ADDRESSES AND PACKET BUFFERING 51
4.4 PERFORMANCE METRICS 51
4.5 "MOTORWAY" MODEL 52
4.5.1 Traffic and Mobility Models 52
4.5.2 Simulations 54
4.6 "AIRPORT" MODEL 60
4.6.1 Traffic and Mobility Models 60
4.6.2 Simulations 61
CONCLUSION 62
5 CONCLUSION 63
REFERENCES 64

1 INTRODUCTION

In recent years, mobile computing has enjoyed a huge increase in popularity [1]. The continued miniaturization of mobile computing devices and the exponential growth of processing power which is available in mobile computers involve more and better computer-based applications. At the same time, the markets for wireless telephones and communication devices are experiencing rapid growth. It is well known that nowadays there are more than a billion wireless communication devices in use, and more than 200 million wireless telephone handsets have been purchased annually. The rise of wireless telephony will change what it means to be “in touch”. Already today many people use their office telephone for receiving messages while they are away and rely on their mobile telephone for more important or timely messages. Similar transformation waits for mobile computer users, and it can be expected that new applications will be constructed for equally mundane but immediately convenient uses.

Nowadays technology production seems like a great expansion to support mobile computing. Also many new applications are being developed and wireless data communication products are becoming available that have improved much over the past years. The laptop computers can use bandwidth over radio and infrared links are easily used 10 to 100 times more than it was available just ten years ago.

As a wide class of customers uses Internet applications and as wireless network nodes proliferate, customers will expect to use networking applications even in situations where the Internet itself is not available. For example, people using laptop computers at a conference in a hotel might wish to communicate in a variety of ways, without the mediation of routing across the global Internet. Yet today such obvious communication requirements cannot be easily met using Internet protocols [2]. The mobile computer users form a possibly short-lived network just for the current communication needs in other words, an ad hoc network.

In an ad hoc network, mobile nodes communicate with each other using multi-hop wireless links [11]. There is no stationary infrastructure such as base station network. Each node in the network also acts as a router, forwarding data packets for other nodes. A central chal-

lenge in the design of ad hoc networks is the development of dynamic routing protocols that can efficiently find routes between two communicating nodes. The routing protocol must be able to keep up with the significant topology changes over time due to user mobility [12]. Recently there has been a renewed interest in this field due to the common availability of low-cost laptops and palmtops with radio interfaces [3]. Interest is also partly maintained by growing enthusiasm in running common network protocols in dynamic wireless networks.

A mobile ad hoc networking (MANET) working group has also been formed within the Internet Engineering Task Force (IETF) to develop a routing framework for IP-based protocols in ad hoc networks [3].

1.1 Motivation

Ad hoc networks are concerned with ways that mobile devices can perform like routers for Internet infrastructure. Keeping track of the connections between computers is something without which a computer network cannot work.

An ad hoc network could exist without any existing stationary infrastructure. For example, one could turn on 15 laptop computers, each with the same kind of infrared data communications adapter, and hope that they could form a network among themselves [1]. This feature is also useful even if the laptops were stationary.

In ad hoc networks, most of the discussion focuses on the following aspects:

- The nodes are using Internet Protocol, and they have IP addresses.
- The nodes are far away from each other, thus they are not within range of each other.
- The nodes may be mobile hence that two nodes within range at one point in current time, may be out of range later.

- The nodes are able to assist each other in the process of delivering packets of data.

An example of a simple ad hoc network, which consists of 3 mobile hosts, can be seen in Figure 1.1. It illustrates that mobile host C is not within the range of host A (the range of each node is indicated by a circle around the node), and the node A is also not in range of node C wireless transmitter, but can communicate via node B.

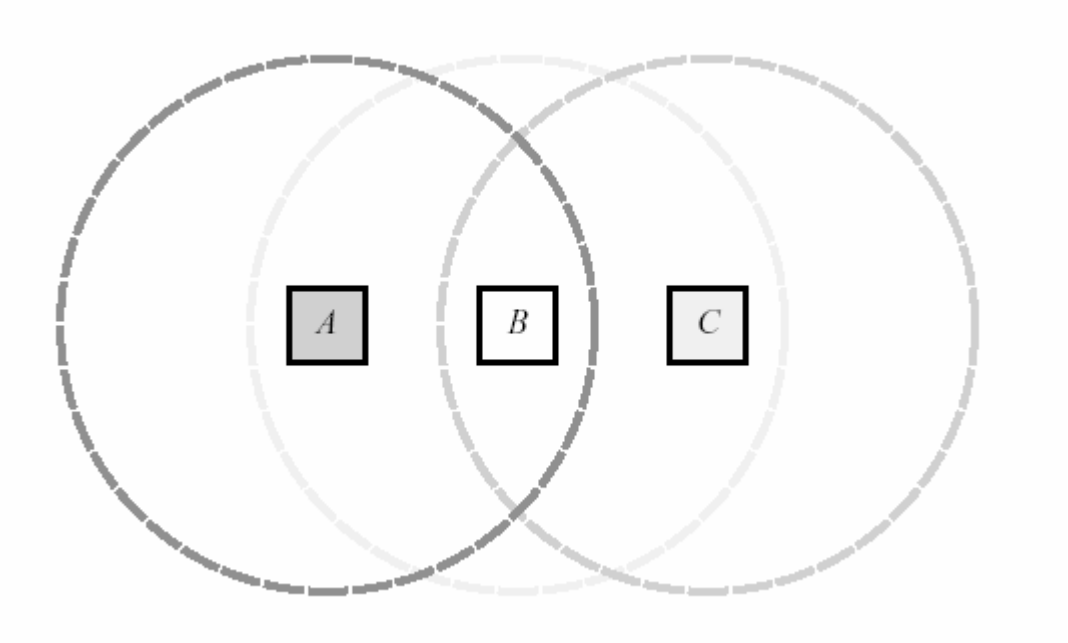


Figure 1.1 –Example of simple ad hoc network.

One of the purposes for ad hoc network implementation is found in the military need for battlefield survivability. To survive on a battlefield, the military personnel and their mobile platforms must be able to move freely without any restrictions, which wired devices could impose. Thus, these needs require avoiding single points of failure such as centralized base stations in cellular networks.

Additionally, the military cannot rely on access to a fixed, preplaced communications infrastructure in battlefield environments, because in some regions, such as the desert or the jungle, there is no terrestrial communications infrastructure. Also it is possible that access is unavailable or there is a very high damage probability to the local communications infra-

structure. A rapidly deployable, self-organizing mobile infrastructure is the major factor that distinguishes ad hoc network design issues from those associated to cellular systems.

Also one more motivating factor for ad hoc network implementation comes from the physics of electromagnetic propagation, because frequencies much higher than 100 MHz cannot propagate beyond line of sight (LOS). Terrain, foliage, and man-made obstacles can also prevent LOS connectivity. Consequently, multihop (store-and-forward) packet routing must be used between users who are not within LOS of each other.

Finally, mobile wireless, distributed, multihop networking developed out of the military need for survivability, operation without preplaced infrastructure, and connectivity beyond LOS.

1.2 The Present

The growth of the Internet infrastructure and the microcomputer revolution has made the initial packet radio network ideas both more applicable and feasible [2]. Packet switching techniques, such as asynchronous transfer mode (ATM) and the Internet Protocol (IP) are widely accepted. Therefore, a MANET must be able to interact with the dominant protocols of the existing infrastructure, whether or not it will actually use them or their variants [1].

Nowadays, the memory storage of the microcomputers, the computational capability and signal processing has been significantly increased. However, wireless communications devices have much lower data rates than fiber optic cable, but on the other hand, handheld cellular phones are much smaller than a decade ago [13]. It is a safe assumption for the predictable future that the data rate and bit error rate for wireless communications will be several orders of magnitude behind those for the state-of-the-art wired media, especially if the wireless user is moving.

The use of spread spectrum is no longer confined to military users who operate in low probability of interception/ detection (LPI/D) modes. The use of some form of spread spectrum is now considered a wise decision for multipath and co-channel interference mitigation [1]. In the commercial cellular standard IS-95, the motivation for using the direct-sequence (DS) form of spread spectrum is to increase the capacity of the cellular system. It allows dynamic sharing of the channel. Both DS and frequency hopping (FH) modes of operation are contained in the IEEE 802.11 standard for wireless local area networks (LANs) to meet power spectrum density requirements.

User expectations have risen with the increase in number of wireless technology. Users will soon expect the same connectivity available in handheld devices as in an office environment, through either the cellular infrastructure or emerging commercial mobile communications satellites.

In the midst of all these changes, the laws of physics and chemistry are fundamental: frequencies above roughly 100 MHz will rarely propagate beyond LOS, and the speed of light will not exceed roughly 3×10^8 m/s. Moreover, the energy storage capabilities of batteries have increased only gradually over the last few decades.

The issue of spectrum availability has both positive and negative aspects. It is difficult for commercial or military users to gain access to significant amounts of contiguous bandwidth. The inability to occupy a contiguous portion of the spectrum that is sufficiently wide reduces some of the advantages of using direct sequence waveforms to mollify multipath [1].

1.2.1 Tactical Internet

Embodying many of the basic attributes of a MANET, the U.S. Army's March 1997 TF XXI AWE [1] may be the largest-scale implementation (comprising thousands of nodes) of a mobile, wireless, multihop packet radio network. In the tactical Internet (TI), nodes consist of both vehicular and man-packed radios. They were already in the government inventory primarily Enhanced Position Location Reporting Systems (EPLRSs) and Single

Channel Ground Airborne Radio Systems, running modified commercial Internet protocols. Originally they were designed for robust geolocation but they are now used for data connectivity. EPLRS is a direct-sequence spread-spectrum, time-division multiple access (TDMA) radio capable of transmitting data at tens of kilobits per second. It was decided to use commercial protocols as a basis in the TI and in all future DoD communications systems.

The TI was a success and allowed the army to demonstrate important doctrinal and operational concepts [1]. However, existing commercial protocols specifically developed for the fixed, wired infrastructure must be appropriately modified before they are used in the mobile, wireless environment. The Joint Technical Architecture (JTA) has contributed greatly to the improved interoperability of the vast majority of the DoD communications infrastructure.

1.2.2 ELB

The purpose of the April 1999 Extending the Littoral Battlespace Advanced Concept Technology Demonstration was to demonstrate the feasibility of Marine Corps warfighting concepts [1]. They require over-the-horizon (OTH) communications from ships at sea to Marines on land via an aerial relay. The physical layer for the MANET of the ELB ACTD was a commercial wireless local area network (WLAN) product, Lucent's Wave LAN and VRC-99A, a direct-sequence spread-spectrum radio descended from the DARPA LPR [15]. Wave LAN was used to connect to an access point on a terrestrial or airborne relay. VRC-99A was used as the mobile backbone to connect the terrestrial or airborne routers. While the number of nodes in the network is approximately 20, the ELB ACTD was successful in demonstrating the use of aerial relays for connecting users beyond LOS.

The ELB ACTD set very aggressive cost and performance goals for MANET technology [14]. Wave LAN and the VRC-99A were selected primarily to reduce system cost by utilizing existing commercial or military equipment, respectively. However, Wave LAN required several modifications, such as the addition of an external power amplifier and the

increase in certain time-out settings, to operate at the extended ranges involved. The sum of these modifications reduced much of the advantage of using existing commercial equipment. Additionally, the ability of nodes to roam seamlessly throughout the network was limited [1]. The second phase of the ELB ACTD is targeting these areas for improvement.

1.2.3 G1oMo

The widespread implementation of Internet and web technologies provided incentive to both commercial and defence sectors to extend the global information infrastructure into the mobile wireless environment. One DoD response to these changes was the initiation of the DARPA GloMo in 1994, [1] which has just recently concluded.

The goal of the GloMo program [1] was “to make the mobile, wireless environment a first-class citizen in the defence information infrastructure by providing user friendly connectivity and access to services for mobile users.” It aimed to provide office-environment, Ethernet-type multimedia (voice, video, images, etc.) connectivity any time, anywhere, in handheld devices. The G1oMo program had the following five thrusts:

1. Infrastructure design, such as computer-aided design tools.
2. Nodes that provide low-cost, low-power wireless access with sufficient processing power.
3. Network protocols and algorithms with robust architectures that can be rapidly deployed.
4. End-to-end networking in heterogeneous environments
5. Mobile applications that adapt to varying network connectivity and quality of service (QoS).

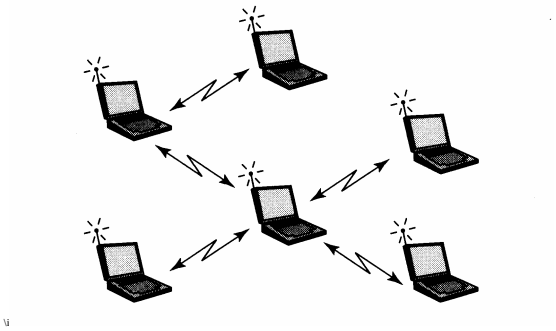


Figure 1.2 - Flat Network Architecture

GloMo followed a number of networking approaches for MANET, including the Wireless Internet Gateways (WINGs) and the Multimedia Mobile Wireless Network (MMWN). The goal of WINGS was to design and demonstrate seamless operation between a MANET and the Internet. It had to be done without treating the MANET as an opaque sub-network that uses an intranet routing protocol below IP for packet forwarding [16]. WINGs uses a flat (peer-to-peer) network architecture (Figure 1.2). Several prototype versions were demonstrated within the GloMo program.

MMWN is based on a hierarchical network architecture (Figure 1.3) that has its roots in the Survivable Radio Network (SURAN) program [1]. It uses a modular system of link and network layer algorithms to support distributed, real-time multimedia applications in a MANET.

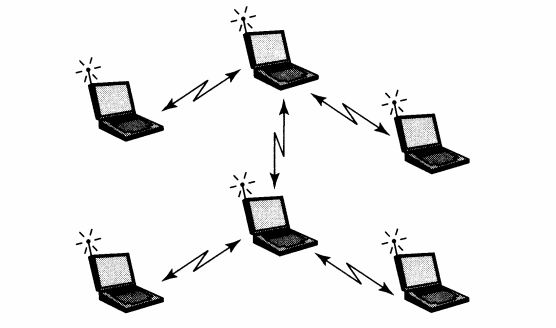


Figure 1.3 - Hierarchical Network Architecture

MMWN has three components: clustering techniques; location management; and virtual circuit setup and repair. It is currently being demonstrated as part of the GloMo program.

Philosophically similar techniques are being applied in the Near-Term Digital Radio (NTDR) program.

1.2.4 IETF MANET Working Group

One focal point for recent interest in MANET is the Internet Engineering Task Force MANET Working Group (IETF MANET WG) [17]. The activities of an IETF WG may not appear to be relevant to the military, but the DoD mandate to use open standards has made it critical to participate in standards groups such as the IETF. Many of the commercial protocols developed for the fixed, wired infrastructure rarely translate well to the mobile, wireless regime. Future DoD communications systems should comply with the JTA. The best way to ensure the existence of network protocols that operate effectively in military environments is to be involved in the development of standards for commercial protocols.

While the original motivation for MANET was military needs, its non-military applications have grown substantially since the middle of 1980s [1]. Police, fire and rescue, disaster relief, robotics, space, distributed sensors, and impromptu team communications are a few possible applications of MANET technology.

The short term goal of the MANET WG is to “standardize an interdomain unicast routing protocol, which provides one or more modes of operation, each mode specialized for efficient operation in a given mobile networking “context”, where a context is a predefined set of network characteristics” [1].

1.3 Thesis Layout

This Master’s Thesis is organized as follows. Section 2 highlights the two on-demand and one table-driven routing protocols for ad hoc wireless networks. Its purpose is to describe the routing techniques and principles of these protocols. Section 3 introduces a set of simulators for wireless networks. It also represents a set of resources of each simulator. Section

4 shows the simulation result for three protocols, which are simulated for “Motorway” and “Airport” movement scenarios.

2 ROUTING PROTOCOLS

In the ad hoc networks environment, the problem of routing is essentially the distributed version of the shortest-path problem. Routing protocols for such networks should deal with the typical restriction of these networks, which includes high power consumption, low bandwidth, and high error rates[6]. As shown in Figure 2.1 these protocols basically can be divided in two groups: table-driven and on-demand.

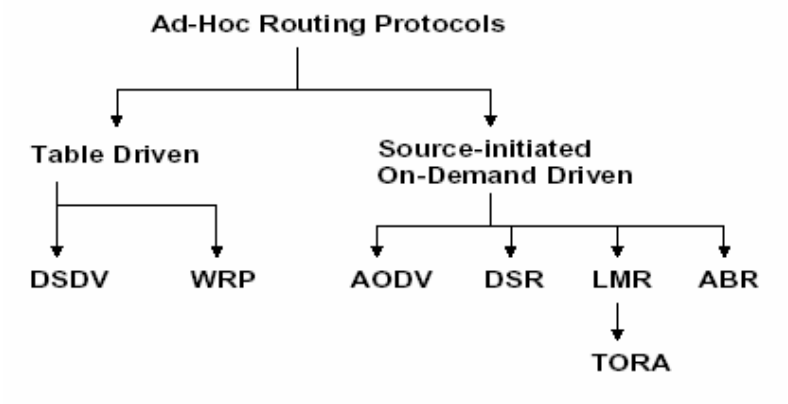


Figure 2.1 – Categorization of ad hoc routing protocols.

Table-driven protocols are similar to the routing protocols that have been developed for wired networks. Each mobile node has a routing table, which contains entries for every node in the network, and routing information is exchanged periodically. Some of the table-driven protocols follow the ideology of the link state routing and other protocols are based on Distributed Bellman-Ford. These protocols can also be called proactive protocols because they have a route to every destination in the network before they need it.

On-demand protocols do not have routes to every possible destination beforehand. At first, a node sends data and then starts searching a route to the destination. Usually the destination is discovered using a route request flood. These protocols are also called reactive protocols.

2.1 DSDV Protocol

Destination-Sequenced Distance Vector Routing (DSDV) protocol is one of the first routing protocols introduced for ad hoc networks, C. E. Perkins and P. Bhagwat published it. The protocol is very often used as a comparison protocol for new protocols. DSDV is very similar to Routing Information Protocol (RIP), which has developed for wired networks. These two protocols use the distributed Bellman-Ford (DBF) routing algorithm, which tries to find the shortest loop-free routes to every destination in the network. DSDV is an extension of the DBF routing algorithm for ad hoc networks.

In the DSDV protocol, routing information is advertised by broadcasting or multicasting the packets. These packets are transmitted periodically and incrementally, when nodes move within the network. During the time from arrival of the first route to the best route for each particular destination data is kept. On the base of this data, a decision may be made to delay advertising routes, which are going to change, in that way dumping fluctuation of the route tables. The purpose of the delaying advertisement of possibly unstable routes is to reduce the number of possible rebroadcasts of entries that arrive with the same sequence number.

2.1.1 Route Advertisement

The DVDV protocol requires for each node to advertise its own route table for all of its current neighbours (for instance, by broadcasting its entries)[1]. The entries in this list may change dynamically over time. In this case, advertisement should be made often enough to guarantee that every mobile host could almost always store every other mobile host in the collection.

Each mobile computer, in addition, agrees to transmit data packets to other computers upon request. This agreement allows determining the shortest path for a route to a destination. To avoid disturbing unnecessarily mobile hosts, which are in sleep mode, computers should exchange data between other hosts in the group. It is also possible if the target of the data is not within range for direct communication.

2.1.2 Topology Changes

As mobile nodes move from place to place almost all the time, they cause broken links as can be seen in Figure 2.2. In this example MH1 moves into the general vicinity of MH8 and MH7 and away from the others (especially MH2). The broken link can be detected by the data link protocol, or it may be inferred from the fact that no broadcasts have been received for a while from a previous neighbor. A broken link is described by a metric of ∞ (i.e., any value greater than the maximum allowed metric)[1]. In case of a broken link, any route via that link is immediately assigned an ∞ metric and an updated sequence number (for more details see section 2.1.4). Since this characterizes an essential route change, such modified routes are immediately disclosed in a broadcast routing information packet. Building information to describe broken links is the only situation in which the sequence number is generated by any mobile node. The new sequence number, which indicates ∞ hops to a destination, is one greater than the last sequence number received from the destination. In a case when a node receives an ∞ metric, and it has an equal or later sequence number with a finite metric, it triggers a route update broadcast to disseminate the important news about that destination. In this way routes containing any finite metric will replace routes generated with the ∞ metric.

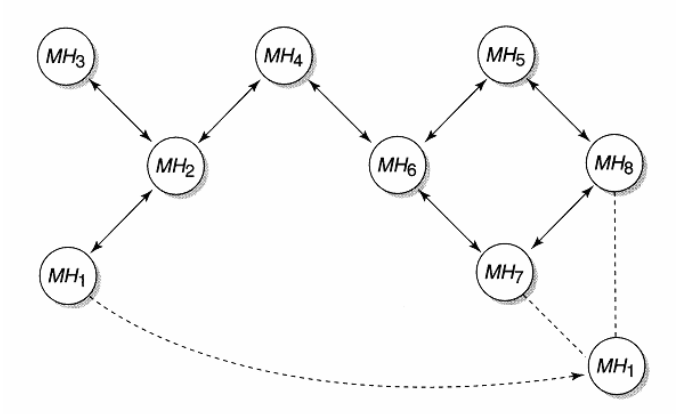


Figure 2.2 – Movement in an Ad Hoc Networks.

In networks with a very large population of mobile nodes, regulation will be needed for the time between broadcasts of the routing information packets. To reduce the amount of information carried in these packets, two types of update messages will be defined: incremental and full dump updates.

The incremental updates are sent whenever possible to save resources. These packets contain only information that has changed since the last update. The size of the packet is limited to only one network protocol data unit (NPDU). A full dump update message is sent (in a case) when the size of the incremental update is close to the size of an NPDU. This message includes the entire routing table of the node.

The length of the incremental update message is within the limits of 13 bytes to the length of the NPDU. The update message has a minimum size if it contains only the sequence number of the sender and information of the destination. The size of the full dump update is $4+9\times N$ bytes, where N is the number of the nodes in the network.

2.1.3 Route Selection Criteria

When a node receives new routing information (usually in an incremental packet), it compares this information to the information already available from previous routing information packets. After information comparison the node decides which route is preferable. There are two cases:

- Any route with a more recent sequence number is preferable.
- Routes with older sequence number equal to an existing route is chosen if it has a “better” metric and the existing route is discarded or stored as less preferable.

When the node has chosen a route from the newly received broadcast information, it will increment each metric by one hop. Newly recorded routes are scheduled for immediate advertisement to the current mobile node neighbours [1]. Routes that show a more recent sequence number may be scheduled for advertisement at a later time. This time depends on the average settling time for routes to the particular destination under consideration.

Every node must have an estimation of route settling time for every destination. These times are included in the table of the settling time. The table for each route contains two settling times: the last and the average settling time. The average settling time is calculated from the observed settling times. These times are weighted so that the recent ones have a

higher weighting factor than the ones in the past. This way the recent conditions in the network have greater effect on the estimate. When the settling time of a route goes below a predefined threshold, the route is considered to be stable.

The average settling time of a route is calculated using formula (1)

$$T_{average} = \alpha \times T_{average} + (1-\alpha) \times T_{latest} \quad (1)$$

Where α is an experimentally found constant.

$T_{average}$ is the current average settling time of the route.

T_{latest} is the latest settling time of the route.

The optimal value of α is non-trivial problem and it will need extensive studies before α can be fixed to a certain value.

Sometimes may arise a situation when a node receives new routing information in a pattern that causes it to compatibly change routes from one next hop to another, even when the destination has not moved [10]. This happens because there are two ways for new routes to be chosen:

- They might have a later sequence number.
- They might have a better metric.

A mobile host may receive two routes to the same destination with a newer sequence number, one after another (via different neighbours), but it always gets the route with the worse metric first. This will lead to a continuing burst of few route transmittals upon every new sequence number from that destination. Each new metric is propagated to every mobile host in the neighbourhood, which propagates to its neighbours, and so on.

One solution of this problem is delaying the advertisement of such routes. In this case a node can determine that a route with a better metric is likely to show up soon. Thus, a route

with a later sequence number must be available for use, but it should not be advertised immediately unless it is a route to a previously unreachable destination.

2.1.4 Routing Information

A node has to maintain two routing tables: for forwarding packets and for the route updates the node broadcasts. The tables may differ because the nodes do not advertise a destination whose route is still fluctuating, with the exception of new destinations and nodes that previously have been unreachable [6]. The nodes may also store multiple routes to the destinations, if there are any of them. The nodes should also maintain the destination sequence number and a table for the settling times of the routes.

An entry in the routing table used for forwarding packets contains:

- The address of the destination.
- The address of the next hop node.
- The metric of the route.
- The sequence number of the destination.
- A timer, some flags and a pointer.

An example of the forwarding table for MH_4 can be seen in Table 2.1, which concerns Figure 2.2 before MH_1 moves far away from MH_2. In this example, it was supposed that the address of each node represented as MH_i. Also sequence numbers are denoted $SNNN\_MH_i$, where MH_i specifies the computer that created it and SNNN is a sequence number value and entries for all other nodes, with sequence numbers $SNNN\_MH_{i,}$ before MH_1 moves away from MH_2. The install time field helps determine when to delete stale routes. After the timer of an entry has expired, the entry is deleted from the routing table. $Ptr1\_MH_i$ is a pointer. The pointer tells if there are other routing table entries for this destination. This routing table needs at least 13 bytes for every destination.

Destination	Next Hop	Metric	Sequence Number	Install	Stable Data
MH_1	MH_2	2	S406_ MH_1	T001_ MH_4	Ptr1_MH_1
MH_2	MH_2	1	S128_ MH_2	T001_ MH_4	Ptr1_MH_2
MH_3	MH_2	2	S564_ MH_3	T001_ MH_4	Ptr1_MH_3
MH_4	MH_4	0	S710_ MH_4	T001_ MH_4	Ptr1_MH_4
MH_5	MH_6	2	S392_ MH_5	T002_ MH_4	Ptr1_MH_5
MH_6	MH_6	1	S076_ MH_6	T001_ MH_4	Ptr1_MH_6
MH_7	MH_6	2	S128_ MH_7	T002_ MH_4	Ptr1_MH_7
MH_8	MH_6	3	S050_ MH_8	T002_ MH_4	Ptr1_MH_8

Table 2.1- MH_4 Forwarding Table.

Destination	Metric	Sequence Number
MH_1	2	S406_ MH_1
MH_2	1	S128_ MH_2
MH_3	2	S564_ MH_3
MH_4	0	S710_ MH_4
MH_5	2	S392_ MH_5
MH_6	1	S076_ MH_6
MH_7	2	S128_ MH_7
MH_8	3	S050_ MH_8

Table 2.2 - MH_4 Advertised Route Table.

The routing table that the node advertises to other nodes contains for every relevant destination the following information:

- The address of the destination
- The metric of the route
- The sequence number.

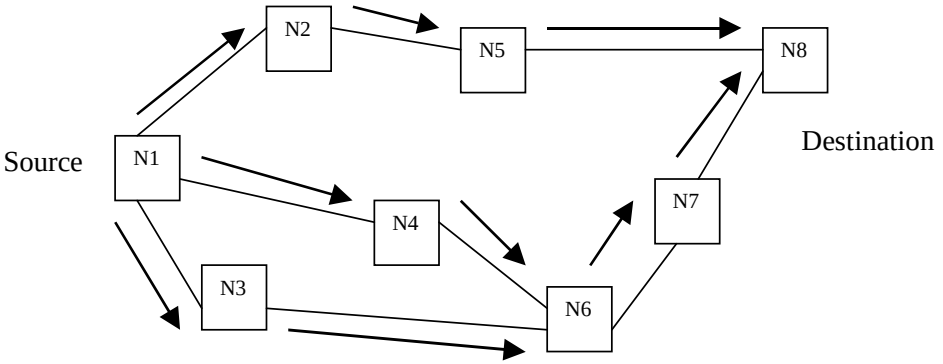
An example of the advertised table for MH_4 can be seen in Table 2.2, which also concerns Figure 2.2. The advertised routing table needs 9 bytes for every stored destination.

2.2 AODV Protocol

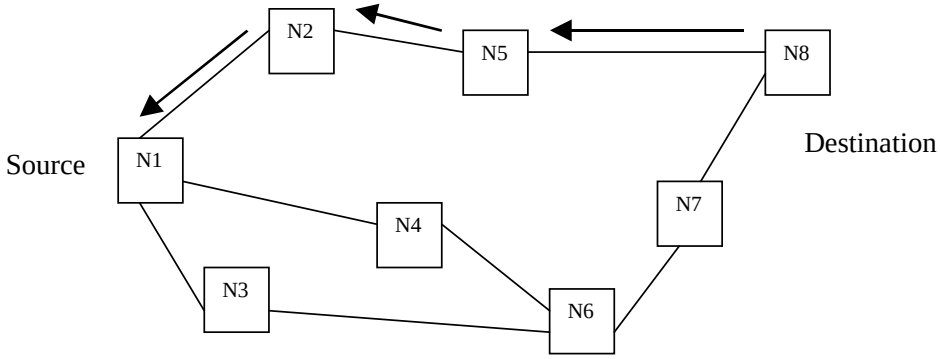
The Ad hoc On-Demand Distance Vector (AODV) algorithm is a dynamic, self-starting, multihop routing protocol between participating mobile nodes wishing to establish and maintain an ad hoc network. AODV is also one of the first protocols introduced in MANET working group in IETF. The initial design of AODV was undertaken after experience with the DSDV routing algorithm. AODV allows mobile nodes to obtain routes quickly for new destinations, and it does not require nodes to maintain routes to destinations that are not in active communication. AODV is a responsible for link breakages and changes in network topology. The operation of AODV is loop-free. To avoid the Bellman-Ford “counting to infinity” problem it offers quick convergence when the ad hoc network topology changes. In a case of link breakage, AODV causes the affected set of nodes to be notified so that they are able to void the routes using the lost link.

2.2.1 Route Establishment

When a node needs a route to some previously unknown destination or the routing table entry for the destination has become invalid, it must initiate a route discovery process. To begin such a process, the node originates a route request (RREQ) and buffers the data packets going to that destination[1][6]. A propagation of RREQ packets can be seen in Figure 2.3a. The destination sequence number field in the RREQ message is the last known destination sequence number for this destination. It is copied from the routing table[6]. If no sequence number is known, the unknown sequence number flag must be set. The originator sequence number in the RREQ message is the node's own sequence number. The RREQ ID field is incremented by one from the last RREQ ID used by the current node [5] and the hop count field is set to zero.



(a) Propagation of the RREQ



(b) Path of the RREP to the source

Figure 2.3 - AODV Route Discovery.

After broadcasting a RREQ, a node waits for a route reply message (RREP) (or other control message with current information about a route to the destination). If a route is not received within fixed time, the node may try again to discover a route by broadcasting another RREQ at the maximum Time to Live (TTL) value.

Data packets waiting for a route (i.e., waiting for a RREP) should be buffered. The buffering uses “first-in, first-out” (FIFO) algorithm. If a route discovery has been attempted RREQ_RETRIES times (a fixed amount of sent RREQ till the node has to receive an RREP) at the maximum TTL without receiving any RREP, all buffered data have to be dropped from the buffer and a destination unreachable message should be delivered to the application.

2.2.2 Expanding Ring Search

To prevent unnecessary network-wide dissemination of RREQs, the originating node should use an expanding ring search technique. When the source uses the expanding ring search technique, at first it sets the initial TTL value of PREQ. If no reply is received within the discovery period, the next PREQ is broadcast with an increment TTL value. This process of increasing the TTL value continues until a threshold value is reached, beyond which the PREQ is broadcast across the entire network up to RREQ_RETRIES more times[1].

When a new route is established, the distance to the destination is recorded in the route table. If route discovery must be initiated to this same destination later, the initial TTL value for the new RREQ is set to this distance plus the increment value.

2.2.3 Route Reply

If a node receives a route request for a destination, and either has a fresh enough route to satisfy the request or is itself the destination, the node generates a RREP message. A propagation of RREQ packets can be seen in Figure 2.3b. When the RREP is forwarded back towards the source of the RREQ message, the hop count field is incremented by one at each hop. Thus, when the originator of the RREP is received, the hop count represents the distance, in hops, of the destination from the originator. There are three kinds of RREP:

- Route reply generation by the destination. The destination must increment its own sequence number by one if the sequence number in the RREQ packet is equal to that incremented value.

- Route reply generation by an intermediate node. This node copies its known sequence number for the destination into the destination sequence number field in the RREP message.

- Gratuitous route reply. In a case of bi-directional communication, the source sets a flag in the route request to inform intermediate nodes. The intermediate node send-

ing the route reply to the source also sends this message at the same time. It creates the reverse route from the destination to the source.

The protocol uses acknowledgements to detect unidirectional links. If the link layer does not use acknowledgements for unicast packets, the node may request the recipient of a route reply acknowledgement message. If this message is not received in response to a route reply, the node has detected a unidirectional link. In this situation the node inserts the recipient of the failed route reply to a black list. All route requests received from nodes in this list are ignored. The nodes are removed from this list after a predefined time period.

2.2.4 Route Error Messages

A node initiates processing for a route error (RERR) message in three situations:

- If it detects a link break for the next hop of an active route in its routing table while transmitting data.
- If it gets a data packet destined to a node for which it does not have an active route and is not repairing (if using local repair).
- If it receives a RERR from a neighbour for one or more active routes.

In the first case, the node makes a list of unreachable destinations consisting of the unreachable neighbour and any additional destinations in the local routing table that use the unreachable neighbour as the next hop.

For the second case, there is only one unreachable destination, to which data cannot be delivered.

In the last example, the list should consist of those destinations in the RERR for which there exists a corresponding entry in the local routing table that has the transmitter of the received RERR as the next hop.

2.2.5 Local Repair

When a link break in an active route occurs, the node can repair that link locally. In a case of repairing, the node increments the sequence number for the destination and then broadcast a RREQ for that destination[5]. After that the node waits the discovery period to receive RREPs in response to the RREQ. During local repair, data packets should be buffered. If it has not received a RREP (or other control message creating or updating the route) for that destination, it will send a RERR message for that destination.

On the other hand, if the node receives one or more RREPs (or other control message creating or updating the route to the desired destination) during the discovery period, it compares the new hop count value with the value of the invalid route table entry for that destination. If the new value to the destination is greater than the hop count of the previous route, the node sends a RERR message for the destination. Then it proceeds, updating its route table entry for that destination.

A node that receives a RERR message does not delete the route to that destination; it should only retransmit this message. If the RERR arrived from the next hop, the node transmits along that route, and if there are one or more precursor nodes for that route to the destination.

Local repair of link breaks sometimes has the affect of increasing path lengths to those destinations. Since data packets will not be dropped as the RERR is received, local repair also increases the number of data packets that can be delivered to the destinations. Sending a RERR to the originator after repairing may allow the originating node to find a better route to the destination, which is based on current node positions. It does not require the originator to rebuild the route, as it is done with the data session.

If there is no other way for a node to know its neighbours, the node may use hello messages. These messages are sent if the node has not broadcast anything in a predefined interval[6]. This hello message is a RREP with the address of the current node as its destination address and the sequence number of the node as its destination sequence number.

2.2.6 Route Table Management

AODV is a routing protocol, and it deals with route table management. Route table information should be kept even for short-lived routes. A routing table contains the following entries:

- The destination IP address.
- The destination sequence number.
- The valid destination sequence number.
- Interface.
- Hop count (number of hops needed to reach destination).
- Next hop.
- The list of precursors.
- The lifetime (expiration or deletion time of the route).
- Routing Flags.
- State.

Sequence number management is critical to avoid routing loops, even when links break, and a node is not reachable. A destination becomes unreachable when a link breaks or is deactivated. When these conditions occur, the route is involving the sequence number and marking the route table entry state as invalid.

The “destination sequence number” is a sequence number that is located in a route table entry for the destination IP address for which the route table entry is maintained. It is updated whenever a node receives new (i.e., not stale) information about the sequence number from RREQ, RREP, or RERR messages that may be received from the destination. Each node in the network maintains its destination sequence number to guarantee the loop-freedom of all routes towards that node. The node determines which destinations use a

particular next hop by conferring its routing table. In this case, for each destination that uses the next hop, the node increments the sequence number and marks the route as invalid. Whenever any fresh enough (i.e., containing a sequence number at least equal to the recorded sequence number) routing information for the invalid route is received, the node has to update its route table information. A node may change the sequence number in the following cases:

- It is itself the destination node, and offers a new route to itself.
- It receives an AODV message with new information about the sequence number for a destination node.
- The path towards the destination node expires or breaks.

The AODV must know the interface over which packets are transmitted, because it should work over wired, as well as wireless networks. This includes the reception of RREQ, RREP, and RERR messages. The interface on which a packet was received from the new neighbour is recorded into the route table entry. Similarly, whenever a route to a new destination is learned, the interface through the destination is also recorded into the destination's route table entry.

For each valid route a list of precursors that may be forwarding packets on this route is maintained. These precursors will receive detection of the next hop link loss. The list of precursors in a routing table entry contains neighbouring nodes to which a route reply was generated or forwarded.

AODV uses three flags. One of the flags is used to indicate that the node wants a bi-directional route to the destination. The other two are used for multicast[6].

2.3 DSR Protocol

D.B. Johnson introduced the original idea of the Dynamic Source Routing (DSR) protocol, and the protocol has been developed further by D.B. Johnson, J. Broch, D.A. Maltz, J.G.

Jetcheva, and Y. -C. Hu. Also this protocol has been developed by the MANET working group [9] of IETF rights from the beginning of the existence of the working group. The DSR [13, 14] is a simple and efficient routing protocol designed specifically for use in multi-hop wireless ad hoc networks of mobile nodes. Using DSR, the network is completely self-organizing and self-configuring, requiring no existing network infrastructure or administration. Network nodes communicate to each other over multiple “hops” between nodes not directly within wireless transmission range of one another. DSR automatically determines and maintains changes in the network topology. Since the number or sequence of intermediate hops needed to reach any destination may change at any time, [6] the resulting network topology may be quite rich and rapidly changing.

The DSR protocol allows dynamically discover a source route across multiple network hops to any destination in the ad hoc network. The DSR protocol is composed of two following main mechanisms:

- Route Discovery. In this mechanism a source wishing to send a packet to a destination obtains a source route to it. Route discovery is used only when the source node does not already know a route to destination.

- Route Maintenance. In this mechanism the source is able to detect, while using a source route to destination, network topology changes, such as the path to the destination is broken. When route maintenance indicates a source that a route is broken, the source can use any other route, or can invoke route discovery again to find a new route to the destination. Route maintenance for this route is used only when the sender node is actually sending packets to destination.

In DSR, both of these mechanisms operate entirely “on demand”. In particular, DSR requires no periodic packets of any kind at any layer within the network. For example, DSR does not use any periodic routing advertisement, link status sensing, or neighbour detection packets, and does not rely on these functions from any underlying protocols in the network. This entirely on-demand behaviour and lack of periodic activity allows the number of overhead packets caused by DSR to scale all the way down to zero. It happens when all nodes are approximately stationary with respect to each other, and all routes to these nodes

have already been discovered. When nodes start to move more or communication patterns change, the routing packet overhead of DSR automatically scales to track only the routes currently in use. Network topology changes not affecting routes currently in use are ignored and do not cause reaction from the protocol.

The operations of both route discovery and route maintenance in DSR are designed to allow unidirectional links and asymmetric routes to be easily supported. There are three types of basic control messages: route request, route reply and route error.

2.3.1 Basic DSR Route Discovery

When some source node originates a new packet to some destination node, it places the sequence of hops to the destination in the packet header. Normally, the sender is searching previously learned routes in its “Route Cache” of routes previously learned. If it does not find a route in its cache, it will initiate the route discovery to dynamically find a new route to this destination.

An example of route discovery can be seen in Figure 2.4, here node A is attempting to discover a route to node E and the route discovery would proceed as follows [1]:

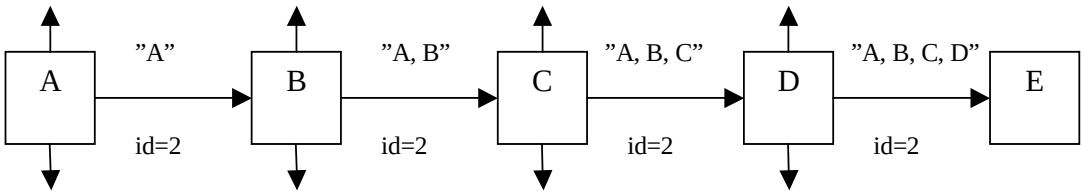


Figure 2.4 – Route discovery example.

To initiate the route discovery, node A transmits a “Route Request” as a single local broadcast packet, which is received by (approximately) all nodes currently within wireless transmission range of A, including node B in this example. The initiator of a route discovery also saves a copy of the original packet (that triggered the discovery) in a local buffer

called the “Send Buffer”. Each route request identifies the initiator and target of the route discovery. It also contains a unique request identification (2, in this example), determined by the request sender. Each RREQ also contains a record listing the address of each intermediate node through which this particular copy of the request message has been forwarded. This route record is initialized to an empty list by the route discovery initiator. In this example, the route record initially lists only node A.

When another node receives this route request (such as node B in this example), and it is the target, it will send RREP to the route discovery initiator. This route reply contains a copy of the accumulated route record from the RREQ. When the initiator receives this route reply, it caches this route in its route cache for use in sending subsequent packets to this destination.

Otherwise, if this node receiving the route request already has another RREQ from this initiator bearing this same request identification and target address, or the receiver’s own address is already listed in the route record, thus this node discards the request. Consider example from Figure 2.4, node E replying back to node A, and it will typically examine its own route cache for a route back to node A. In a case of node E found, it would use this route for the source to deliver route reply. In another case, node E should perform its own route discovery for target node A. To avoid possible infinite recursion of Route Discoveries, [8] it must piggyback this route reply on the packet containing its own route request for A.

Node E could instead simply reverse the sequence of hops in the route record that it is trying to send in the route reply, and use this as the source route on the packet [1] carrying the route reply itself.

If a new route discovery was initiated for each packet sent by a node in such a partitioned network, it would involve a network overhead. In order to reduce such overhead, a node should use an exponential back-off algorithm to limit the rate at which it initiates new route discoveries for the same target. This algorithm doubles the timeout between each successive discovery initiated for the same target.

2.3.2 Additional Route Discovery Features

Caching Overheard Routing Information

A node forwarding or overhearing any packet should add all usable routing information from that packet to its own route cache[8]. The usefulness of routing information in a packet depends on the directionality characteristics of the physical medium, as well as the MAC protocol being used. Specifically, three distinct cases are possible:

- Links in the network frequently operate unidirectionally only (not bidirectionally), and the MAC protocol in use in the network is capable of transmitting unicast packets over unidirectional links.
- Links in the network occasionally operate unidirectionally only. This unidirectional restriction on any link is not incessant, almost all links are physically bidirectional, and the MAC protocol is capable of transmitting unicast packets over unidirectional links.
- The MAC protocol is not capable of transmitting unicast packets over unidirectional links. In this case only bidirectional links can be used by the MAC protocol to transmit unicast packets.

In the first case, for example, the source route used in a data packet, the accumulated route record in a route request, or the route being returned in a route reply should all be cached by any node in the “forward” direction. The node should cache this information from any such packet received in any following modes: the packet was addressed to this node, in broadcast (or multicast), promiscuous mode. However, the “reverse” direction of the links identified in such packet headers should not be cached.

In the second example, the links described above should be cached in both directions.

In the last case, links from a source route should be cached in both directions, except when the packet also contains a route reply. In this case only the links already traversed in this

source route should be cached, but the links not yet traversed in this route should not be cached.

Replying to Route Requests using Cached Routes

When an intermediate node has received a route request, it searches a route to the target in its own route cache. If a route is found, the node generally returns a route reply to the initiator itself rather than forwarding the route request.

The route for the message is obtained either by reversing the route accumulated in the route request or by piggybacking the route reply in a route request for the source node. A route reply message contains the address of the source node, the address of the destination, a flag, an identification number, and source route accumulated to the route request. The flag indicates if the route is leading to external network for a destination. The identification number is copied from the route request.

Preventing Route Reply Storms

Sometimes broadcasting of a route request may lead to a route reply storm. This is possible when a node broadcasts a route request and several of its neighbors have a route to the requested destination. This can lead to collisions and local congestions. To prevent this the nodes use the promiscuous mode to listen the transmission of route replies from neighbors and also they delay the route request transmission. The formula (2) is used to calculate the delay (d).

$$d = H \times (h - 1 + r) \qquad (2)$$

Where h is the length of the complete route from the source to the destination,

r is a random floating point number between 0 and 1,

H is a small constant delay. It must be at least twice the maximum wireless link propagation delay[6].

If the node hears a route reply for the source before the delay time expires, it compares the route in the message to the route it has to offer. If the route in the message is shorter or the routes are equal, the node stops its timer and discards the route reply it was about to send.

Route Request Hop Limits

Each route request message contains a “hop limit”. It is used to limit the number of intermediate nodes, which are forwarding that copy of the request. This hop limit is implemented using the Time-to-Live (TTL) field in the IP header in a route request. TTL value is decremented after each transmitted hop. When the value reaches zero before finding the target, the request packet is discarded. This route request hop limit can be used to implement a variety of algorithms for controlling the spread of a RREQ during a route discovery attempt.

A node using this hop limit can implement “non-propagating” route request and “expanding ring” search technique. When a node uses the “non-propagating” route request technique, it sends its first route request a hop limit of 1, and only the neighbours will receive it. If one of them knows a route to the destination, it sends a route reply to the source. The advantages of this technique are reducing the latency of the route discovery and no flooding the whole network. If no route reply is received from neighbours, the route request can be flooded to the entire network.

When a node uses the “expanding ring” search technique, it sends an initial non-propagating route request, and if it does not received a reply message, the node originates another request with a hop limit twice as high as on the previous attempt. There is, however, the possibility that the latency of the route discovery increases because of the number of retransmitted route requests.

2.3.3 Basic DSR Route Maintenance

When originating or forwarding a packet using a source route, each node transmitting the packet is responsible for confirming that data can flow over the link from that node to the

next hop. For example, in the situation shown in Figure 2.4, node A has originated a packet for node E using a source route through intermediate nodes B, C, and D[1]. In this case, node A is responsible for the link from A to B, node B is responsible for the link from B to C, and so on.

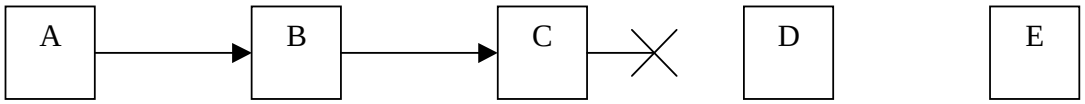


Figure 2.4 – Route maintenance example.

An acknowledgment can provide confirmation that a link is capable of carrying data. In wireless networks, acknowledgments are often provided at no cost, either as an existing standard part of the MAC protocol in use or by a "passive acknowledgment"[6]. If a built-in acknowledgment mechanism is not available, the node transmitting the packet can explicitly request a DSR-specific software acknowledgment, which can be returned by the next node along the route. This software acknowledgment will normally be transmitted directly to the sending node. In a case of unidirectional link between these two nodes, this software acknowledgment could travel over a different, multi-hop path.

After an acknowledgment has been received from some neighbour, a node may choose to not require acknowledgments from that neighbour for a brief period of time[8]. The network interface connecting a node to that neighbour always receives an acknowledgment in response to unicast traffic.

In a case of using software acknowledgment, the acknowledgment request should be retransmitted up to a maximum number of times[6]. A retransmission of the acknowledgment request can be sent in the following ways: as a separate packet, piggybacked on a retransmission of the original data packet, or piggybacked on any packet with the same next-hop destination that does not also contain a software acknowledgment.

The sender of the packet treats the link to the next-hop destination as currently "broken", if an acknowledgment request has been retransmitted the maximum number of times, and no

acknowledgment has been received. The sender removes this link from its route cache and should return a RERR message to each node that has sent a packet routed over that link.

2.3.4 Additional Route Maintenance Features

Packet Salvaging

When an intermediate node detects that the next hop along the route is broken and it has another route to the destination, the node has to “salvage” the packet rather than discard it. For this purpose, the node replaces the original source route on the packet with the route from its route cache[8]. The node then forwards the packet to the next node indicated along this source route. During salvaging the node should also return a route error (before it begins salvaging the packet) to the originator of the packet, identifying the link over which the packet could not be forwarded.

Automatic Route Shortening

Source routes in use may be automatically shortened if one or more intermediate nodes in the route become no longer necessary. This mechanism of automatically shortening routes in use is similar to the use of passive acknowledgments[8]. [6] In particular, if a node is able to overhear a packet carrying a source route (e.g., by operating its network interface in promiscuous receive mode), then this node examines the unspent portion of that source route. And if this node is also not the intended next-hop destination for the packet but is named in the later unexpended portion of the packet's source route, then it can infer that it does not longer need in the route. In this case, the node returns a “gratuitous” route reply to the original sender of the packet.

Increased Spreading of Route Error Messages

When a source node receives a route error for a data packet that it originated, the source propagates this route error to its neighbors by piggybacking it on its next request. In this way, stale information in the caches of nodes around this source node will not generate RREP messages because they contain the same invalid link.

2.3.5 Routing Information

Route Cache

All ad hoc networks routing information needed by a node is stored in the node's route cache. Each node in the network maintains its own Route Cache[8]. When a node learns new links between nodes in the network, it adds information to its route cache. A node also removes information from its route cache if it learns that existing links are broken. The route cache stores more than one route to each destination.

The cache should also contain a flag for every node in the routes. The flag tells whether the node is in the ad hoc network or belongs to an external network. This flag is for a situation where the ad hoc network is connected to some another network. The external nodes can only be at the beginning or at the end of a source route. This node cannot be an intermediate node in a route. The nodes are not allowed to send cached route replies that contain a node with the external flag set. This way the nodes will rather select routes that lead directly to the destination than routes that use the external network.

The two possible solutions for the structure of the cache are introduced, path cache and link cache. The path cache is indexed by the address of the destination [6]. The link cache contains all the known links to the node, and the routes to destinations are calculated using, for example, Dijkstra's shortest path first algorithm.

The needed memory space of route cache depends on the selected structure. The path cache needs at least $4+5\times l$ bytes memory for each entry. Here l is the average length of a source route.

Route Request Table

The route request table records information about route requests, which have been recently originated or forwarded by this node. The table is indexed by IP address. The information about the requests originated by the node include the following information:

- The time-to-live field used in the IP header of the route request used in the last route discovery for the current source and destination.
- The time for the last route request for that target node.
- The number of consecutive route discoveries since this node received a valid route reply from the target node.
- The remaining amount of time before which a node may attempt at a route discovery process for that target node.

An entry for a recently seen route request contains the address of the originator of the message and a queue of (address, identification) pairs[6]. A member of the queue contains the address of the requested destination and the unique identification of the corresponding route request message.

The route request table needs 14 bytes for each destination it requested, and 4+6×k bytes for each recently seen request. Here k is either the size of the table or the number of recently seen requests stored in the table on average. The exact definition depends on the chosen format of the table. The former definition applies to a situation where the size of the table is fixed.

Gratuitous Route Reply Table

The gratuitous route reply table contains information about “gratuitous” route replies sent by this node as part of automatic route shortening. A node uses this table to limit the rate of gratuitous route replies to the same [8] originator for the node from which it overheard a packet to trigger the gratuitous Route Reply.

Each entry in the gratuitous route reply table of a node contains the following fields:

- The address of the gratuitous route reply target node.
- The address of the overheard node.
- Entry expiring time.

Send Buffer

The send buffer is a queue of packets that cannot be sent by that node because it does not yet have a source route to each such packet's destination. Each packet in this buffer is associated with the "lived" time in the buffer, and time when it should be removed from and silently discarded. If necessary, a FIFO strategy can be used to evict packets before they timeout to prevent the buffer from overflowing. The buffer needs 4+s bytes for each packet, where s is the average size of the packet.

Network Interface Queue and Maintenance Buffer

The network interface queue is an output queue of packets from the network protocol stack. These packets are waiting to be transmitted by the network interface.

The maintenance buffer is a queue of packets sent by this node that are awaiting next-hop reachability confirmation. For each packet in the maintenance buffer, a node maintains a count of the number of retransmissions and the time of the last retransmission. The buffer will remove packets when the recipient acknowledges packets or the retransmission counter reaches a predefined threshold. In this case, the node originates a route error for the source of a removed packet.

Conclusion

In this chapter were presented two on-demand and table driven routing protocols for ad hoc networks. To sum up of this chapter, a rough comparison of the routing strategies of these protocols can be made. If consider the advantages of the table-driven protocols, they generally have better optimal routes and QoS support, lower route acquisition latency than the on demand protocols. On the other hand, pro-active protocols are inferior to reactive protocols in signalling overhead, network initialisation latency, time of the adaptation to topological changes and congestion avoidance. While this section has theoretical material and comparison of DSDV, AODV and DSR protocols, later these protocols will be compared in real network simulations.

3 SIMULATORS

Nowadays there are several ways to measure network performance: simulations, lab tests, and live tests. Considering the simulations, some following disadvantages can be picked up: it can be very difficult to create good simulations and errors in the simulator can make the results completely useless. On the other hand, the advantages of the simulations overbalance their disadvantages. There are some simulations benefits: they are cheap, easy to control in detail, and they provide the ability to experiment with hardware that is not available.

Simulation is the main tool for studying network protocols before deploying them in a wide scale. It also helps to understand how protocols will behave under various conditions[19]. Almost all network simulators are scalable simulators. They can be used as a guide for studying existing network protocols as well as a research tool for developing new protocols.

Protocols designed for ad hoc networks are complex to evaluate analytically, because they have many factors such as complex channel access protocol, node mobility, channel propagation properties, and radio characteristics. Excessive execution time of detailed models forms a barrier for the effective use of simulation. In this chapter will be highlighted some network simulators, such as GloMoSim, NS-2, QualNet and OPNET.

3.1 GloMoSim

GloMoSim (for Global Mobile information system Simulator) is a library-based sequential and parallel simulator for wireless networks. It is designed as a set of library modules, each of which simulates a specific wireless communication protocol in the protocol stack[24]. The library has been developed using PARSEC (for PARallel Simulation Environment for Complex systems), a C-based parallel simulation language. GloMoSim has been designed to be extensible and composable. It has been implemented on both shared memory and

distributed memory computers. It also can be executed using a variety of synchronization protocols.

PARSEC adopts the process interaction approach to discrete-event simulation. A logical process represents an object or set of objects in the physical system[23]. Interactions among physical processes (events) are modelled by time-stamped message exchanges. They are exchanging among the corresponding logical processes. One of the important distinctive features of PARSEC is its ability to execute a discrete-event simulation model using several different asynchronous parallel simulation protocols on a variety of parallel architectures. PARSEC is developed to separate the description of a simulation model from the underlying simulation protocol, sequential or parallel, used to execute it. A PARSEC program may also be executed using the traditional sequential (Global Event List) simulation protocol or one of many parallel optimistic or conservative protocols. Additionally, PARSEC provides powerful message receiving constructs that result in shorter and more natural simulation programs.

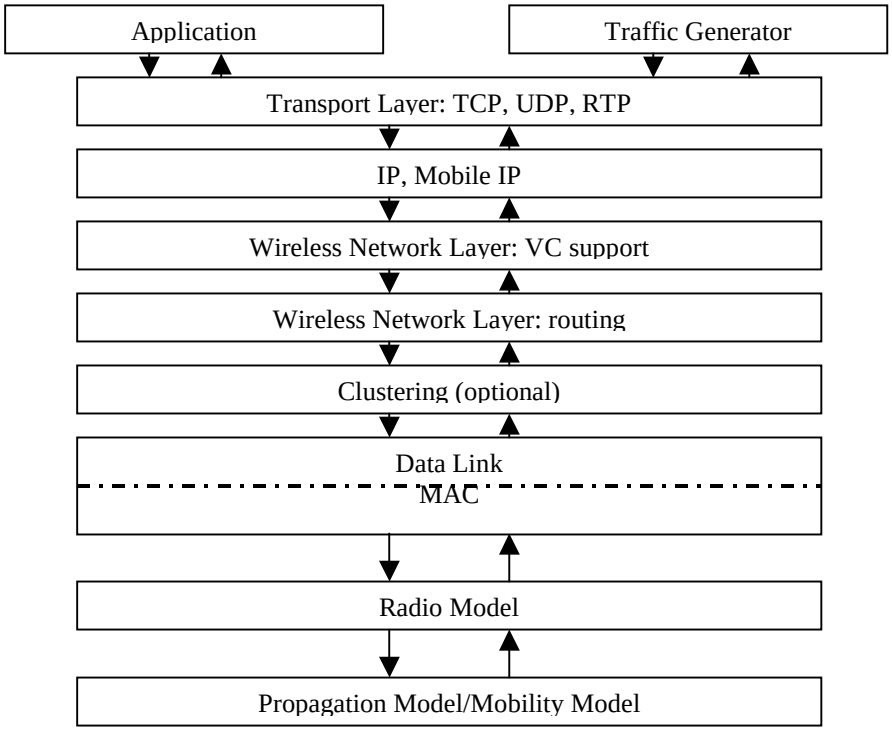


Figure 3.1 – The network stack in GloMoSim

The network stack is divided into a number of layers as shown in Figure 3.1. A number of protocols have been developed at each layer. The models of these protocols or layers can be developed at different levels of granularity.

For instance, the channel propagation layer includes a free space, an analytical and a fading channel model. A free space model calculates signal strength based only on the distance between every source and receiver pair. An analytical model computes signal attenuation using a logarithmic normal distribution. A fading channel model is computationally much more expensive but in calculating signal strength uses the effect of multi-path, shadowing and fading. GloMoSim libraries contain all these layers.

Table 3.1 lists the GloMoSim models currently available at each of the major layers. GloMoSim also supports two different node mobility models such as “random waypoint” and “random drunken” models. In “random waypoint” model a node chooses a random destination within the simulated terrain and moves with the speed specified in the configuration file. In this file pause time is also specified. In the “random drunken” model a node periodically moves to a position chosen randomly from its current neighbouring positions. The frequency of the change in node position is based on a parameter specified in the configuration file.

Since GloMoSim is a parallel simulator, it must address three sets of concerns:

1. Efficient synchronization to reduce simulation overheads.
2. Model decomposition or partitioning to achieve load balance.
3. Efficient process to processor mappings to reduce communications and other overheads in parallel execution.

The performance of GloMoSim library is evaluated as a function of the three following different conservative synchronization algorithms: null message protocol, conditional event protocol, and Accelerated Null Message protocol (ANP). ANP is a combination of the preceding two schemes[26]. The PARSEC visual front-end (PAVE) allows choice of a specific conservative runtime to be selected simply as an option in the execution command.

Layer	Models
Physical (Radio propagation)	Free space, Rayleigh, Ricean, SIRCIM
Data Link (MAC)	CSMA, MACA, MACAW, FAMA, 802.11
Network (Routing)	Flooding, Bellman-Ford, OSPF, DSR, WRP
Transport	TCP, UDP
Application	Telnet, FTP

Table 3.1 – Models currently in the GloMoSim library.

The goal of partitioning is to decompose the simulation model into a number of components. They keep the computational load approximately balanced while minimizing the communication overheads. This is attempted by assigning an approximately equal number of network nodes to each partition and an equal number of entities (or partitions) to each processor[24]. For example, a network consists of 2000 nodes, which have been placed in an 800×800 m area. The region can be partitioned in a number of ways: consider the following three partitions, respectively referred to as 4 x 4, 8 x 2 and 16 x 1 partition. Each of the various partitions have a different number of neighbors, thus the communication topology is asymmetric. This implies that cross-border message traffic and hence the computational load will be unbalanced.

In GloMoSim, each partition consists of a set of entities. Each of them simulates a certain network model executed by a group of mobile nodes. For example, given P partitions and M layers, this implies M×P entities. Since this is larger than the available number of processors, the entities must be aggregated on processors using some scheme. In GloMoSim are defined horizontal and vertical mappings or aggregation of the entities. In the first scheme, entities from all partitions that simulate the same network layer are mapped to one processor; in the latter scheme, all entities that simulate different layers within a given partition are mapped to one processor.

Simple APIs between every two neighbouring models on protocol stacks are predefined to support their composition. These APIs specify parameter exchanges and services between neighbouring layers. The simplicity of the APIs allows developers to model their protocols rapidly in an independent fashion.

3.2 NS

Network Simulator (NS) is an event driven network simulator [18] developed at UC Berkeley. It implements network protocols such as TCP and UPD, traffic source behaviour such as FTP, Telnet, Web, CBR and VBR, router queue management mechanism such as Drop Tail, RED and CBQ, routing algorithms such as Dijkstra, and more. NS also implements multicasting and some of the MAC layer protocols for LAN simulations. The NS project is now a part of the VINT project[27]. It develops tools for simulation results display, analysis and converters that convert network topologies generated by well-known generators to NS formats. Nowadays, NS (version 2) written in C++ and OTcl (Tcl script language with Object-oriented extensions developed at MIT) is available.

As shown in Figure 3.2, [20, 21] in a simplified user's view, NS is an Object-oriented Tcl (OTcl) script interpreter. It has a simulation event scheduler and network component object libraries, and network setup (plumbing) module libraries. In other words, the program should be written in OTcl script language. To setup and run a simulation network, a user should write an OTcl script that initiates an event scheduler, sets up the network topology using the network objects and the plumbing functions in the library, and tells traffic sources when to start and stop transmitting packets through the event scheduler. The term “plumbing” is used for a network setup, because setting up a network is plumbing possible data paths among network objects by setting the “neighbour” pointer of an object to the address of an appropriate object. When a user wants to make a new network object, he or she can easily make an object either by writing a new object or by making a compound object from the object library, and plumb the data path through the object. The power of NS comes from this plumbing.

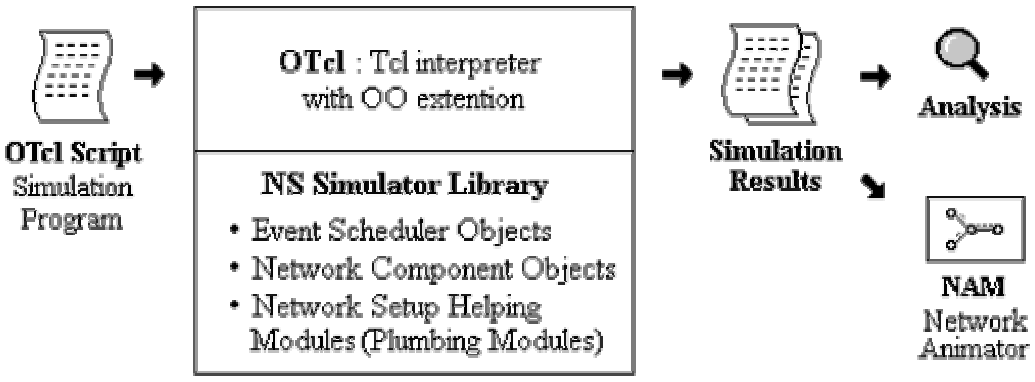


Figure 3.2 - Simplified User's View of NS

Another major component of NS is the event scheduler. An event in NS is a packet ID that is unique for a packet with scheduled time and the pointer to an object that handles the event. An event scheduler keeps track of simulation time and launches all the events in the event queue. This queue is scheduled for the current time by invoking appropriate network components. They issued the events, and let them do the appropriate action associated with the packet pointed by the event. Network components communicate with one another by passing packets. The event scheduler also serves for time handling packets (i.e. need a delay). It issues an event for the packet and waits for the event to be fired to itself. Another use of an event scheduler is timer. For example, TCP needs a timer to keep track of a packet transmission time out for retransmission (transmission of a packet with the same TCP packet number but different NS packet ID). Timers use event schedulers in a manner similar to delay[20]. The only difference is that a timer measures a time value associated with a packet and then makes action related to that packet after a certain time goes by, and does not simulate a delay [18].

NS is written not only in OTcl but in C++ also. For efficiency reasons, NS separates the data path implementation from control path implementations[21]. Using of objects, which are written in C++ language, reduces packet and event processing time (not simulation time). OTcl linkage makes these compiled objects available to the OTcl interpreter. It also matches an OTcl object for each of the C++ objects, makes the control functions and the configurable variables. In this way, the controls of the C++ objects are given to OTcl. Figure 3.3 shows an object hierarchy example in C++ and OTcl. In the figure for C++ objects,

which have an OTcl linkage forming a hierarchy, there is a matching OTcl object hierarchy very similar to that of C++.

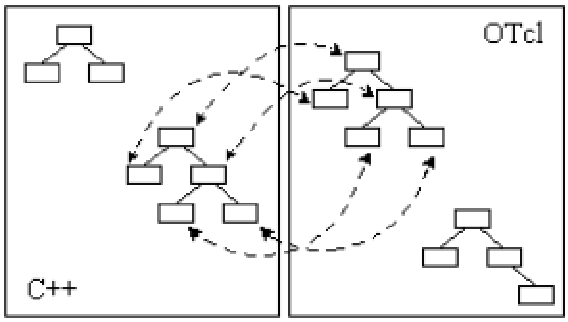


Figure 3.3 - C++ and OTcl: The Duality

Figure 3.4 shows the general architecture of NS. In this figure a general user (not an NS developer) can be thought of standing at the bottom left corner, designing and running simulations in Tcl using the simulator objects in the OTcl library. The event schedulers and most of the network components are implemented in C++ and available to OTcl through an OTcl linkage that is implemented using tclcl. The whole thing together makes NS, which is an object oriented extended Tcl interpreter with network simulator libraries.

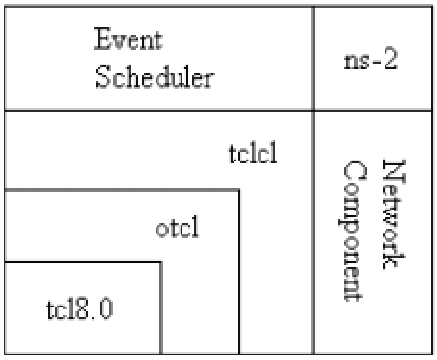


Figure 3.4 - Architectural View of NS

As shown in Figure 3.2, when a simulation is finished, NS produces one or more text-based output files that contain detailed simulation data. The data can be used for simulation analysis or as an input to a graphical simulation display tool called Network Animator (NAM). NAM is developed as a part of VINT project[22]. It has a graphical user interface similar to that of a “CD player” and it can graphically present information such as

throughput and number of packet drops at each link, although the graphical information cannot be used for accurate simulation analysis.

3.3 QualNet

QualNet is a simulator for large heterogeneous networks and the distributed applications that execute on such networks[28]. The QualNet has the following features:

- Robust set of wired and wireless network protocol and device models, useful for simulating diverse types of networks.
- Optimised for speed and scalability on one processor.
- Designed from the ground-up as a parallel simulator
- A robust graphical user interface covers all aspects of the simulation (scenario creation, topology setup, etc).
- QualNet has been used to simulate models of wireless networks with as many as 50000 mobile nodes.

The QualNet simulator is contained in the QualNet Developer toolkit and it was designed for parallel execution[25]. It also uses the parallel simulation kernel provided by the PARSEC discrete-event simulation language. QualNet includes detailed models of commonly used protocols at each of the primary layers of the protocol stack. These range from commonly used applications like file transfer (ftp) and web browsing (http) to transport, routing and MAC layer protocols. In each case, commonly used protocols in both wired and wireless networks have been modelled. Different kinds of protocols used in three model libraries such as standard, MANET and QoS libraries. QualNet model library protocols can be seen in table 3.2

Application	FTP, Telnet, CBR, HTTP, ModSAF, synthetic traffic generation, self-similar traffic with long range dependency
Transport	TCP, UDP
Routing	Bellman-Ford, OSPFv2, RIPv2, Flooding, Fisheye, DSR, LAR, AODV, ODMRP
MAC	CSMA, MACA, IEEE 802.11
Physical	point-point link, wired bus, IEEE 802.11 DSSS radio
Propagation	Analytical (free space, rayleigh), TIREM, 2-ray ground reflection, path loss trace files
Mobility	Random waypoint, group mobility, ModSAF, trace files

Table 3.2 - QualNet Model Library Protocols

QualNet defines simple APIs between neighbouring layers to enhance modular composition of protocol models developed at different layers by different designers[25]. The APIs are kept as close as possible to the operational protocol stack, such that even operational code is easily integrated into QualNet with this layered design. As the operational code has typically been extensively tested, this provides substantial benefits. The integration capability has already been demonstrated at the transport layer in QualNet by extracting the TCP Lite model from the protocol code distributed with the FreeBSD operating system. The QualNet APIs have such restriction as the network nodes can communicate with other nodes only through the lowest layer, and models at other layers cannot directly access data for other network nodes. A number of statistical metrics at each layer of the protocol stack are collected automatically by the simulator and can subsequently be used by the analyst to understand the application level performance metrics.

3.4 OPNET

The Optimised Network Engineering Tool (OPNET) Modeller is the industry's leading network technology development environment, which allows to design and study communication networks, devices, protocols, and applications with high flexibility[29].

Modeller is an object-oriented modelling approach, and its graphical editors mirror the structure of actual networks and network components. Modeller supports all network types and technologies. It is based on a series of hierarchical editors that directly parallel the structure of real networks, equipment, and protocols such as network editor, node editor and process editor.

The Network Editor graphically represents the topology of a communications network. Networks consist of node and link objects, configurable via dialog boxes. Drag and drop nodes and links from the editor's object palettes to build the network, or use import and rapid object deployment features. Use objects from OPNET's extensive Model Library, or customize palettes to contain own new node and link models. The Network Editor provides geographical context, with physical characteristics reflected appropriately in simulation of both wired and mobile/wireless networks. It allows using the protocol menu for quick protocols configuration and protocol activation.

The Node Editor captures the architecture of a network device or system by depicting the flow of data between functional elements, called "modules". Each module can generate, send, and receive packets from other modules to perform its function within the node. Modules typically represent applications, protocol layers, algorithms and physical resources, such as buffers, ports, and buses. Modules are assigned process models (developed in the Process Editor) to achieve any required behaviour.

The Process Editor uses a powerful finite state machine (FSN) approach to support specification, at any level of detail, of protocols, resources, applications, algorithms, and queuing policies. States and transitions graphically define the progression of a process in response to events. Each state of a process model contains C/C++ code, supported by an extensive library of functions designed for protocol programming. Each FSM can define private state

variables and can make calls to code in user-provided libraries. FSMs are dynamic and can be spawned (by other FSMs) during simulation in response to specific events. Dynamic FSMs dramatically simplify the specification of protocols that manage a scalable number of resources or sessions, such as TCP or ATM. Process Editor can develop entirely new process models.

OPNET uses a “pipeline” for communication between nodes. This pipeline differs depending on the type of communication link being modelled (e.g. broadcast radio versus point-to-point). The following effects in OPNET have been taken into account:

- Terrain effect. This model involves processing terrain data and calculating line-of-sign. This effect is especially important for the ad hoc network part. Once a terrain is incorporated with the ad hoc model, a mobile network with nodes moving over hills can be fully simulated with changes in transmission and reception quality.

- Error pattern. This model generates errors according to a special pattern (Markovian, Self-similar, etc) in addition to the bit error rate (BER) parameter. This will be used in simulating both the satellite and wireless channels.

- Coding scheme. This model simulates the effect of different coding/decoding schemes, which can be used in conjunction with the error pattern model and BER.

As mentioned above, OPNET's Model Library is recognized industry-wide as the most advanced suite of models of network protocols, technologies, and applications available. The Model Library is designed to work with all OPNET simulation solutions. The Model Library consists of two main parts: standard and specialized models.

Conclusion

As mentioned above, some major network simulation softwares are the free network simulator NS-2, the partially free GloMoSim, and QualNet and OPNET, which are commercial products. All are fairly complex and the choice of software to be used for this research is based on what is already available, colleagues experience, and the available budget.

Since QualNet and OPNET are commercial software, they are not suitable for further simulations. As for NS-2 and GloMoSim, they both are in the development phase and will probably remain there forever since they are continually extended for the research needs of the involved groups. As compared to commercial software, they require a greater effort to get started with but they are powerful enough for simulation purposes. Both packages are distributed freely over the Internet.

NS-2 will be used in further simulations. The decision is based on the staff experience. It has some obvious disadvantages, such as its open source nature and documentation is often limited and out of date with the current release of the simulator, but fortunately consulting the highly dynamic newsgroups and browsing the source code may solve most problems.

4 SIMULATION RESULTS

In this section will be presented the effect of increased average speed in the network to routing protocol performance metrics for two on-demand protocols (i.e. AODV and DSR) and table-driven protocol such as DSDV. Simulations will be shown for two movement scenarios, which are described in this chapter.

4.1 Physical and Data Link Layer Model

In NS-2 two radio waves propagation models were implemented, and it combines both of them depending on the distance between the transmitter and the receiver. The first model is the free-space propagation model[7]. This model generally attenuates the power of a signal as $1/d^2$ (d is the distance between the antennas), and it is used when a transmitter is within the reference distance (the crossover point) of the receiver. In a case of long distance (i.e. the receiver is outside of the reference distance), the simulator uses the second model, which is the two-ray ground reflection model. This model generally attenuates the power of a signal as $1/d^4$. A theoretical formula for calculating the received power for the free-space model is presented by formula (3). It is useful if the receiver is in LOS and $d>0$.

$$P_r(d) = \frac{P_t \cdot G_t \cdot G_r \cdot \lambda^2}{(4 \cdot \pi)^2 \cdot d^2 \cdot L} \tag{3}$$

where P_t is the transmitted power

G_t, G_r are transmitter and receiver antenna gains respectively

λ is the operating wavelength ($\lambda = c / f$, where $c=3\times10^8$ meters/sec is the speed of light and f is the operating frequency)

L is additional system losses ($L>1$)

The equation for two-ray ground reflection model can be seen in formula (4).

$$P_r = P_t \cdot G_t \cdot G_r \cdot \frac{h_t^2 \cdot h_r^2}{d^4} \quad (4)$$

where h_t and h_r are transmitter and receiver antenna heights.

Simulations were done according to the values presented in Table 4.1.

$h_t=h_r$	P_t	f	$G_t=G_r$
1,5 m	281,8 mW	914 MHz	1(Omni-directional antenna)

Table 4.1 - Values used in simulations.

In NS-2 each mobile node is characterized by the following parameters: current position and velocity, and it moves in a predefined area[7]. The position of each mobile node can be calculated as a function of time. The radio propagation model uses this calculation to compute the propagation delay from one node to another, and also to determine the power level of a received signal at each mobile node.

Each mobile node has one or more wireless network interfaces. All these interfaces have the same type (on all mobile nodes), and they are linked together by a single physical channel. In a case of packet transmission, a network interface passes the packet to the appropriate physical channel object. This object calculates the propagation delay from the packet originator to each on the channel. Then it schedules a “packet reception” event for each interface. This event informs the receiving interface about the first bit reception of a new packet. After it happens, the received power level compares to two following values: the carrier sense threshold and the receive threshold. If the received power level is less than the carrier sense threshold (in current simulation it equals $1{,}559\times10^{-11}$ W) then the packet is discarded as noise. If the received power level is above the carrier sense threshold but below the receive threshold (in current simulation it equals $3{,}652\times10^{-10}$ W), the packet is marked as a packet in error, and it is not passed to the MAC layer. In other cases, the packet is handed up to the MAC layer.

Once the MAC layer receives a packet, it should check the receiver for the “idle” status. If it is not idle, verification of the received power level occurs[7]. If the received power level of the new packet is at least 10 dB less than the previous level, then the receiver discards the new packet, and it allows the receiving interface to continue with its current receive operation. Otherwise, a collision occurs and both packets are dropped. If the MAC layer is idle, it computes the transmission time of the incoming packet and schedules a “packet reception complete” event for itself. In this case, the MAC layer confirms that the packet is error-free, performs destination address filtering, and passes the packet up the protocol stack.

4.2 Medium Access Control

In the NS-2 simulator the link layer implements the complete IEEE 802.11 standard [31] MAC protocol Distributed Coordination Function (DCF). It accurately models the contention of nodes for the wireless medium. DCF is similar to Medium Access Collision Avoidance protocol (MACA) [32] and a Media Access Protocol for Wireless LAN's MACAW [30]. DCF also is designed to use both physical carrier sense and virtual carrier sense mechanisms, which are reducing the probability of collisions due to hidden terminals. The transmission of each unicast packet is preceded by a Request-to-Send/Clear-to-Send (RTS/CTS) exchange. It reserves the wireless channel for transmission of a data packet. The receiver sends an Acknowledgment (ACK) to the sender to confirm that the unicast packet was received correctly. Sender retransmits the packet a limited number of times until this ACK is received. Broadcast packets are sent only when virtual and physical carrier senses indicate that the medium is clear. However, they are not preceded by an RTS/CTS and are not acknowledged by their recipients.

4.3 Addresses and Packet Buffering

In NS-2 simulator, the address space consists of 2 parts, the node-id and the port-id. The node's address is situated in the higher bits and the address space consists of 32 bits. The port-id or the identification of the agent attached to the node is located in the lower bits, and that space consists of 32 bits as well. One of the higher bits is assigned for multicast. Additionally, the address space may also be set in hierarchical format, consisting of multiple levels of addressing hierarchy. NS-2 also supports Address Resolution Protocol (ARP) [4]. It translates addresses from network layer protocol to hardware addresses, in other words IP addresses to MAC addresses.

Each node in NS-2 has a queue for packets awaiting transmission by the network interface. In current simulations it holds up to 50 packets and is managed in a drop-tail interface priority fashion. This fashion gives priority queue to routing packets. Each on-demand routing protocol (i.e., DSR, or AODV), can buffer separately an additional 50 packets that are awaiting discovery of a route through the network.

4.4 Performance Metrics

In these simulations, the following performance metrics were evaluated: packet delivery fraction, normalized routing to traffic ratio, average and standard deviation values of the route changes.

Packet delivery fraction – ratio of the data packets delivered to the destination to those sent by the sources.

Routing to traffic ratio – the number of routing packets normalized to total amount of packets.

The standard deviation numerically is the square root of the variance, and in turn the variance measures how given data is distributed. It can help to learn about normal distribution of data. A normal distribution of route changes means that most of the examples in a set of

route changes data are close to the “average”, while relatively few examples lead to one extreme or the other. In a normal distribution, about 68% of the route changes are within one standard deviation away from the average route changes value. The standard deviation is calculated according to Formula (5).

$$\sigma = \sqrt{\frac{\sum_{i}^{N} x_i^2}{N} - \left(\frac{\sum_{i}^{N} x_i}{N}\right)^2} \qquad (5)$$

where x_i is the amount of route changes during simulated time.

N is the amount of nodes.

Each hop-wise transmission of a routing packet is counted as one transmission.

The packet delivery fraction is one of the most important metrics to evaluate traffic efficiency [3]. The first metric shows packet losses, which are generally caused by packet collisions. Collision probability increases when the hop-count from source to destination increases. Invalidated routes also cause losses. The route usage metric shows the efficiency of the routing protocol. These three metrics are not independent. For example, if the packet path length is longer, collision is higher and it involves that the average delay decreases because the routing protocol drops the most waiting packets in the queue. Routing to traffic ratio also has an influence on the delivery fraction and delay. For instance, lower routing to traffic ratio does less network congestion.

4.5 “Motorway” Model

4.5.1 Traffic and Mobility Models

The NS-2 supports TCP and UDP transport protocols. In current simulations the transport protocol is UDP and traffic sources are Continuous Bit-Rate (CBR), and the bandwidth is equal to 2 Mbits/sec. In the “motorway” simulations it was supposed that the source-

destination pairs are spread along the side of the road and they are stationary. In Figure 4.1 these pairs are represented as four nodes, which are located some distance away from the group of the nodes. This figure is shown with help of network animator. The seed value equals to 1. Only 512-byte data packets are used and the data rate is 4 packets/sec.

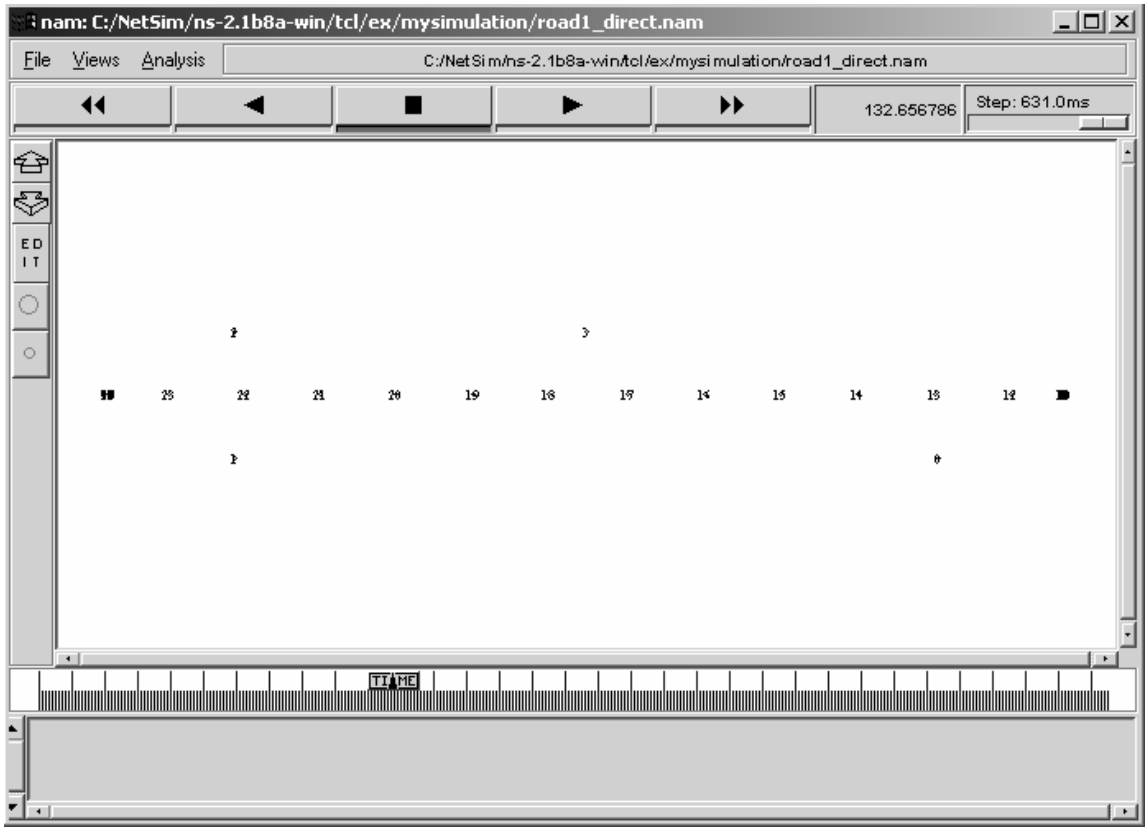


Figure 4.1 – The node movement scenario for “motorway” model.

The mobility scenario in the “motorway” model is predefined in a rectangular field. The field configuration is 3000×1000 m. In this experiment each node starts its journey from the location with coordinates X=0 and Y=500 (in the Figure 4.1 it is extreme left point). Although the road width equals 1000 m, this motorway represents a one-way road, as shown in Figure 4.1, and all nodes are moving toward the location with coordinates X=3000 and Y=500 (extreme right point). The set of experiments varied the distance between nodes and the node velocity. The node velocity is distributed between 15–28 m/s (this corresponds fairly to traffic speed on the road, which is distributed between 54–100 km/h). The nodes range is used 250 m. Simulations are run for 500 simulated seconds.

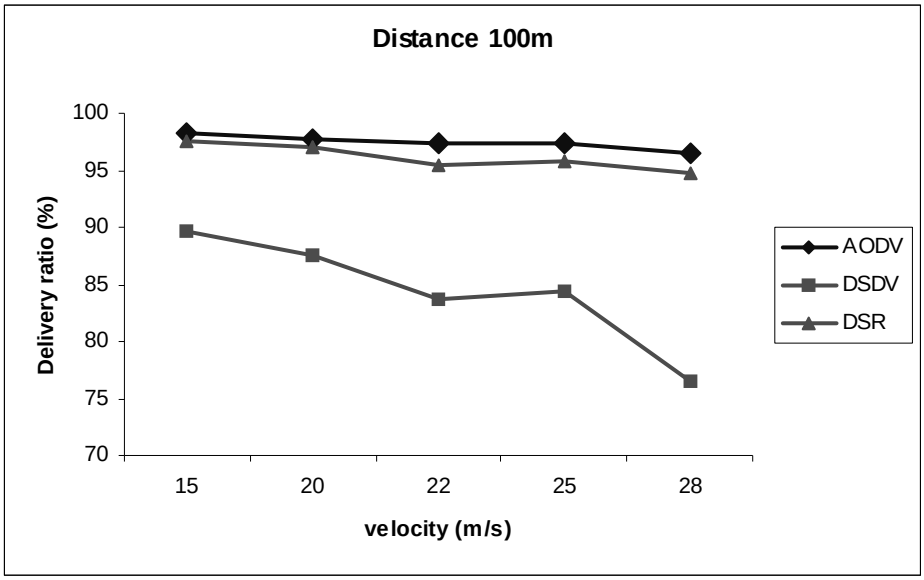
4.5.2 Simulations

This set of experiments used differing distance between nodes and changing node velocity. In current simulations the set of distances includes 100, 160, 200 and 240 meters. For each distance experiment was used 15, 20, 22, 25 and 28 m/s node speed. Since each experiment has a constant simulated time and different speed, thus for fair compliance each experiment has a various amount of nodes. The particular value for each experiment can be seen in Table 4.2.

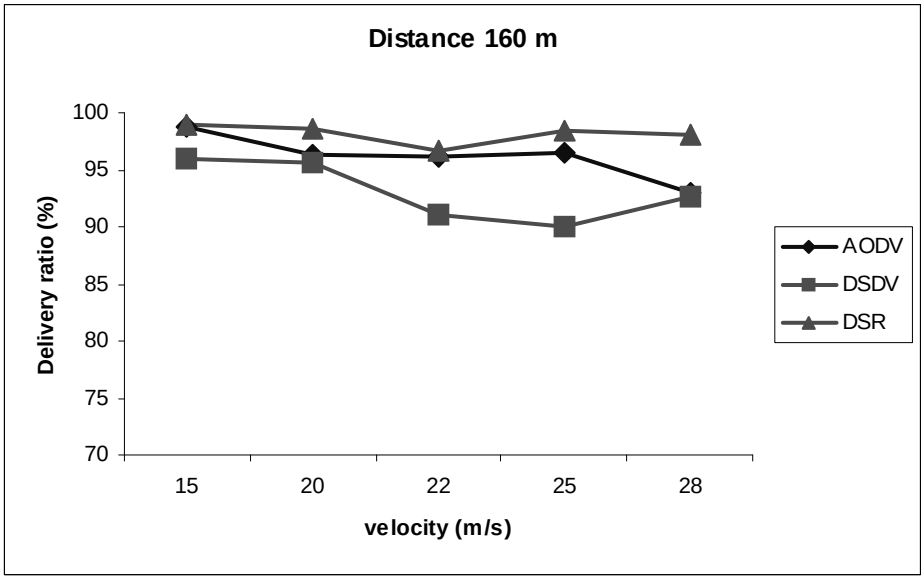
Distance (m)	Speed (m/s)				
	15	20	22	25	28
100	93	113	121	133	145
160	59	72	77	84	92
200	49	58	63	68	74
240	41	49	53	58	63

Table 4.2 – Amount of the nodes for “motorway” model

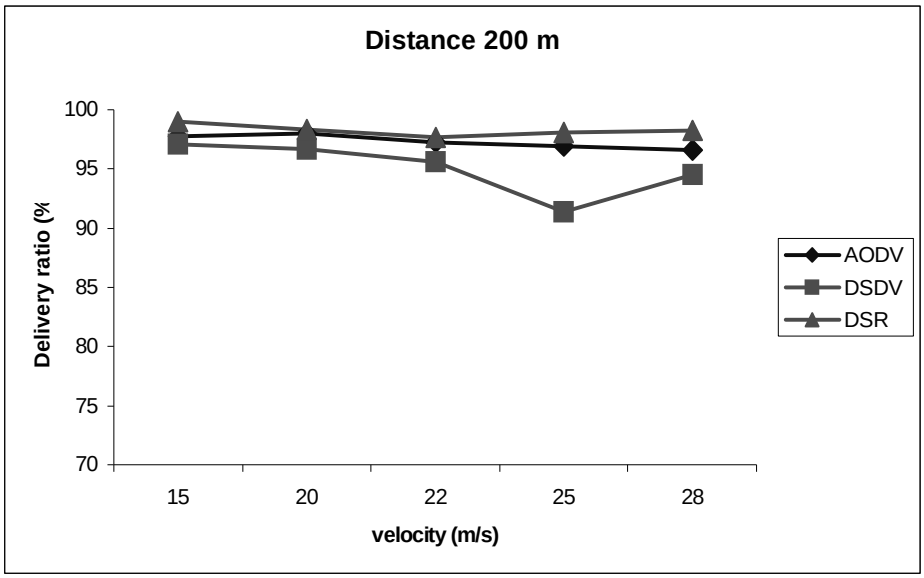
Figure 4.2 shows packet delivery ratio versus node velocity. As can be seen from this picture, the packet delivery ratio for all protocols generally decreases for higher mobility, because nodes start to drop packets because of absence of routing information. DSDV shows the worst results for all experiments, because DSDV maintains only one route per destination. Performance of AODV and DSR are very similar for all cases and almost independent for nodes velocity. The delivery for both protocols is between 94% and 100%. AODV outperform the DSR only for 100 m case for highest load (i.e. in this case the lowest distance between nodes means the highest amount of nodes).



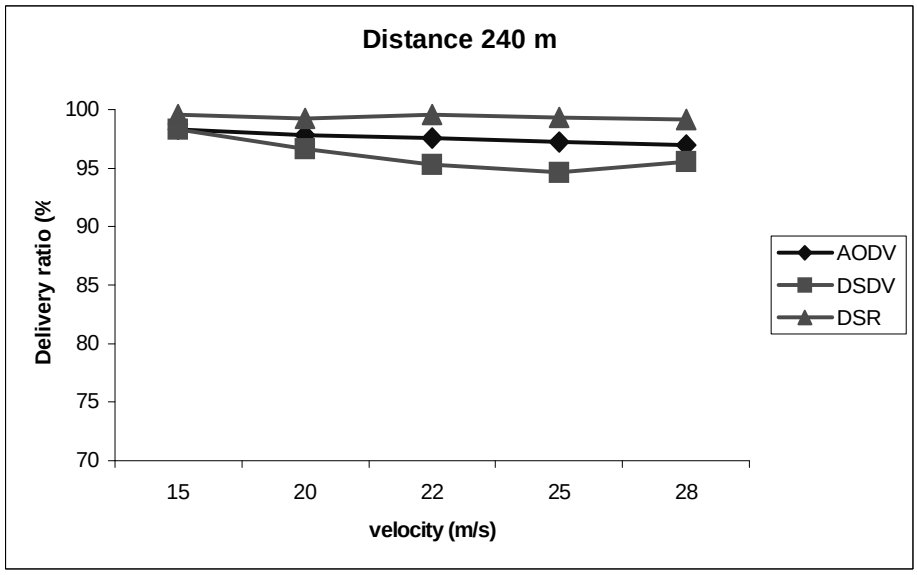
(a) 100 meters distance



(b) 160 meters distance

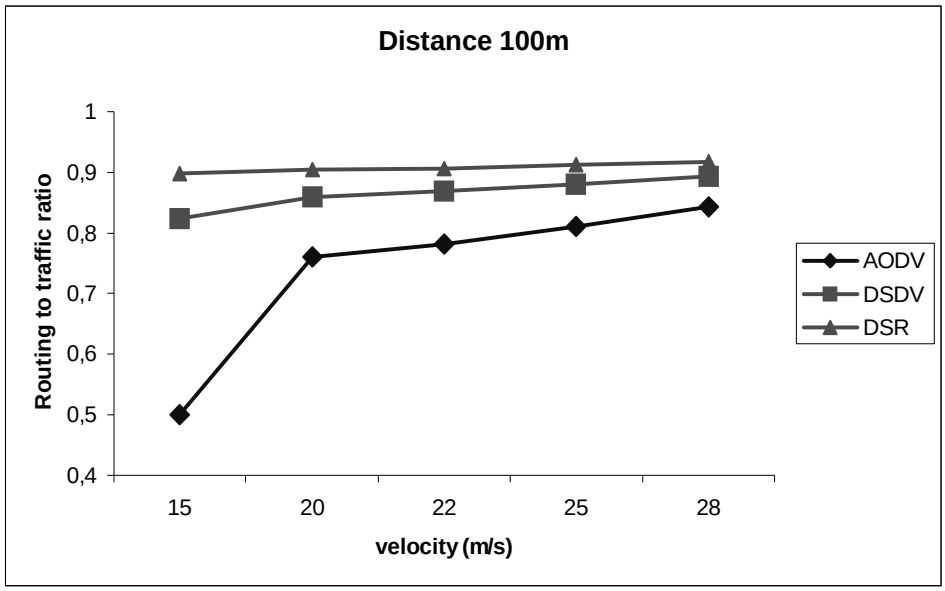


(b) 200 meters distance

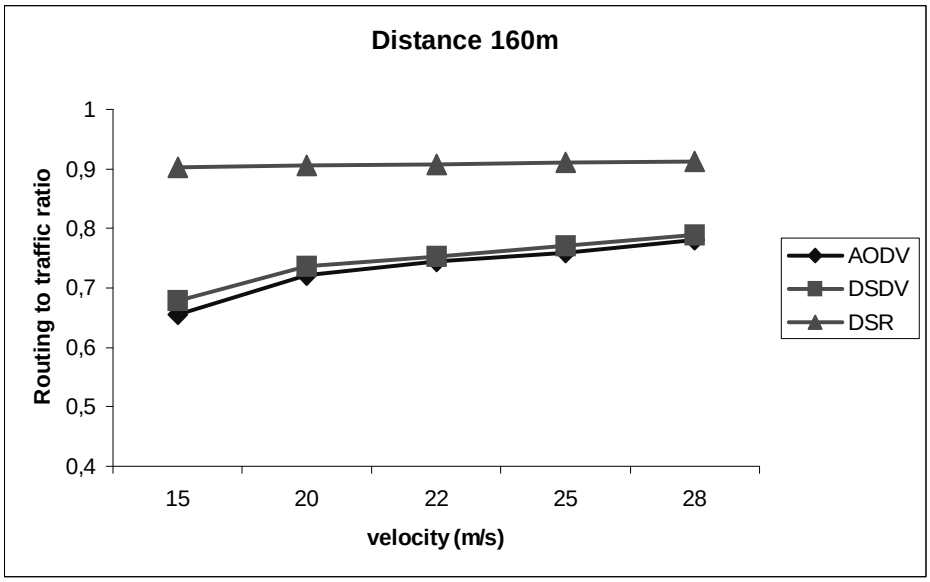


(d) 240 meters distance

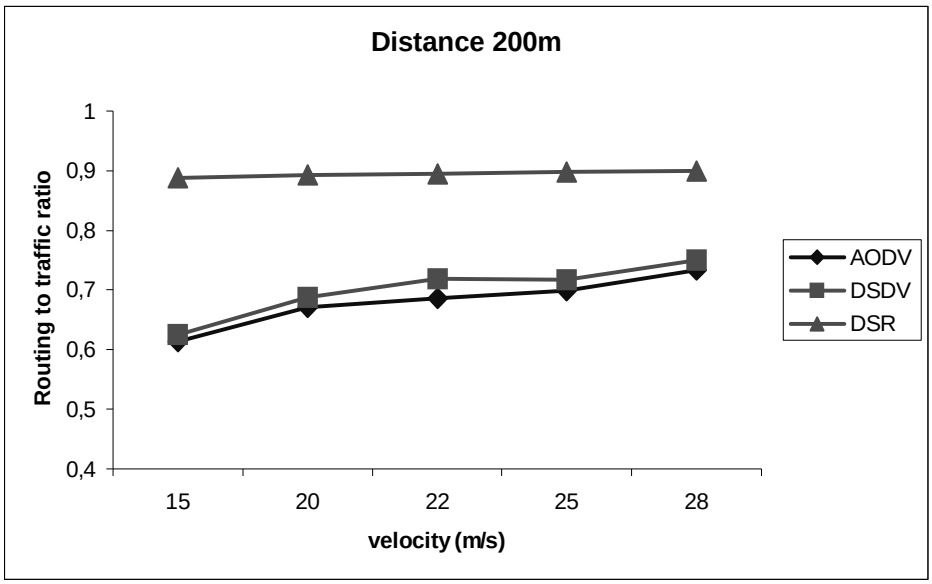
Figure 4.2 – Packet delivery ratio as a function of node velocity for various distances between nodes.



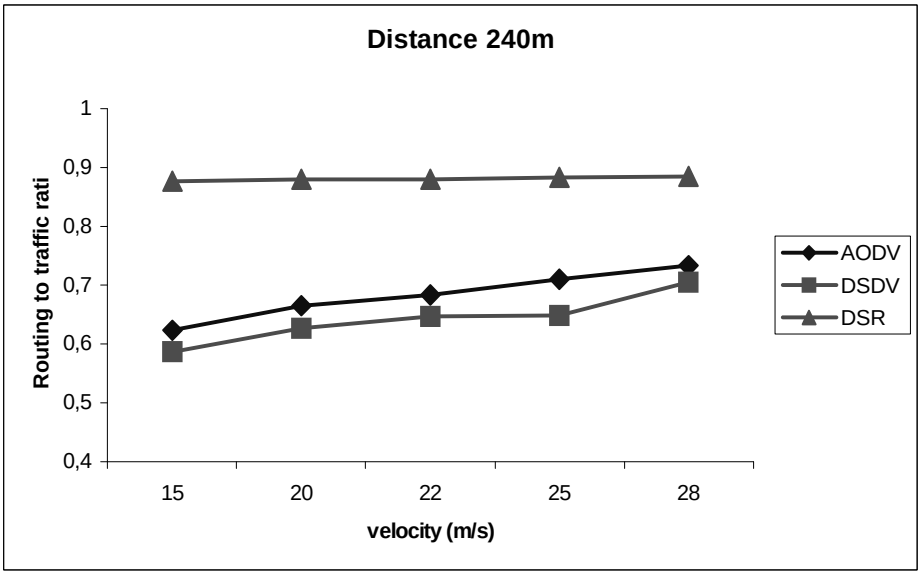
(a) 100 meters distance



(b) 160 meters distance



(b) 200 meters distance



(d) 240 meters distance

Figure 4.3 – Routing to traffic ratio as a function of node velocity for various distances between nodes.

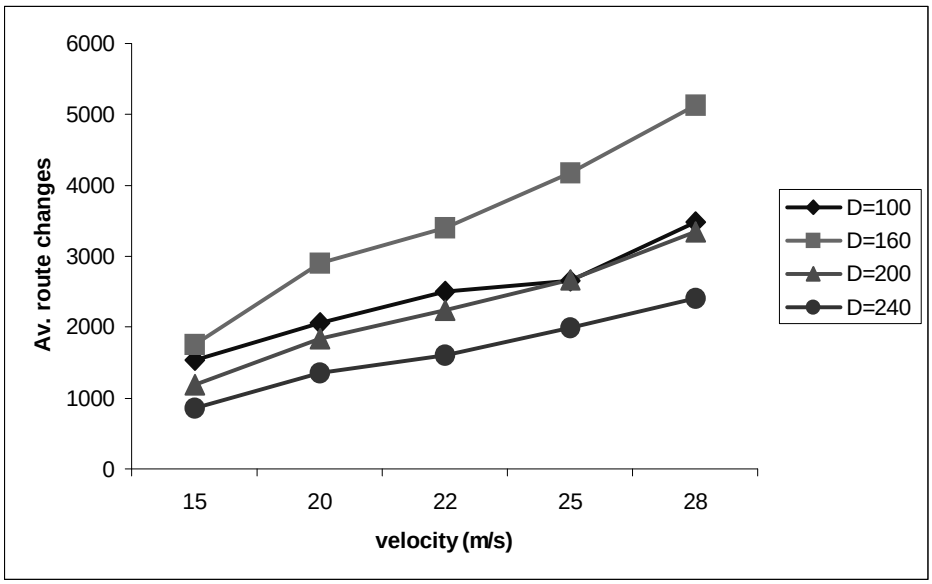


Figure 4.4 - Average value of route changes as a function of node velocity.

Distance (m)	Speed (m/s)				
	15	20	22	25	28
100	1038,09	1173,75	1974,50	1630,17	3142,12
160	489,81	786,82	924,72	1147,67	1400,58
200	315,45	441,98	585,19	636,59	967,45
240	260,00	405,39	476,53	589,80	714,09

Table 4.3 – Standard deviation value of routing changes.

The routing to traffic ratio versus node velocity shown in Figure 4.3 depends on node velocity, amount of nodes, the distance between them and nodes range. Generally the routing to traffic ratio for all protocols increases when the velocity of the node and the amount of the nodes grows. It happens because protocols demand more routing packets for greater amount of nodes. As can be seen from the figure, DSR has the greatest routing to traffic ratio because it is the most dynamic protocol and demands more routing information than the others. Also DSR has rather stable routing to traffic ratio with increasing velocity and distance between nodes. It shows a desirable property for the scalability of the protocol. AODV has a higher routing to traffic ratio in all cases except the case of 240 m distance.

The average value of route changes for set of experiments can be seen in Figure 4.4. Each data dot in the picture has a corresponding standard deviation value in table 4.3. Average route changes and “sigma” values increase linearly for higher velocity and distance between nodes for all movement scenarios. Note that a 160 m distance scenario has a somewhat highest “sigma” route changes values. “Sigma” values spread in values from 0,5 to 0,75 times of average value. It means that the maximum and the minimum values of routing changes are distributed far away from the average value.

4.6 “Airport” Model

4.6.1 Traffic and Mobility Models

The “airport” model uses the same communication model as described in section 4.5.1. In this model the source-destination pairs are also stationary, and it was assumed that all senders have only one destination.

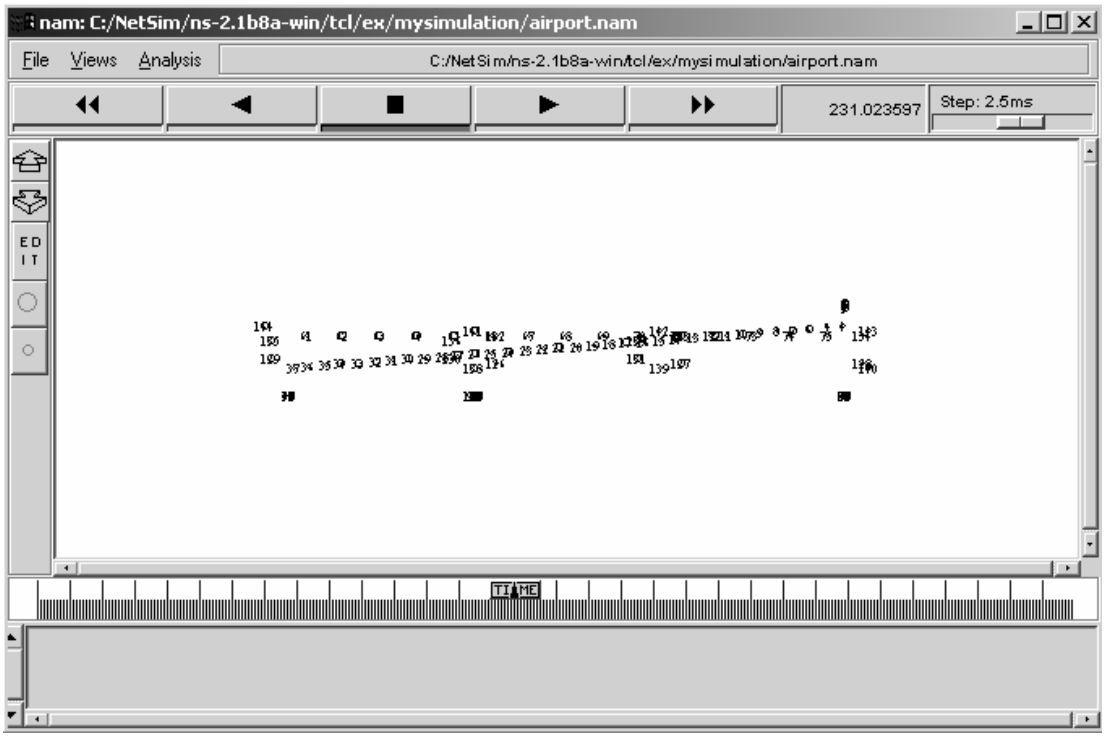


Figure 4.5 – The node movement scenario for “airport” model.

The mobility scenario in the “airport” model is also a rectangular field, which has dimensions 200m and 20 m. In the “airport” simulation the area has four passages, and, as can be seen in Figure 4.5, one of them is not used. Here, all nodes are divided in five groups; three of them are passengers moving from hall entry to airplane. The fourth group is people, which are shopping in “tax free” stores. The last group is passengers, which are waiting for their air trip. The nodes velocity is distributed between 0–1 m/s (this speed is comparable to speed of the pedestrians, which is distributed between 1-3,6 km/h). The used node range is 10 m. Simulations are also run for 500 simulated seconds. In a case of mobility, the “airport” model represents opposite instance for the “motorway” model to see protocol response in different scenarios.

4.6.2 Simulations

This experiment used the movement scenario, which is described above and includes 145 nodes. The obtained results are shown in Table 4.4.

Protocol	Metrics	
	Delivery ratio	Routing to traffic ratio
AODV	0,96	0,60
DSR	1	0,50
DSDV	1	0,61

Table 4.4 – Results from the “airport” experiment.

The experiment uses all metrics as have been described for the previous experiment. All protocols have a good delivery ratio. The average and standard deviation routing change values are 12143,12 and 1537,97 respectively. As can be seen from these numbers, the “sigma” value is about 10 times less than the average value. It allows deciding that route change values are distributed close to the average value, and it can simplify route change calculation for each node.

Conclusion

In this chapter were accurately described the simulation environment, such important models as MAC and physical models for wireless LAN standard. This simulation environment provides a powerful tool for evaluating ad hoc networking protocols and other wireless protocols.

Using this simulation environment, in this chapter were presented the result of simulations for three multi-hop ad hoc routing protocols. These protocols are AODV, DSR and DSDV. DSR and AODV are both on-demand protocols, but they are use different routing mechanisms. Each simulated protocol performs well. For all protocols, increasing the network speed involves decreasing the amount of successful delivered packets. It happens because table-driven protocols, such as DSDV, maintain their successful delivery rate as a result of their simple management of route maintenance. On-demand protocols suffer from their slowness to update routing topology changes. DSDV is effective for small ad hoc networks because it uses periodical routing table exchange among neighbor nodes and due to this feature it has a small routing to traffic ratio. The performance of DSR was very good for all movement speeds, although it had a most significant routing to traffic ratio in the “motorway” model. As shown in the “airport” model, DSR has the lowest routing to traffic ratio value. It allows deciding that DSR does not overload network for low mobility. AODV performs almost as well as DSR at all movement speeds and it uses less routing information. To sum up it can be concluded that it would be better to use DSR protocol for lower node mobility, and AODV for higher node speed.

5 Conclusion

In recent years more and more researchers have become interested in ad hoc networks, as now the availability of wireless networking and mobile computing hardware can promise to support that kind of networks. In the last few years, a set of new routing protocols, which are especially developed for ad hoc networks, have been proposed. The large number of different routing protocols shows how important and forceful this field is.

In this paper were considered the features and principles of three routing protocols for ad hoc wireless networks such as AODV, DSR and DSDV. Also the performance of these protocols was estimated and compared for “Motorway” and “Airport” models, using the NS-2 simulator. Input values for these models try to fairly correspond to real conditions. The represented simulations for accurately corresponding network layer models were obtained by using the NS-2 simulator. During the simulations it was concluded that varying the speed of the nodes, the network size, and the number of route changes influences the protocol performance.

Nowadays, unique choice of routing protocols for ad hoc networks does not exist because the values of performance metrics are situation dependent. Thus the evaluation of routing protocols helps to determine the situation when a particular protocol will be useful. As a result of this evaluation can be implemented a multi-protocol system, which changes automatically the routing protocol depending on the situation.

References

[1] Charles Perkins, *"Ad hoc networking"*, editor ISBN 0-201-30976-9

[2] J. Macker and S. Corson. *"Mobile ad hoc networks (MANET)"*. IETFWorking Group Charter, 1997. <URL: http://www.ietf.org/html.charters/manet-charter.html >, 20.07.2002.

[3] Samir R. Das, Charles E. Perkins, Elizabeth M. Royer, *"Performance Comparison of Two On-demand Routing Protocols for Ad Hoc Network"* < URL: http://citeseer.nj.nec.com/broch98performance.html >, 16.05.2002

[4] David C. Plummer. *"An Ethernet address resolution protocol: Or converting network protocol addresses to 48.bit Ethernet addresses for transmission on Ethernet hardware."* RFC 826, November 1982.

[5] Charles E. Perkins Elizabeth M. Belding-Royer, Samir R. Das, *"Ad hoc On-Demand Distance Vector (AODV) Routing"*, Internet draft < URL: http://www.potaroo.net/ietf/ids/draft-ietf-manet-aodv-11.txt >, 02.06.2002.

[6] Elizabeth M. Royer, Chai-Keong Toh, *"A Review of Current Routing Protocols for Ad Hoc Mobile Wireless Networks"*, < URL: http://citeseer.nj.nec.com/royer99review.html >, 07.09.2002.

[7] 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"*, < URL: http://citeseer.nj.nec.com/broch98performance.html >, 26.05.2002.

[8] David B. Johnson, David A. Maltz, Yih-Chun Hu, Jorjeta G. Jetcheva, *"The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks"* < URL: http://www.potaroo.net/ietf/ids/draft-ietf-manet-dsr-07.txt >, 18.08.2002.

[9] J.P. Macker and M.S. Corson (chairs), *"Mobile Ad Hoc Networks (manet),"* 1997, < URL: http://www.ietf.org/html.charters/manet-charter.html>, 28.08.2002.

[10] Xuanming Dong and Anuj Puri, *"A DSDV-based Multipath Routing Protocol for Ad-hoc Mobile Networks"*, < URL: http://www-inst.eecs.berkeley.edu/~xuanming/papers/dong-icwn02.pdf >, 06.06.2002.

[11] Jie Wu, *"Extended dominating-set-based routing in ad hoc wireless networks with unidirectional links"*, Parallel and Distributed Systems, IEEE Transactions on, Volume: 13 Issue: 9, Sept. 2002

[12] Grossglauser, M, *"Mobility increases the capacity of ad hoc wireless networks"*; Tse, D.N.C. Networking, IEEE/ACM Transactions on, Volume: 10 Issue: 4, Aug. 2002

[13] Xiaoyan Hong; Kaixin Xu; Gerla,*"Scalable routing protocols for mobile ad hoc networks"***,** M. IEEE Network, Volume: 16 Issue: 4, July-Aug. 2002

[14] Charles N. Cardinal, *"Delivery joint information superiority"*, JFQ / Autumn/Winter 1999–2000

[15] Fred C. Belen, *"Littoral Battlespace ACTD Offers Clear Combat Edge"*, National Defense, April 1998,

[16] J.J. Garcia-Luna-Aceves, Chane L. Fullmer, Ewerton Madruga, David Bayer, Thane *Frivold, "Wireless Internet Gateways"*, < URL: http://www.cse.ucsc.edu/research/ccrg/publications/chane.milcom97.pdf >, 15.09.2002.

[17] The Internet Engineering Task Force web page,<URL: http://www.ietf.org/ >, 18.05.2002.

[18] CS4514 Project 4, Transport Protocols and Router Queue Management web page < URL: http://www.cs.wpi.edu/~claypool/courses/4514-B99/projects/proj4/>, 29.08.2002.

[19] Andreas Terzis, Konstantinos Nikoloudakis, Lan Wang , Lixia Zhang, *"IRLSim: A General Purpose Packet Level Network Simulator"*, < URL: http://irl.cs.ucla.edu/papers/irlsim.pdf>, 19.07.2002.

[20] 5th UCB/LBNL Network Simulator (NS): June 1999 Tutorial < URL: http://www.isi.edu/nsnam/ns/ns-tutorial/ucb-tutorial.html>, 23.09.2002.

[21] The *ns* Manual (formerly *ns* Notes and Documentation, The VINT Project, < URL: http://www.isi.edu/nsnam/ns/doc/index.html>, 01.06.2002.

[22] Nam: Network Animator, web page, < URL: http://www.isi.edu/nsnam/nam/>, 08.06.2002.

[23] Lokesh Bajaj, Mineo Takai, Rajat Ahuja, Ken Tang, Rajive Bagrodia, Mario Gerla, *"GloMoSim: A Scalable Network Simulation Environment"*, < URL: http://citeseer.nj.nec.com/225197.html>, 12.09.2002.

[24] Xiang Zeng Rajive Bagrodia Mario Gerla, *"GloMoSim: A Library for Parallel Simulation of Large-scale Wireless Networks"*< URL: http://www.scalable-networks.com/pdf/glomosim.pdf>, 30.08.2002.

[25] Tak Kin Yung, JayMartin, Mineo Takai, and Rajive Bagrodia, *"Integration of fluid-based analytical model with Packet-Level Simulation for Analysis of Computer Networks"*,< URL: http://citeseer.nj.nec.com/458076.html>, 16.10.2002.

[26] GloMoSim Manual, web page < URL: http://pcl.cs.ucla.edu/projects/glomosim/GloMoSimManual.html>, 19.10.2002.

[27] VINT, Virtual InterNetwork Testbed web page, < URL: http://www.isi.edu/nsnam/vint/>, 25.10.2002.

[28] QualNet User Manual. < URL: http://www.scalable-networks.com/ >, 26.10.2002.

[29] OPNET User Manual. < URL: http://www.opnet.com/home.html>, 26.10.2002.

[30] Vaduvur Bharghavan, Alan Demers, Scott Shenker, Lixia Zhang, SIGCOMM, *"MACAW: A Media Access Protocol for Wireless LAN's"***,** In Proceedings of the SIGCOMM '94 Conference on Communications Architectures,Protocols and Applications, pages 212–225, August 1994.

[31] Jack Glas, Mihai Banu, Vladimir Prodanov, Peter Kiss, *"Wireless LANs"* < URL: http://citeseer.nj.nec.com/glas02wireless.html >, 01.11.2002.

[32] Phil Karn, *"MACA - A New Channel Access Method for Packet Radio"* < URL: http://icawww.epfl.ch/pdf-files/maca.pdf >, 29.10.2002.

.