Tribhuvan University

Institute of Science and Technology

ANALYSIS OF ADHOC ON DEMAND DISTANCE VECTOR (AODV) AND DYNAMIC SOURCE (DSR) ROUTING ALGORITHMS IN MOBILE ADHOC NETWORKS

Dissertation

Submitted to

Central Department of Computer Science and Information Technology

Kirtipur, Kathmandu, Nepal

In partial fulfillment of the requirements

for the Master's Degree in Computer Science and Information Technology

By

Vivek Pokhrel

December, 2011

Tribhuvan University

Institute of Science and Technology

ANALYSIS OF ADHOC ON DEMAND DISTANCE VECTOR (AODV) AND DYNAMIC SOURCE (DSR) ROUTING ALGORITHMS IN MOBILE ADHOC NETWORKS

Dissertation

Submitted to

Central Department of Computer Science and Information Technology

Kirtipur, Kathmandu, Nepal

In partial fulfillment of the requirements

for the Master's Degree in Computer Science and Information Technology

By

Vivek Pokhrel

December, 2011

Supervisor

Prof. Dr. Shashidhar Ram Joshi

Department of Electronics and Computer Engineering

Institute of Engineering, Pulchowk, Nepal

(Head)

Tribhuvan University

Institute of Science and Technology

Central Department of Computer Science and Information Technology

Student's Declaration

I hereby declare that I am the only author of this work and that no sources other than listed here have been used in this work.

… … … … … … …

Vivek Pokhrel

Date: December, 2011

Tribhuvan University

Institute of Science and Technology

Central Department of Computer Science and Information Technology

Supervisor's Recommendation

I hereby recommend that the dissertation prepared under my supervision by **Mr. Vivek Pokhrel** entitled **"ANALYSIS OF ADHOC ON DEMAND DISTANCE VECTOR (AODV) AND DYNAMIC SOURCE (DSR) ROUTING ALGORITHMS IN MOBILE ADHOC NETWORKS"** be accepted as fulfilling in partial requirements for the degree of M. Sc. in Computer Science and Information Technology.

Prof. Dr. Shashidhar Ram Joshi

Department of Electronics and Computer Engineering,

Institute Of Engineering, Pulchowk, Nepal

(Head)

Date:________________________________

Tribhuvan University

Institute of Science and Technology

Central Department of Computer Science and Information Technology

LETTER OF APPROVAL

We certify that we have read this dissertation work and in our opinion it is satisfactory in the scope and quality as a dissertation in the partial fulfillment for the requirement of Master of Science in Computer Science and Information Technology.

Evaluation Committee

____

Dr. Tank Nath Dhamala

Head, Central Department of Computer

Science and Information Technology

Tribhuvan University

____

Prof. Dr. Shashidhar Ram Joshi

Head, Department of Electronics and Computer
Engineering, Institute of Engineering

Pulchowk, Nepal (Supervisor)

____

(External Examiner)

____

(Internal Examiner)

Date: _______________

Acknowledgement

First and foremost I offer my sincerest gratitude to my supervisor, Dr. Shashidhar Ram Joshi, who has supported me throughout my thesis with his patience and knowledge whilst allowing me the room to work on my own way. I attribute the level of my Masters degree to his encouragement and effort and without him this thesis, too, would not have been completed or written. One simply could not wish for a better supervisor.

Besides my advisor, I would like to thank the rest of my thesis committee for their encouragement, insightful comments, and great suggestions.

I would also like to thank the staff members and my colleagues with all the help and facilities, which I required, for the completion of my thesis.

Most importantly, I would like to thank my family, friends and the almighty for showing me the right direction to help me stay calm in the oddest of the times and keep moving.

Abstract

Ad-hoc networking is a concept in computer communications, which means that users wanting to communicate with each other form a temporary network, without any form of centralized administration. Each node participating in the network acts both as host and a router and must therefore is willing to forward packets for other nodes. For this purpose, a routing protocol is needed.

An ad-hoc network has certain characteristics, which imposes new demands on the routing protocol. The most important characteristics are the dynamic topology, which is a consequence of node mobility. Nodes can change position quite frequently, which means that we need a routing protocol that quickly adapts to topology changes. The nodes in an ad-hoc network can consist of laptops and personal digital assistants and are often very limited in resources such as CPU capacity, storage capacity, battery power and bandwidth. This means that the routing protocol should try to minimize control traffic, such as periodic update messages. Instead the routing protocol should be reactive, thus only calculate routes upon receiving a specific request.

The Internet Engineering Task Force currently has a working group named Mobile Ad-hoc Networks that is working on routing specifications for ad-hoc networks. This master thesis evaluates some of the protocols put forth by the working group. This evaluation is done by means of simulation using Network Simulator (NS-2) from Berkeley.

The simulations have shown that there certainly is a need for a special ad-hoc routing protocol when mobility increases. More conventional routing protocols like DSDV have a dramatic decrease in performance when mobility is high. Two of the proposed protocols in this work are DSR and AODV. They perform very well when mobility is high. However, we have found that a routing protocol that entirely depends on messages at the IP-level will not perform well. Some sort of support from the lower layer, for instance link failure detection or neighbor discovery is necessary for high performance.

The size of the network and the offered traffic load affects protocols based on source routing, like DSR, to some extent. A large network with many mobile nodes and high offered load will increase the overhead for DSR quite drastically, in these situations; a hop-by-hop based routing protocol like AODV is more desirable.

Keywords: MANETS, DSR, AODV, NS2- network simulator

Abbreviations

ACK	Acknowledgement
CBR	Continuous Bit Rate
DSDV	Destination Sequenced Distance Vector
DARPA	Defense Advanced Research Project Agency
EPPCSIT	Emerging Principles and Practices of Computer Science and Information Technology
ETSI	European Telecommunications Standards Institute
FC4	Fedora Core 4
FTP	File Transfer Protocol
GUI	Graphical User Interface
IEEE	Institute of Electrical and Electronics Engineers
IP	Internet Protocol
LN	Local Area Network
MAC	Medium Access Control
MANET	Mobile Ad hoc Network
MH	Mobile Hosts
NAM	Network Animation
NS	Network Simulator
OSI	Open System Interconnect
OTCL	Object Oriented Tool Command Language
PAN	Personal Area Network
PHY	Physical
TCL	Tool Command Language
TCP/IP	Transmission Control Protocol/Internet Protocol
Tx	Transmission
UDP	User Datagram Protocol
VINT	Virtual Inter Network Test-bed
WPAN	Wireless Local Area Network

Table of Contents

List of figures

List of tables

Chapter 1: Ad-hoc Network

1.1 Introduction

Wireless communication systems continue to show rapid growth as a result of significant advancements in digital communications, commercial laptop computers, and semiconductor technologies. The most popular networks of the traditional wireless model are cellular and mobile IP networks, which have been configured with a wired backbone, where the last hop is a wireless link, essentially a point-to-point wireless channel between the base station and the mobile user. In the wireless cell domain, the base station provides centralized control for the mobile users to access the medium. Some representative specifications are IS-54 (the first generation of the digital standard with TDMA technology), IS-95 (the standard for CDMA), GSM, cdma2000, and W-CDMA. For the past few years, mobile ad hoc networks (MANETs) have been emphasized as an emerging research area due to the growing demands for mobile and pervasive computing, where the dynamic topology for the rapid deployment of independent mobile users becomes a new factor to be considered. For instance, mobile users across a campus can transmit data files to each other, group members of a search, disaster rescue, recovery team, or military solders in a battlefield can communicate in order to coordinate their actions, without using a base station. Especially, in battlefield circumstances, the infrastructure may not be built in advance for soldiers to communicate with each other. These example networks are called ad hoc wireless networks where centralized and organized connectivity cannot be possible. The examples show that MANETs need to have the ability to provide for establishing survivable, efficient, dynamic communication for emergency, search-and-rescue operations, disaster relief efforts, law enforcement in crowd control and military networks. One of the outstanding features of MANETs could be the self-creating, self-administrating, self-organizing, and self-configuring multihop wireless network characteristic.

MANETs differ from conventional cellular networks because all links are wireless and the mobile users communicate with each other without using a base station. Several basic properties of MANETs are described below. An autonomous collection of mobile users composes a MANET, where they communicate over relatively bandwidth constrained wireless links. MANETs use peer-to-peer wireless connections, where the packets from a source node are transmitted via intermediate nodes called relay nodes towards a destination node. A MANET topology dynamically changes as mobile users join, leave, or rejoin the network. Sometimes, radio links in a MANET may not be usable due to the node mobility.

Most research and development (R&D) funding for MANET applications are for military applications for defense and security systems, where the Office of Naval Research (ONR) and the Defense Advanced Research Project Agency (DARPA) of the U.S. Department of Defense (DoD) have been leading the research in this area. Other MANET related applications are found in government-industry funded projects such as the R&D of Intelligent Transportation Systems (ITS). One important issue in MANETs is the time varying network topology which may be unpredictable over time and, therefore, MANET outing algorithms must keep updating their neighbor discovery data and inform the nodes of the network topology change due to node mobility. The MANET working group (WG) of the Routing Area of the Internet Engineering Task Force (IETF) has defined and standardized IP routing functionality suitable for MANET wireless routing applications within both static and dynamic topologies. Several MANET routing protocols ([1] and [2]) have been accepted as Internet Standards or are under development as Internet drafts ([3], [4], [5], [6], and [7]) under the IETF.

1.2 Motivation

The objective for this thesis was to evaluate proposed routing protocols for wireless ad-hoc networks based on performance. This evaluation should be done theoretically and through simulation. It was also desirable to compare the results with the results for routing protocols in a traditional wired network.

The thesis also included the goal to generate a simulation environment that could be used as a platform for further studies within the area of ad-hoc networks. This simulation environment was based on Network simulator 2 from Berkeley.

The goal of this thesis was to:

- Get a general understanding of ad-hoc networks
- Generate a simulation environment that could be used for further studies
- Implement the proposed routing protocols, AODV and DSR, for wireless ad-hoc networks
- Analyze the protocol theoretically and through simulation using NS-2

- Produce a classification of the protocols with respect to applicability in combinations of small/large networks, and mobile/semi-mobile nodes
- Recommend protocols for specific network scenarios

Despite, MANET nodes have the capability to cooperate in routing each others' data packets. Due to the lack of any centralized control and possible node mobility in MANETs, many issues at the network, medium access, and physical layers currently become new research topics since no counterparts in the wired networks like Internet, or in cellular networks can satisfy these MANET requirements.

1.2.1. Network Layer in MANET

At the network layer, the main problem is that of routing, which is awfully deteriorated by the time-varying network topology, power constraints, and the characteristics of the wireless channel. MANET routing protocols can be categorized into proactive, reactive, hybrid, hierarchical and location-based protocols. For a MANET to have trustworthy routing protocol, several factors should be considered.

a) The mechanisms for neighbor discovery, topology update, route discovery, route maintenance, data forwarding, link error-checking, and link error recovery when nodes power up, reboot, join, leave and rejoin the network, could be definitely one consideration factor in MANET routing protocols.
b) Performance issues such as network-imposed delay, delay variance, reliability defined as the average loss ratio of the medium by the routing/switching design, the number of hops per route, route discovery time, routing traffic (bps), end-to-end delay, hop-by-hop and end-to-end packet delivery ratio, number of data packets transmitted, number of control bytes transmitted, effect of the traffic load on the routing protocol, and effect of node mobility on the routing protocol should also be studied to satisfy a guaranteed QoS.
c) The technology to setup self-organized wireless interconnection of communication devices in dynamic topologies also needs to be considered.
d) Standardization for different MANET routing protocols to implement interoperability in heterogeneous MANET networks will be demanded to be incorporated within other MANET routing protocols. As node mobility increases, the control routing overhead also increases dramatically, this causes problems with the allocation of network resources. The

scalability issues in MANET routing protocols have been continuously studied to reduce routing control overhead. Due to the different link characteristics in opposite directions in a wireless link, the implementation of a symmetric route from a source to a destination in MANET routing protocol becomes one challenging research topic.

e) With well-defined MANET protocol models, wireless network capacity could be mathematically analyzed to define the maximum network throughput. One of the on-going research areas focuses on defining the maximum throughput in MANETs to be used as a reference guide.

f) Interoperability issues with the other layer stacks such as TCP/UDP protocols, and RSVP/LDP signaling protocols, is one of the primary challenges to design compatible MANET routing protocols.

g) To reduce the number of vulnerabilities, security mechanisms which include authorization and admission control are being developed, and it is an open research area.

h) Assigning IP address to mobile nodes can be expanded as another research topic in MANETs.

i) Another research topic in MANETs focuses on the implementation of multicasting, in which a MANET node can send a data packet to multiple destinations in a group.

1.2.2. MAC Layer in MANET

MANETs consist of a number of mobile users that communicate with each other over a wireless channel, which causes an issue regarding on how to share a wireless medium among all the users. Due to the time-varying network topology and the lack of centralized control, the choice of the medium control access (MAC) scheme technology is also difficult in ad hoc networks. The ultimate purpose of the MAC is to establish the mechanism for traffics, and to provide the classification for the different requirements of each traffic class. Two types of multiple access protocols such as the contention-based protocol and the contention-free protocol have been developed. TDMA (Time Division Multiple Access), FDMA (Frequency Division Multiple Access), CDMA (Code Division Multiple Access), Token Ring, and DQDB (Distributed Queue Dual Bus) are widely used as the contention-free multiple access protocols. On the other hand, Contention-based protocols can be classified into random access and collision resolution methods based on the methods used to resolve the packet collisions. ALOHA, CSMA (Carrier Sensing Multiple Access), BTMA (Busy Tone Multiple Access), ISMA (Idle Signal Multiple Access) are well-known random access contention-based protocols, in which they use a random delay before resending conflicted packets. The TREE and WINDOW protocols use a sophisticated method to solve the packet retransmission instead of using a random delay. The increasing complexity in TDMA due to the non-centralized control or dynamic assignment of frequency bands in FDMA could not be the best solution in MANETs. The FDMA scheme in MANETs tends to be inefficient when the MANET becomes densely populated. One issue in CDMA is the need to keep track of frequency hopping patterns and/or spreading codes of the time-varying neighborhood. CSMA/CA (Carrier Sensing Multiple Access/Collision Avoidance) has been standardized in the IEEE 802.11 (IEEE stands for Institute of Electronics and Electrical Engineering) and can be one of the possible MAC protocols for MANETs. Since the birth of the ALOHA protocol, several variants have been developed. The difference between ALOHA and CSMA is that in ALOHA protocol, when a sender transmits a packet, it does not check whether the channel is busy or not. However, in CSMA protocol when the transmitter sends a packet, it listens and checks whether the channel is busy or not in order to prevent potential packet collisions. Therefore, the CSMA protocol could achieve a better throughput accomplished by listening to the channel before transmitting a packet. Even though two transmitters in the CSMA protocol detect the collision when they transmit packets, they don't stop transmitting packets, but continue sending packets and complete the packet transmission, and this transmission occupies the wireless medium uselessly for an entire packet time and the transmitted packets eventually collide. However, in the CSMA/CD protocol, whenever two nodes detect a collision when transmitting packets, they stop the transmissions immediately. CSMA/CD can only detect packet collisions and there is no collision avoidance mechanism. However, CSMA/CA protocol has a special request to send (RTS) and clear to send (CTS) hand-shaking mechanism to avoid packet collisions before sending a packet. The distributed coordination function (DCF) defines the mechanism of RTS and CTS frames prior to the transmission of the actual data frame. Based on the above discussions, for a MANET to have trustworthy medium access control protocol, several factors should be considered as follows.

a) The selection of the layer 2 mechanisms among TDMA, FDMA, DTDD (Dynamic Time Division Duplex), CDMA, ALOHA, CSMA/CD and CSMA/CA is a difficult choice for MANET medium access control. Many simulation results based on OPNET, MATLAB, ns-2, GlomoSim and QualNet with various scenario analyses are published to show the various performances of the MAC schemes.

b) Performance issues such as utilization of bandwidth, packet delay rate, channel busy probability, packet blocking rate, packet dropping rate, packet error rate, and throughput could be measured to evaluate the various MAC schemes.

c) Various scheduling issues have also been studied to provide QoS differentiation over the scheduling switch.

d) The technology of preemptive or non preemptive priority-based access control scheme for broadband MANETs has also been considered by many researchers.

e) Error control schemes such as automatic repeat request (ARQ) to achieve reliable data transmission over wireless transmission links have been proposed.

1.2.3. Physical Layer in MANET

At the physical layer, power control is one of the most important issues, and the focus is on getting the sufficient transmission range of a node, which needs to be controlled so that it is wide enough to reach the intended receiver, while causing minimal interference to other nodes.

a) Based on the parameters coming from the physical layer, several efficient power aware routing protocols and battery cost routing protocols have been proposed.

b) To get higher wireless channel capacity, Multi-Input-Multi-Output (MIMO) systems, in which both the transmitter and the receiver have multiple antennas, are currently under study by many researchers.

c) To get different QoS levels in the physical layer, Dynamic Time Division Duplex (DTDD) can be used, where portions of the downlink and uplink bandwidth in cellular network are dynamically assigned. In Static TDD (STDD) the portions of downlink and uplink bandwidth in cellular networks are fixed. This could be one challenging research topic.

d) To get better signal-to-interference (SIR) ratio at the receiver, many research studies to avoid the co-channel interference (CCI) and inter-symbol interference (ISI) have been addressed.

1.3 Thesis Organization

This report consists of 6 chapters and one appendix. Chapter 1 explains the concept of ad-hoc networks and routing in general. Chapter 2 is the collection of literature reviews. Chapters 3 and 4 describe the different routing protocols and simulation background necessary for the performance output. Chapter 5 discusses the simulation and results. Chapter 6 concludes the whole report and project further work. And finally, Chapter 7 is the reference that we have used. The appendices contain some terminology, details about the implementation of AODV that we did for the simulator and some screenshots of the simulator.

1.4 Ad-hoc Characteristics

Ad-hoc networks are often characterized by a dynamic topology due to the fact that nodes change their physical location by moving around. This favors routing protocols that dynamically discover routes over conventional routing algorithms like distant vector and link state [8]. Another characteristic is that a host node has very limited CPU capacity, storage capacity, battery power and bandwidth, also referred to as a "thin client". This means that the power usage must be limited thus leading to a limited transmitter range.

The access media, the radio environment, also has special characteristics that must be considered when designing protocols for ad-hoc networks. One example of this may be unidirectional links. These links arise when for example two nodes have different strength on their transmitters, allowing only one of the hosts to hear the other, but can also arise from disturbances from the surroundings. Multihop in a radio environment may result in an overall transmit capacity gain and power gain, due to the squared relation between coverage and required output power. By using multihop, nodes can transmit the packets with much lower output power.

1.5 Routing

It may be necessary to hop several hops (multi-hops) before a packet reaches the destination, a routing protocol is needed. The routing protocol has two main functions, selection of routes for various source-destination pairs and the delivery of messages to their correct destination. The second function is conceptually straightforward using a variety of protocols and data structures (routing tables). This work is focused on selecting and finding routes which would be discussed later in the chapter 3.

Chapter 2: Literature Review

2.1. Routing

Conventional routing protocols like link state and distance vector cannot perform as desired for the mobile ad-hoc networks. They are well tested and most computer communications people are familiar with them. The main problem with link-state and distance vector is that they are designed for a static topology, which means that they would have problems to converge to a steady state in an ad-hoc network with a very frequently changing topology.

Link state and distance vector would probably work very well in an ad-hoc network with low mobility, i.e. a network where the topology is not changing very often. The problem that still remains is that link-state and distance vector is highly dependent on periodic control messages. As the number of network nodes can be large, the potential number of destinations is also large. This requires large and frequent exchange of data among the network nodes. This is in contradiction with the fact that all updates in a wireless interconnected ad hoc network are transmitted over the air and thus are costly in resources such as bandwidth, batter power and CPU. Because both link-state and distance vector tries to maintain routes to all reachable destinations, it is necessary to maintain these routes and this also wastes resources.

Another characteristics for conventional protocols are that they assume bi-directional links, e.g. that the transmission between two hosts works equally well in both direction. In the wireless radio environment this is not always the case.

Because many of the proposed ad-hoc routing protocols have a traditional routing protocol as underlying algorithm, it is necessary to understand the basic operation for conventional protocols like distance vector, link state and source routing.

2.1.1. Link State

In link-state routing [8], each node maintains a view of the complete topology with a cost for each link. To keep these costs consistent; each node periodically broadcasts the link costs of its outgoing links to all other nodes using flooding. As each node receives this information, it updates its view of the network and applies a shortest path algorithm to choose the next-hop for each destination.

2.1.2. Distance Vector

In distance vector [8] each node only monitors the cost of its outgoing links, but instead of broadcasting this information to all nodes; it periodically broadcasts to each of its neighbors an estimate of the shortest distance to every other node in the network. The receiving nodes then use this information to recalculate the routing tables, by using a shortest path algorithm.

Compared to link-state, distance vector is more computation efficient, easier to implement and requires much less storage space. However, it is well known that distance vector can cause the formation of both short-lived and long-lived routing loops. The primary cause for this is that the nodes choose their next-hops in a completely distributed manner based on information that can be stale.

2.1.3. Source Routing

Source routing [8] means that each packet must carry the complete path that the packet should take through the network. The routing decision is therefore made at the source. The advantage with this approach is that it is very easy to avoid routing loops. The disadvantage is that each packet requires an extra overhead.

2.1.4. Flooding

Many routing protocols uses broadcast to distribute control information, that is, send the control information from an origin node to all other nodes. A widely used form of broadcasting is flooding [8] and operates as follows. The origin node sends its information to its neighbors (in the wireless case, this means all nodes that are within transmitter range). The neighbors relay it to their neighbors and so on, until the packet reaches all nodes in the network. A node will only relay a packet once and to ensure this some sort of sequence number can be used. This sequence number is increased for each new packet a node sends.

2.1.5. Classification

Routing protocols can be classified [9] into different categories depending on their properties.

- ❖ Centralized vs. Distributed
- ❖ Static vs. Adaptive

- Reactive vs. Proactive

One way to categorize the routing protocols is to divide them into centralized and distributed algorithms. In centralized algorithms, all route choices are made at a central node, while in distributed algorithms, the computation of routes is shared among the network nodes.

Another classification of routing protocols relates to whether they change routes in response to the traffic input patterns. In static algorithms, the route used by source-destination pairs is fixed regardless of traffic conditions. It can only change in response to a node or link failure. This type of algorithm cannot achieve high throughput under a broad variety of traffic input patterns. Most major packet networks uses some form of adaptive routing where the routes used to route between source-destination pairs may change in response in congestion.

A third classification that is more related to ad-hoc networks is to classify the routing algorithms as either proactive or reactive. Proactive protocols attempt to continuously evaluate the routes within the network, so that when a packet needs to be forwarded, the route is already known and can be immediately used. The family of Distance-Vector protocols is an example of a proactive scheme. Reactive protocols, on the other hand, invoke a route determination procedure on demand only. Thus, when a route is needed, some sort of global search procedure is employed. The family of classical flooding algorithms belongs to the reactive group. Proactive schemes have the advantage that when a route is needed, the delay before actual packets can be sent is very small. On the other side proactive schemes needs time to converge to a steady state. This can cause problems if the topology is changing frequently.

Chapter 3: Ad-hoc routing protocols

This chapter describes the different ad-hoc routing protocols that we have chosen to simulate and analyze.

3.1. Desirable properties

If the conventional routing protocols do not meet our demands, we need a new routing protocol. The question is what properties such protocols should have? These are some of the properties [10] that are desirable:

3.4.1. Distributed operation

The protocol should of course be distributed. It should not be dependent on a centralized controlling node. This is the case even for stationary networks. The difference is that nodes in an ad-hoc network can enter/leave the network very easily and because of mobility the network can be partitioned.

3.4.2. Loop free

To improve the overall performance, we want the routing protocol to guarantee that the routes supplied are loop-free. This avoids any waste of bandwidth or CPU consumption.

3.4.3. Demand based operation

To minimize the control overhead in the network and thus not wasting network resources more than necessary, the protocol should be reactive. This means that the protocol should only react when needed and that the protocol should not periodically broadcast control information.

3.4.4. Unidirectional link support

The radio environment can cause the formation of unidirectional links. Utilization of these links and not only the bi-directional links improves the routing protocol performance.

3.4.5. Security

The radio environment is especially vulnerable to impersonation attacks, so to ensure the wanted behavior from the routing protocol, we need some sort of preventive security measures. Authentication and encryption is probably the way to go and the problem lies within distributing keys among the nodes in the ad-hoc network. There are also discussions about using IP-sec [11] that uses tunneling to transport all packets.

3.4.6. Power conservation

The nodes in an ad-hoc network can be laptops and thin clients, such as PDAs that are very limited in battery power and therefore uses some sort of stand-by mode to save power. It is therefore important that the routing protocols have to support the power saver sleep modes.

3.4.7. Multiple routes

To reduce the number of reactions to topological changes and congestion, multiple routes could be used. If one route has become invalid, it is possible that another stored route could still be valid and thus saving the routing protocol from initiating another route discovery procedure.

3.4.8. Quality of service (QoS) support

Some sort of Quality of Service support is probably necessary to incorporate into the routing protocol. This has a lot to do with what these networks will be sued for. It could for instance be real-time traffic support.

None of the proposed protocols from MANET have all these properties, but it is necessary to remember that the protocols are still under development and are probably extended with more functionality. The primary function is still to find a route to the destination, not to find the best/ optimal/ shortest-path route.

The remainder of this chapter will describe the different routing protocols and analyze them theoretically.

3.2. MANETS

In situations where networks are constructed and destructed in ad-hoc manner, mobile Adhoc networking is an excellent choice. The idea of mobile ad-hoc or packet radio networks has been under development since 1970s. Since the mid-90s, when the definition of standards such as IEEE802.11 helped cause commercial wireless technology to emerge, mobile ad-hoc networking has been identified as a challenging evolution in wireless technology.

A MANET is an autonomous collection of mobile users communicating over a relatively bandwidth-constrained wireless link with limited battery power with highly dynamic environments [12]. The network topology, due to the mobility in the network, is dynamic and may change rapidly and unpredictably over time. Hence, the connectivity among the nodes may vary with time because of node departures, new node arrivals, and the possibility of having mobile nodes. To maintain communication between the nodes in the network, each node works as a transmitter, host, and, a router. The management and control functions are also distributed among the nodes.

3.3. Mobile Ad hoc Networks Communication Architecture: Protocol Stack

In this section the **protocol stack** for mobile ad hoc networks is described. This gives a comprehensive picture of, and helps to better understand, mobile ad hoc networks. Figure 1, shows the protocol stack which consists of five layers: physical layer, data link layer, network layer, transport layer and application layer. It has similarities to the TCP/IP protocol suite. As can be seen the OSI layers for session, presentation and application are merged into one section, the application layer.

On the left of figure 1, the OSI model is shown. It is a layered framework for the design of network systems that allows for communication across all types of computer systems. In the middle of the figure 1, the TCP/IP suite is illustrated. Because it was designed before the OSI model, the layers in the TCP/IP suite do not correspond exactly to the OSI layers. The lower four layers are the same but the fifth layer in the TCP/IP suite (the application layer) is equivalent to the combined session, presentation and application layers of the OSI model.

On the right, the MANET protocol stack -which is similar to the TCP/IP suite -is shown. The main difference between these two protocols stacks lies in the network layer. Mobile nodes (which are both hosts and routers) use an ad hoc routing protocol to route packets. In the physical and data link layer, mobile nodes run protocols that have been designed for wireless channels. Some options are the IEEE standard for wireless LANs, IEEE 802.11, the European ETSI standard for a high-speed wireless LAN, and finally an industry approach toward wireless personal area networks, i.e. wireless LANs at an even smaller range, Bluetooth. In the simulation tool used in this project, the standard IEEE 802.11 is used in these layers. [13]

OSI MODEL	TCP/IP SUITE	MANET PROTOCOL STACK	
APPLICATION	APPLICATION	APPLICATION	
PRESENTATION			
SESSION			
TRANSPORT	TRANSPORT	TRANSPORT	
NETWORK	NETWORK	NETWORK	ADHOC ROUTING
DATA LINK	DATA LINK	DATA LINK	
PHYSICAL	PHYSICAL	PHYSICAL	

Figure 1: Three Models of Protocol Stack

This thesis focuses on ad hoc routing which is handled by the network layer. The network layer is divided into two parts: Network and Ad Hoc Routing. The protocol used in the network part is Internet Protocol (IP) and the protocols which can be used in the ad hoc routing part are Dynamic Source Routing (DSR), Ad-hoc on Demand Distance Vector Routing (AODV).

3.4. Characteristics of MANETS

MANETs have several salient characteristics:

3.4.1. Dynamic topologies

Nodes are free to move arbitrarily; thus, the network topology--which is typically multihop--may change randomly and rapidly at unpredictable times, and may consist of both bidirectional and unidirectional links.

3.4.2. Bandwidth-constrained, variable capacity links

Wireless links will continue to have significantly lower capacity than their hardwired counterparts. In addition, the realized throughput of wireless communications—after accounting for the effects of multiple access, fading, noise, and interference conditions, etc.--is often much less than a radio's maximum transmission rate.

3.4.3. Energy-constrained operation

Some or all of the nodes in a MANET may rely on batteries or other exhaustible means for their energy. For these nodes, the most important system design criteria for optimization may be energy conservation.

3.4.4. Limited physical security

Mobile wireless networks are generally more prone to physical security threats than are fixed-cable nets. The increased possibility of eavesdropping, spoofing, and denial-of service attacks should be carefully considered.

These characteristics create a set of underlying assumptions and performance concerns for protocol design which extend beyond those guiding the design of routing within the higher-speed, semi-static topology of the fixed Internet.

3.5. Applications of MANETS

With the increase of portable devices as well as progress in wireless communication, ad hoc networking is gaining importance with the increasing number of widespread applications [3]. Ad hoc networking can be applied anywhere where there is little or no communication infrastructure or the existing infrastructure is expensive or inconvenient to use. Ad hoc networking allows the devices to maintain connections to the network as well as easily adding and removing devices to and from the network. The set of applications for MANETs is diverse, ranging from large-scale, mobile, highly dynamic networks, to small, static networks that are constrained by power sources. Besides the legacy applications that move from traditional infrastructure environment into the ad hoc context, a great deal of new services can and will be generated for the new environment. Typical applications include:

3.5.1. Military battlefield

The modern digital battlefield demands robust and reliable communication in many forms. Most communication devices are installed in mobile vehicles, tanks, trucks etc. Also soldiers could carry telecomm devices that could talk to a wireless base station or directly to other telecom devices if they are within the radio range. However these forms of communication are considered to be primitive. At times when wireless base station is destroyed by enemy, a soldier will be prohibited from communicating with other soldiers if the called party is not within the radio range. This is the scenario where mobile ad hoc networks come into play. Ad hoc networks are well known as self organizing networks since they are robust when nodes disappear due to destruction or mobility. Through multi-hop communication, soldiers can communicate to remote soldiers via data hoping and data forwarding from one radio device to another.

3.5.2. Sensor networks [14]

Another application of MANETs is sensor networks. This technology is a network composed of a very large number of small sensors. These can be used to detect any number of properties of an area. Examples include temperature, pressure, toxins, pollutions, etc. Applications are the measurement of ground humidity for agriculture, forecast of earthquakes. The capabilities of each sensor are very limited, and each must rely on others in order to forward data to a central computer. Individual sensors are limited in their computing capability and are prone to failure and loss. Mobile ad hoc sensor networks could be the key to future household security.

3.5.3. Automotive Applications

Automotive networks are widely discussed currently. Cars should be enabled to talk to the road, to traffic lights, and to each other, forming ad-hoc networks of various sizes. The network will provide the drivers with information about road conditions, congestions, and accident-ahead warnings, helping to optimize traffic flow.

3.5.4. Disaster Relief

Ad hoc can be used in emergency/rescue operations for disaster relief efforts, e.g. in fire, flood, or earthquake. Emergency rescue operations must take place where non-existing or damaged communications infrastructure and rapid deployment of a communication network is needed. Information is relayed from one rescue team member to another over small handheld devices. Other commercial scenarios include e.g. ship-to-ship ad hoc mobile communication, law enforcement, etc.

3.5.5. Personal Area Network

Personal Area Networks (PANs) are formed between various mobile (and immobile) devices mainly in an ad-hoc manner, e.g. for creating a home network. They can remain an autonomous network, interconnecting various devices, at home, for example, but PANs will become more meaningful when connected to a larger network. In this case PANs can be seen as an extension of the telecom network or Internet. Closely related to this is the concept of ubiquitous / pervasive computing where people, noticeable or transparently will be in close and dynamic interaction with devices in their surroundings.

3.6. MANETS challenges

The ad hoc networks have its own share of challenges which are listed below:

3.6.1. Spectrum allocation [15]

Issues such as interference, limited range, limited data throughput, device mobility and the sharing of the RF spectrum amongst devices all need addressing. Regulation regarding the use of radio spectrum is currently under the control of FCC. Most experimental Ad hoc networks are based on the ISM band. To prevent interference Ad hoc networks must operate over some form of allowed or specified spectrum range.

3.6.2. Routing

Routing of data is done between devices outside their RF range. The routing protocols used on wired networks do not perform well on networks involving mobility and rapid membership changes. More effective routing protocols are required. In Ad Hoc networks, we need new routing protocols because of the following reasons:

- Nodes in Ad Hoc networks are mobile and topology of interconnections between them may be quite dynamic.
- Existing protocols exhibit least desirable behavior when presented with a highly dynamic interconnection topology.
- Existing routing protocols place too heavy a computational burden on each mobile computer in terms of the memory-size, processing power and power consumption.
- Existing routing protocols are not designed for dynamic and self-starting behavior as required by users wishing to utilize Ad-Hoc networks.
- Existing routing protocols like Distance Vector Protocol take a lot of time for convergence upon the failure of a link, which is very frequent in Ad Hoc networks.
- Existing routing protocols suffer from looping problems either short lived or long lived.
- Methods adopted to solve looping problems in traditional routing protocols may not be applicable to Ad Hoc networks.

3.6.3. Existing IP Usage

For a mobile host to be able to communicate as it moves from one location to other, one of the following of the two things have to be in place:

- Mobile Hosts must changes its IP address whenever it moves to new place
- Host specific routes must be propagated throughout Internet Routing fabric.

There are problems with either of these options. If a host has an open TCP [16] session with another host, that session will be terminated if the IP address changes. Also, if other hosts must be able to initiate communication with a mobile host, how can they do so if their IP address changes every time they move? How does the host obtain a new IP address as it joins a network?

What is also of concern and it not addressed in this IETF draft [17] or in any publications is the convergence of two separate auto configured ad-hoc networks, merging together to form one larger ad-hoc network. Depending on the amount of participating hosts in each network and given the size of the address space given to link local addressing in IPv4[18] (65,563 possible hosts), there is a possibility of hosts having duplicate addresses. The main issue with using TCP in MANETs comes from the assumption that a packet being dropped is an indication of congestion occurring, not an indication of a lossy link or a data transmission error. This is due to the observation that that packet error/ loss rates over the internet due to transmission errors are of the order of 1%. However, in a wireless network, the amount of transmission errors is of a much higher order. The factors affecting the percentage of transmission errors include interference from other radio signals, device mobility, the sharing of a wireless link with other devices. All these can affect the delivery of TCP segments to the receiver, the timely return of ACK packets from the receiver and give variations in the RTT compared to the estimated value. Any of these occurring will result in the sender assuming that congestion is occurring and will use TCP's mechanisms to drastically reduce its transmission rate.

MANETs also provide additional challenges to TCP operation. The mobility of hosts means that routes between hosts are open to change. When a route is broken due to host mobility, a route reconstruction procedure is invoked. This reconstruction results in a delay that the TCP sender is unaware of. Overall data throughput has had to suffer initially because of the route reconstruction delay, but TCP has now further drastically decreased the data throughput on false pretences.

3.6.4. Security and Privacy

Following are the security and privacy challenges in the area of ad hoc networks:

- ❖ Firstly, use of wireless links renders an ad hoc network susceptible to link attacks ranging from passive eavesdropping to active impersonation, message replay, and message distortion. Eavesdropping might give an adversary access to secret information, violating confidentiality. Active attacks might allow the adversary to delete messages, to inject erroneous messages, to modify messages, and to impersonate a node, thus violating availability, integrity, authentication, and non-repudiation.

- Secondly, nodes, roaming in a hostile environment (e.g., a battlefield) with relatively poor physical protection, have non-negligible probability of being compromised. Therefore, we should not only consider malicious attacks from outside a network, but also take into account the attacks launched from within the network by compromised nodes. Therefore, to achieve high survivability, ad hoc networks should have a distributed architecture with no central entities. Introducing any central entity into our security solution could lead to significant vulnerability; that is, if this centralized entity is compromised, then the entire network is subverted.

- Thirdly, an ad hoc network is dynamic because of frequent changes in both its topology and its membership (i.e., nodes frequently join and leave the network). Trust relationship among nodes also changes, for example, when certain nodes are detected as being compromised. Unlike other wireless mobile networks, such as mobile IP, nodes in an ad hoc network may dynamically become affiliated with administrative domains. Any security solution with a static configuration would not suffice. It is desirable for our security mechanisms to adapt on-the-fly to these changes.

- Finally, an ad hoc network may consist of hundreds or even thousands of nodes. Security mechanisms should be scalable to handle such a large network.

3.7. MANET Protocols

IETF has a working group named MANET (Mobile Ad-hoc Networks) [19] that is working in the field of ad-hoc networks. They have developed routing specifications for ad-hoc IP networks that support scaling to a couple of hundred nodes.

IETF currently have seven routing protocol drafts:

- AODV – Ad hoc On Demand Distance Vector [2]
- ZRP- Zone Routing Protocol [20]
- TORA/IMEP- Temporally Ordered Routing Algorithm/ Internet MANET Encapsulation Protocol [6] [21] [22]
- DSR- Dynamic Source Routing [7] [23]
- CBRP- Cluster Based Routing Protocol [24]
- CEDAR- Core Extraction Distributed Ad hoc Routing [25]

- AMRoute- Ad hoc Multicast Routing Protocol [26]
- OLSR – Optimized Link State Routing Protocol [27]

These proposed protocols we have chosen to analyze AODV, DSR, ZRP, CBRP and TORA theoretically. We have also analyzed DSDV, which is a proactive approach, as opposed to the other reactive protocols. We have not analyzed AMRoute because it is a multicast routing protocol, neither CEDAR because it is primarily a QoS routing protocol, nor OSLR, because it was submitted as an Internet draft so late. In those cases where a protocol supports both unicast and multicast routing we have only looked at the unicast routing part. Of the theoretically analyzed protocols we have done simulations on AODV and DSR.

3.7.1. Destination Sequenced Distance Vector routing- DSDV

3.7.1.1. Description

DSDV [28] is a hop-by-hop distance vector routing protocol that in each node has a routing table that for all reachable destinations stores the next-hop and number of hops for that destination. Like distance-vector, DSDV requires that each node periodically broadcast routing updates. The advantage with DSDV over traditional distance vector protocol is that DSDV guarantees loop freedom.

To guarantee loop-freedom DSDV uses a sequence numbers to tag each route. The sequence number shows the freshness of a route and routes with higher sequence numbers are favorable. A route R is considered more favorable than R' if R has a greater sequence number or, if the routes have the same sequence number but R has lower hop-count. The sequence number is increased when a node A detects that a route to a destination D has broken. So that next time node A advertises its routes, it will advertise the route to D with an infinite hop-count and a sequence number that is larger than before.

DSDV basically is a distance vector with small adjustment to make it better suited for ad-hoc networks. These adjustments consist of triggered updates that will take care of topology changes in the time between broadcasts. To reduce the amount of information in these packets there are two types of update messages defined: full and incremental dump. The full dump carries all available routing information and the incremental dump that only carries the information that has changes since the last dump.

3.7.1.2. Properties

Because DSDV is dependent on periodic broadcasts it need some time to converge before a route can be used. This converge time can probably be considered negligible in a static wired network, where the topology is not changing so frequently. In an ad-hoc network on the other hand, where the topology is expected to be very dynamic, this converge time will probably mean a lot of dropped packets before a valid route is detected. The periodic broadcasts also add a large amount of overhead into the network.

3.7.2. Ad-hoc On Demand Distance vector – AODV

3.7.2.1. Description

AODV is capable of both unicast and multicast routing. AODV uses sequence numbers to ensure the freshness of routes. It is self starting, loop-free, and scales to large numbers of mobile nodes. AODV builds routes using a route request / route reply query cycle. When a source node desires a route to a destination for which it does not already have a route, it broadcasts a *Route Request* (*RREQ*) packet across the network. Nodes receiving this packet update their information for the source node and set up backwards pointers to the source node in the route tables. In addition to the source node's IP address, current sequence number, and broadcast *ID*, the *RREQ* also contains the most recent sequence number for the destination of which the source node is aware. A node receiving the *RREQ* may send a *Route Reply* (*RREP*) if it is either the destination or if it has a route to the destination with corresponding sequence number greater than or equal to that contained in the *RREQ*. If this is the case, it unicasts a *RREP* back to the source. Otherwise, it rebroadcasts the *RREQ*. Nodes keep track of the *RREQ*'s source IP address and broadcast *ID*. If they receive a *RREQ* that they have already processed, they discard the *RREQ* and do not forward it.

As the *RREP* propagates back to the source, nodes set up forward pointers to the destination. Once the source node receives the *RREP*, it may begin to forward data packets to the destination. If the source later receives a *RREP* containing a greater sequence number or contains the same sequence number with a smaller hop count, it may update its routing information for that destination and begin using the better route.

The route is maintained as long as there are data packets periodically being sent from the source to the destination along that path. Once the source stops sending data packets, the links will time out and eventually be deleted from the intermediate node routing tables. If a link break occurs while the route is being used for transmission of data, the node upstream of the break propagates a *Route Error* (*RERR*) message to the source node to inform it of the now unreachable destination(s). After receiving the *RERR*, if the source node still desires the route, it can reinitiate route discovery [2].

3.7.2.2. Message Summary

Ad hoc On-demand Distance Vector (*AODV*) routing protocol is composed of a *Hello*, a *Route Request* (*RREQ*), a *Route Reply* (*RREP*), a *Route Error* (*RERR*), and a *Route Reply Acknowledgement* (*RREP-ACK*) message. To reduce the data packet overhead, *AODV* nodes store routing information in the node routing table instead of using source route added into the data packet such as in *DSR*. *AODV* nodes check the link status of next hops in active routes for the route maintenance. If a link break is detected, the node which finds the link break sends a *RERR* message to notify other nodes that the link was broken.

3.7.2.3. Route Request message

An *AODV* node increases the sequence number by one whenever the node triggers an action to do *RREQ*, *RREP*, *RERR*, or *RREP-ACK*. It is assumed that when a source node finds a route towards a destination, previous valid route to the destination is expired in its routing table, and it does not have a valid route to the destination. The destination sequence number in *RREQ* message is copied from the source's routing table which stores the last known destination sequence number. If a source node generating a *RREQ* message does not know the sequence number of the destination, it sets the *U* bit
(*unknown sequence number*) in its *RREQ* message. The field of *RREQ ID* is increased by one from the last *RREQ ID* issued by the source node. When the source node receives the *RREQ* message again from its neighbors due to the local broadcasting property of MANET, it just discards the *RREQ* message after comparing the *RREQ ID* and originator IP address in the received *RREQ* message and the *RREQ ID* and originator IP address stored in the source node's *Path_Discovery_Time* buffer.

To make the bidirectional communication between a source node and a destination node, the source node must not only know the route to the destination, but also the destination node should know a route back to the source node. As one of *RREQ* propagation scenarios, one of the intermediate nodes having a valid route to the destination can reply by sending a *RREP*, therefore, the destination node does not receive the *RREQ* message from the source node and cannot make a route back to the source node. To compensate this situation, the source node sends the *RREQ* message including the bit of *G* (*gratuitous RREP flag*) that is set, which notifies the intermediate node generating the *RREP* message to unicast a gratuitous *RREP* to the destined destination node. To find out a destination, a source node uses an expanding ring search algorithm to avoid the network-wide dissemination of *RREQ*s, which can be implemented by the *TTL* value in the *RREQ* IP header. If a source node does not receive a *RREP* message, it resends a *RREQ* message with the *TTL* increased by *TTL_Increment*. This will continue until the value of *TTL* in the *RREQ* reaches *TTL_Threshold*. The first-in, first-out (*FIFO*) scheme is used to buffer the data packet which waits for a *RREP* after a *RREQ* has been sent.

A source node should wait for a *RREP* message by using a binary exponential backoff mechanism to reduce network congestion. If the source node does not receive a *RREP* message within the *Net_Traveral_Time* after sending a *RREQ* message, the source node resends the second *RREQ* message. In this case, the source node should wait for the *RREP* message for a duration of 2**Net_Traveral_Time* which is two times longer than the first *Net_Traveral_Time*. If the source node does not receive the *RREP* message, it can resend the third *RREQ* message up to *RREQ_Retries*. For this case, the new waiting time for the source node is calculated by multiplying 2 into the previous waiting time, which is 4* *Net_Traveral_Time*. When an intermediate node receives a *RREQ* message, it checks whether it receives a *RREQ* with the same *Originator* IP Address and *RREQ ID*. It discards the *RREQ* if it receives such a *RREQ*. If it did not receive such a *RREQ* before, it first increases the value of the hop count field in the received *RREQ* by one, then it search a reverse route to the *Originator* IP Address. If it finds a reverse route to the *Originator* IP Address, the sequence number of the route in its routing table is copied from the value of the current sequence number of the *RREQ*. When the intermediate node receives a *RREP* message for the response of the *RREQ* message, it should have a reverse route to send the received *RREP* message towards the *Originator* IP Address. To carry out the refresh mechanism of the reverse route, when an intermediate node receives a *RREQ* message, it sets the value of the lifetime of the reverse route entry for the *Originator* IP Address as the maximum of (*ExistingLifeTime*, *MinimalLifeTime*).

0 1 2 3

0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1

Type	*J*	*R*	*G*	*D*	*U*	*Reserved*	*Hop Count*
RREQ ID							
Destination IP Address							
Destination Sequence Number							
Originator IP Address							
Originator Sequence Number							

Figure 2: AODV Route Request Message

The format of the Route Request message illustrated above contains the following fields:

Type 1

J Join flag; reserved for multicast.

R Repair flag; reserved for multicast.

G Gratuitous RREP flag; indicates whether a gratuitous RREP should be unicast to the node specified in the Destination IP Address field

D Destination only flag; indicates only the destination may respond to this RREQ

U Unknown sequence number; indicates the destination sequence number is unknown

Reserved Sent as 0; ignored on reception.

Hop Count The number of hops from the Originator IP Address to the node handling the request.

RREQ ID A sequence number uniquely identifying the particular RREQ when taken in conjunction with the originating node's IP address.

Destination IP Address
The IP address of the destination for which a route is desired.

Destination Sequence Number
The latest sequence number received in the past by the originator for any route towards the destination.

Originator IP Address

The IP address of the node which originated the Route Request.

Originator Sequence Number

The current sequence number to be used in the route entry pointing towards the originator of the route request.

3.7.2.4. Route Reply message

When a destination receives a *RREQ* message or an intermediate node which has an active route to a destination, it should respond by sending a *RREP* message towards the source node. The following section describes the case that the destination receives the *RREQ*. The fields of *Destination IP Address* and the *Originator IP Address* of the *RREP* message come from the corresponding fields in the *RREQ* message. The unicast route for the *RREP* towards the source follows the reverse path from which the *RREQ* is delivered. The value of the *Hop Count* field is increased by one at each intermediate node towards the source. The value of *My_Route_Timeout* in the destination is copied into the value of *Lifetime* field. When the intermediate node sends the *RREP* message, the field value for the *Destination Sequence Number* comes from the value of its *destination sequence number* which can be extracted from its routing table. The value of the *Hop Count* field is calculated from the distance in hops from the destination to the intermediate node. The value obtained from subtracting the current time from the expiration time in the route table entry can be the value of the *Lifetime* field. The *forward route entry* in the precursor list is copied from the source IP address of the received *RREQ*, which indicates the last hop node from which the intermediate node receives the *RREQ*. The *reverse route entry* in the precursor list is copied from the next hop towards the destination stored in its routing table.

0 1 2 3

0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1

Type	*R*	*A*	*Reserved*	*Prefix Sz*	*Hop Count*
Destination IP Address					
Destination Sequence Number					
Originator IP Address					
Lifetime					

Figure 3: AODV Route Reply Message

The format of the Route Reply message illustrated above contains the following fields:

Type 2

R Repair flag; used for multicast.

A Acknowledgment required

Reserved Sent as 0; ignored on reception.

Prefix Size If nonzero, the 5-bit Prefix Size specifies that the indicated next hop may be used for any nodes with the same routing prefix (as defined by the Prefix Size) as the requested destination.

Hop Count The number of hops from the Originator IP Address to the Destination IP address. For multicast route requests this indicates the number of hops to the multicast tree member sending the RREP.

Destination IP Address
The IP address of the destination for which a route is supplied.

Destination Sequence Number
The destination sequence number associated to the route.

Originator IP Address
The IP address of the node which originated the RREQ for which the route is supplied.

Lifetime The time in milliseconds for which nodes receiving the RREP consider the route to be valid.

3.7.2.5. Route Error message

Whenever a node detects a failure of next link in an active route, it should generate a *RERR* message which is broadcasted if there are many precursors or unicasted if there is only one precursor. The fields of the *Unreachable Destination IP Address* and *Unreachable Destination Sequence Number* are the *Destination IP Address* and *Destination Sequence Number* in an active route where the next link of this node is broken to its neighbor. The node runs a local recovery and it does not receive any *RREP* message within a recovery period; it informs a source node of the link failure destined for the destination by sending a *RERR* message.

0 1 2 3

0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1

Type	*N*	*Reserved*	*DestCount*
Unreachable Destination IP Address (1)			
Unreachable Destination Sequence Number (1)			
Additional Unreachable Destination IP Address (if needed)			
Additional Unreachable Destination Sequence Number (if needed)			

Figure 4: AODV Route Error message

The format of the Route Error message is illustrated contains the following fields:

Type 3

N No delete flag; set when a node has performed a local repair of a link, and upstream nodes should not delete the route.

Reserved Sent as 0; ignored on reception.

DestCount The number of unreachable destinations included in the message; MUST be at least 1.

Unreachable Destination IP Address
The IP address of the destination that has become unreachable due to a link break.

Unreachable Destination Sequence Number
The sequence number in the route table entry for the destination listed in the previous Unreachable Destination IP Address field.

3.7.2.6. Route Reply Acknowledgment message

Whenever the node receives a *RREP* message which sets the bit of '*A*' field, it should respond with *Route Reply Acknowledgement* message.

0 1

0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5

Type	*Reserved*

Figure 5: AODV Route Reply Acknowledgment message

The format of the Route Reply Acknowledgment message is illustrated contains the following fields:

Type	4
Reserved	Sent as 0; ignored on reception

3.7.2.7. Hello message

An *AODV* node, which is one of the members on an active route, can use either an appropriate layer 2 message or a *RREQ* with *TTL* value of one in the *RREQ* IP header, which can be used as a *Hello* message. *AODV* nodes should send a *RREQ* or an appropriate layer 2 message for every *Hello_Interval* to refresh the active link to its neighbor node. If it does not send any *RREQ* message within the last *Hello_Interval*, it should broadcast a *RREP* with *TTL* value of one with the following modifications, which is called as an *AODV Hello* message. The field value of *Destination IP Address* in *RREP* message is the node's IP address. The field value of *Destination Sequence Number* is the node's latest sequence number. The field value of *Hop Count* is zero. The field value of *Lifetime* is *Allowed_Hello_Loss * Hello_Interval.* If a node does not receive any *AODV Hello* or other messages from its neighbors within *Allowed_Hello_Loss * Hello Interval*, it can decide the link to its neighbor is broken.

3.7.2.8. Pseudocode

Action taken at a node that desires to forward data Packets

```
1. Check if there is route to the node of interest
2. If yes, send the message
3. If no, Broadcast a RREQ Packet through the Network
4. And wait for a RREP to forward the data packets
5. Upon receiving the RREP forward the data packets to the destination
6. If another RREP is received with a greater sequence number or same sequence number with a
   smaller hop count, start using this better route
```

Action taken at a node on receiving a RREQ Packet

1. *Update the information for the source node*
2. *Set up backwards pointers to the source node in the route tables*
3. *Check if the current node is the destination or it has a route to the destination with corresponding sequence number greater than or equal to that contained in the RREQ*
4. *If yes, send a unicast RREP back to the source and set the forward pointers to the source.*
5. *If no, rebroadcast the RREQ.*
6. *Keep track of the RREQ's source IP address and broadcast ID.*
7. *If the received RREQ has been already processed, discard the RREQ and do not forward it.*

3.7.3. Dynamic Source Routing –DSR

3.7.3.1. Description

To send a packet to another host, the sender constructs a *source route* in the packet's header, giving the address of each host in the network through which the packet should be forwarded in order to reach the destination host. The sender then transmits the packet over its wireless network interface to the first hop identified in the source route. When a host receives a packet, if this host is not the final destination of the packet, it simply transmits the packet to the next hop identified in the source route in the packet's header.

Once the packet reaches its final destination, the packet is delivered to the network layer software on that host. There are two basic operations that take place in *DSR* – namely, route discovery and route maintenance [7], [29].

3.7.3.2. Message Summary

DSR protocol is composed of the mechanisms of *Route Discovery* and *Route Maintenance* which operate totally on-demand. When a source node (*S*) wants to send a packet towards a destination (*D*), *S* finds out an explicit source route to follow on its way to *D* in its *Route Cache*. When a route out of the *Route Cache* provides the route to *D*, *S* can use the route to send packets if the *Route Cache* has the valid route. If *S* cannot find the route in its *Route Cache*, *S* initiates the *Route Discovery* mechanism via a *DSR Route Request* (*RREQ*) and a *DSR Route Reply* (*RREP*) options. *S* propagates the *RREQ* option that includes the destination address to its neighbors. When intermediate nodes propagate the *RREQ* option, they record their addresses to the *Address* fields of the *RREQ* option. When *D* receives the *RREQ* option, it should reply with the *DSR RREP* option that includes a copy of the accumulated route record list from the *RREQ* option. When *S* receives the *RREP* option, *S* stores the route record list in its *Route Cache*, puts the source route into the header of the packets, and sends the packet that has the routing information in their headers [7], [29].

3.7.3.3. RREQ message

When *S* cannot find the route in its *Route Cache*, *S* initiates the *Route Discovery* mechanism via a *Route Request* (*RREQ*) message. The *Target Address* field indicates the IP address of a destination. *Address* [1], *Address* [2], …, *Address* [*n*] fields are accumulated when a *RREQ* is relayed at one of the relaying nodes between *S* and *D*.

0 1 2 3

0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1

Option Type	*Opt Data Len*	*Identification*
Target Address		
Address [1]		
Address [2]		
…		
Address [n]		

Figure 6: DSR Route Request message

The format of the Route Request message is illustrated contains the following fields:

IP fields:

Source Address

MUST be set to the address of the node originating this packet. Intermediate nodes that retransmit the packet to propagate the Route Request MUST NOT change this field.

Destination Address

MUST be set to the IP limited broadcast address (255.255.255.255).

Hop Limit (TTL)

MAY be varied from 1 to 255, for example, to implement non- propagating Route Requests and Route Request expanding-ring searches

Route Request fields:

Option Type

1. Nodes not understanding this option will ignore this option.

Opt Data Len

8-bit unsigned integer. Length of the option, in octets, excluding the Option Type and Opt Data Len fields. MUST be set equal to (4 * n) + 6, where n is the number of addresses in the Route Request Option.

Identification

A unique value generated by the initiator (original sender) of the Route Request. Nodes initiating a Route Request generate a new Identification value for each Route Request, for example based on a sequence number counter of all Route Requests initiated by the node.

This value allows a receiving node to determine whether it has recently seen a copy of this Route Request. If this Identification value (for this IP Source address and Target Address) is found by this receiving node in its Route Request Table (in the cache of Identification values in the entry there for this initiating node), this receiving node MUST discard the Route Request. When a Route Request is propagated, this field MUST be copied from the received copy of the Route Request being propagated.

Target Address

The address of the node that is the target of the Route Request.

Address[1..n]

Address[i] is the IPv4 address of the i-th node recorded in the Route Request option. The address given in the Source Address field in the IP header is the address of the initiator of the Route Discovery and MUST NOT be listed in the Address[i] fields; the address given in Address[1] is thus the IPv4 address of the first node on the path after the initiator. The number of addresses present in this field is indicated by the Opt Data Len field in the option (n = (Opt Data Len - 6) / 4). Each node propagating the Route Request adds its own address to this list, increasing the Opt Data Len value by 4 octets.

The Route Request option MUST NOT appear more than once within a DSR Options header.

3.7.3.4. RREP message

When *D* receives the *RREQ* option, it should reply with the *DSR RREP* option that includes a copy of the accumulated route record list from the *RREQ* option. *S* uses *Address* [1]*, Address* [2]*, …, Address* [*n*] fields as a source route in order to send data packet to *D*.

0 1 2 3

0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1

<table>
<tr><td></td><td><i>Option Type</i></td><td><i>Opt Data Len</i></td><td><i>L</i></td><td><i>Reserved</i></td></tr>
<tr><td colspan="5"><i>Address</i> [1]</td></tr>
<tr><td colspan="5"><i>Address</i> [2]</td></tr>
<tr><td colspan="5">…</td></tr>
<tr><td colspan="5"><i>Address</i> [n]</td></tr>
</table>

Figure 7: DSR Route Reply message

The format of the Route Reply message is illustrated contains the following fields:

IP fields:

Source Address

Set to the address of the node sending the Route Reply. In the case of a node sending a reply from its Route Cache or sending a gratuitous Route Reply, this address can differ from the address that was the target of the Route Discovery.

Destination Address

MUST be set to the address of the source node of the route being returned. Copied from the Source Address field of the Route Request generating the Route Reply or, in the case of a gratuitous Route Reply, copied from the Source Address field of the data packet triggering the gratuitous Reply.

Route Reply fields:

Option Type

2. Nodes not understanding this option will ignore this option.

3.7.3.5. RERR message

When a node finds a link error while it attempts to forward a packet, it generates a *RERR* option. *Error Source Address* field indicates a node which generates a *RERR* message. *Error Destination Address* field indicates a node to which a *RERR* message should be delivered. *Type-Specific Information* includes the detail of error contents.

0 1 2 3

0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1

Option Type	*Opt Data Len*	*Error Type*	*Reserved*	*Salvage*
Error Source Address				
Error Destination Address				
Type-Specific Information				

Figure 8: DSR Route Error message

The format of the Route error message is illustrated contains the following fields:

Option Type

2 Nodes not understanding this option will ignore this option.

Opt Data Len

8-bit unsigned integer. Length of the option, in octets, excluding the Option Type and Opt Data Len fields.

For the current definition of the Route Error option, this field MUST be set to 10, plus the size of any Type-Specific Information present in the Route Error. Further extensions to the Route Error option format may also be included after the Type-Specific Information portion of the Route Error option specified above. The presence of such extensions will be indicated by the Opt Data Len field. When the Opt Data Len is greater than that required for the fixed portion of the Route Error plus the necessary Type-Specific Information as indicated by the Option Type value in the option, the remaining octets are interpreted as extensions. Currently, no such further extensions have been defined.

Error Type

The type of error encountered. Currently, the following type values are defined:

1 = NODE_UNREACHABLE
2 = FLOW_STATE_NOT_SUPPORTED

3 = OPTION_NOT_SUPPORTED

Other values of the Error Type field are reserved for future use.

Reservd

Reserved. MUST be sent as 0 and ignored on reception.

Salvage

A 4-bit unsigned integer. Copied from the Salvage field in the DSR Source Route option of the packet triggering the Route Error.

The "total salvage count" of the Route Error option is derived from the value in the Salvage field of this Route Error option and all preceding Route Error options in the packet as follows:
the total salvage count is the sum of, for each such Route Error option, one plus the value in the Salvage field of that Route Error option.

Error Source Address

The address of the node originating the Route Error (e.g., the node that attempted to forward a packet and discovered the link failure).

Error Destination Address

The address of the node to which the Route Error must be delivered. For example, when the Error Type field is set to NODE_UNREACHABLE, this field will be set to the address of the node that generated the routing information claiming that the hop from the Error Source Address to Unreachable Node Address (specified in the Type-Specific Information) was a valid hop.

Type-Specific Information

Information specific to the Error Type of this Route Error message.

A Route Error option MAY appear one or more times within a DSR Options header.

3.7.3.6. DSR Source Route message

Data packet includes a DSR Source Route option. Each active node forwards the data packet based on the DSR Source Route option.

0 1 2 3

0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1

<table>
<tr><td><em>Option Type</em></td><td><em>Opt Data Len</em></td><td><em>F</em></td><td><em>L</em></td><td><em>Reserved</em></td><td><em>Salvage</em></td><td><em>Segs Left</em></td></tr>
<tr><td colspan="7"><em>Address</em> [1]</td></tr>
<tr><td colspan="7"><em>Address</em> [2]</td></tr>
<tr><td colspan="7">…</td></tr>
<tr><td colspan="7"><em>Address</em> [n]</td></tr>
</table>

Figure 9: DSR Source Route message

The format of the Route error message is illustrated contains the following fields:

Option Type

> 96 Nodes not understanding this option will drop the packet.

Opt Data Len

> 8-bit unsigned integer. Length of the option, in octets, excluding the Option Type and Opt Data Len fields. For the format of the DSR Source Route option defined here, this field MUST be set to the value (n * 4) + 2, where n is the number of addresses present in the Address[i] fields.

First Hop External (F)

> Set to indicate that the first hop indicated by the DSR Source Route option is actually an arbitrary path in a network external to the DSR network; the exact route outside the DSR network is not represented in the DSR Source Route option. Nodes caching this hop in their Route Cache MUST flag the cached hop with the External flag. Such hops MUST NOT be returned in a Route Reply generated from this Route Cache entry, and selection of routes from the Route Cache to route a packet being sent SHOULD prefer routes that contain no hops flagged as External.

Last Hop External (L)

> Set to indicate that the last hop indicated by the DSR Source Route option is actually an arbitrary path in a network external to the DSR network; the exact route outside the DSR network is not represented in the DSR Source Route option. Nodes caching this hop in their Route Cache MUST flag the cached hop with the External flag. Such hops MUST NOT be returned in a Route Reply generated from this Route Cache entry, and selection of routes

from the Route Cache to route a packet being sent SHOULD prefer routes that contain no hops flagged as External.

Reserved MUST be sent as 0 and ignored on reception.

Salvage

A 4-bit unsigned integer. Count of number of times that this packet has been salvaged as a part of DSR routing.

Segments Left (Segs Left)

Number of route segments remaining, i.e., number of explicitly listed intermediate nodes still to be visited before reaching the final destination.

Address[1..n]

The sequence of addresses of the source route. In routing and forwarding the packet, the source route is processed as described in Sections 8.1.3 and 8.1.5. The number of addresses present in the Address[1..n] field is indicated by the Opt Data Len field in the option (n = (Opt Data Len - 2) / 4).

3.7.3.7. Algorithm for DSR

3.7.3.7.1. Originating a Data Packet

When node A originates a packet, the following steps must be taken before transmitting the packet:

STEPS:

1. If the destination address is a multicast address, piggyback the data packet on a Route Request targeting the multicast address. The following fields must be initialized as specified:

IP.Source_Address = Home address of node A

IP.Destination_Address = 255.255.255.255

Request.Target_Address = Multicast destination address

2. Otherwise, call Route_Cache.Get() to determine if there is a cached source route to the destination.

3. If the cached route indicates that the destination is directly reachable over one hop, no Routing Header should be added to the packet. Initialize the following fields:

IP.Source_Address = Home address of node A

IP.Destination_Address = Home address of the Destination

4. Otherwise, if the cached route indicates that multiple hops are required to reach the destination, inserts a Routing Header into the packet.

5. Otherwise, if no cached route to the destination is found, insert the packet into the Send Buffer and initiate Route Discovery

3.7.3.7.2. Processing a Route Request Option

When a node A receives a packet containing a Route Request option, the Route Request option is processed as follows:

STEPS:

1. If Request.Target_Address matches the home address of this node, then the Route Request option contains a complete source route describing the path from the initiator of the Route Request to this node.

(a) Send a Route Reply.

(b) Continue processing the packet in accordance with the Next Header value contained in the Destination Option extension header.

2. Otherwise, if the combination (IP.Source_Address, Request.Identification) is found in the Route Request Table, then discard the packet, since this is a copy of a recently seen Route Request.

3. Otherwise, if Request.Target_Address is a multicast address then:

(a) If node A is a member of the multicast group indicated by Request.Target_Address, then create a copy of the packet, setting IP.Destination_Address = REQUEST.Target_Address, and continue processing the copy of the packet in accordance with the Next Header field of the Destination option.

(b) If IP.TTL is non-zero, decrement IP.TTL, and retransmit the packet.

(c) Otherwise, discard the packet.

4. Otherwise, if the home address of node A is already listed in the Route Request (IP.Source_Address or Request.Address[ ]), then discard the packet.

5. Let

m = number of addresses currently in the Route Request option

n = m + 1

6. Otherwise, append the home address of node A to the Route Request option (Request.Address[n]).

7. Set Request.IN_Index[n] = index of interface packet was received on.
8. If a source route to Request.Target_Address is found in our Route Cache, return a Cached Route Reply
9. Otherwise, for each interface on which the node is configured to participate in a DSR ad hoc network:

(a) Make a copy of the packet containing the Route Request.

(b) Set Request.OUT_Index[n+1] = index of the interface.

(c) If the outgoing interface is different from the incoming interface, then set the C bit on both Request.OUT_Index[n+1] and Request.IN_Index[n]

(d) Link-layer re-broadcasts the packet containing the Route Request on the interface jittered by T milliseconds, where T is a uniformly distributed, random number between 0 and BROADCAST_JITTER.

3.6.1.1.1 Originating a Route Reply

STEPS:

1. If REQPacket.Request.Address[ ] does not contain any hops, then node A is only a single hop from the originator of the Route Request. Build a Route Reply packet as follows:
   REPPacket.IP.Source_Address = REQPacket.Request.Target_Address
   REPPacket.Reply.Target = REQPacket.IP.Source_Address
   REPPacket.Reply.OUT_Index[1] = REQPacket.Request.OUT_index[1]
   REPPacket.Reply.OUT_C_bit[1] = REQPacket.Request.OUT_C_bit[1]
   REPPacket.Reply.Address[1] = The home address of node A
   GOTO step 3.
2. Otherwise, build a Route Reply packet as follows:
   REPPacket.IP.Source_Address = The home address of node A
   REPPacket.Reply.Target = REQPacket.IP.Source_Address
   REPPacket.Reply.OUT_Index[1..n] = REQPacket.Request.OUT_index[1..n]
   REPPacket.Reply.OUT_C_bit[1..n] = REQPacket.Request.OUT_C_bit[1..n]
   REPPacket.Reply.Address[1..n] = REQPacket.Request.Address[1..n]
3. Send the Route Reply jittered by T milliseconds, where T is a uniformly distributed random number between 0 and BROADCAST_JITTER [7], [29].

3.7.4. Zone Routing Protocol- ZRP

3.7.4.1. Description

Zone Routing Protocol (ZRP) [20] is a hybrid of a reactive and a proactive protocol. It divides the network into several routing zones and specifies two totally detached protocols that operate inside and between the routing zones.

The Intrazone Routing Protocol (IARP) operates inside the routing zone and learns the minimum distance and routes to all the nodes within the zone. The protocol is not defined and can include any number of proactive protocols, such as Distance Vector or link-state routing. Different zones may operate with different intrazone protocols as long as the protocols are restricted to those zones. A change in topology means that update information only propagates within the affected routing zones as opposed to affecting the entire network.

The second protocol, the Interzone Routing Protocol (IERP) is reactive and is used for finding routes between different routing zones. This is useful if the destination node does not lie within the routing zone. The protocol then broadcasts (i.e. bordercasts) a Route REQuest (RREQ) to all border nodes within the routing zone, which in turn forwards the request if the destination node is not found within their routing zone. This procedure is repeated until the requested node is found and a route reply is sent back to the source indicating the route. IERP uses a Bordercast Resolution Protocol (BRP) [20] that is included in ZRP. BRP provides bordercasting services, which do not exist in IP. Bordercasting is the process of sending IP datagrams from one node to all its peripheral nodes. BRP keeps track of the peripheral nodes and resolves a border cast address to the individual IP-addresses of the peripheral nodes. The message that was bordercasted is then encapsulated into a BRP packet and sent to each peripheral node.

3.7.4.2. Routing Zone

A routing zone is defined as a set of nodes, within a specific minimum distance in number of hops from the node in question. The distance is referred to as the zone radius. In the example network (Figure 10), node S, A, F, B, C, G and H, all lie within a radius of two from node F. Even though node B also has a distance of 3 hops from node F, it is included in the zone since the shortest distance is only 2 hops. Border nodes or peripheral nodes are nodes whose minimum distance to the node in question is equal exactly to the zone radius. In Figure 10, nodes B and F are border nodes to S.

Consider the network in Figure 10. Node S wants to send a packet to node D. Since node D is not in the routing zone of S, a route request is sent to the border nodes B and F. Each border node checks to see if D is in their routing zone. Neither B nor F finds the requested node in their routing zone; thus the request is forwarded to the respectively border nodes. F sends the request to S, B, C and H while B sends the request to S, F, E and G. Now the requested node D is found within the routing zone of both C and E thus a reply is generated and sent back towards the source node S.

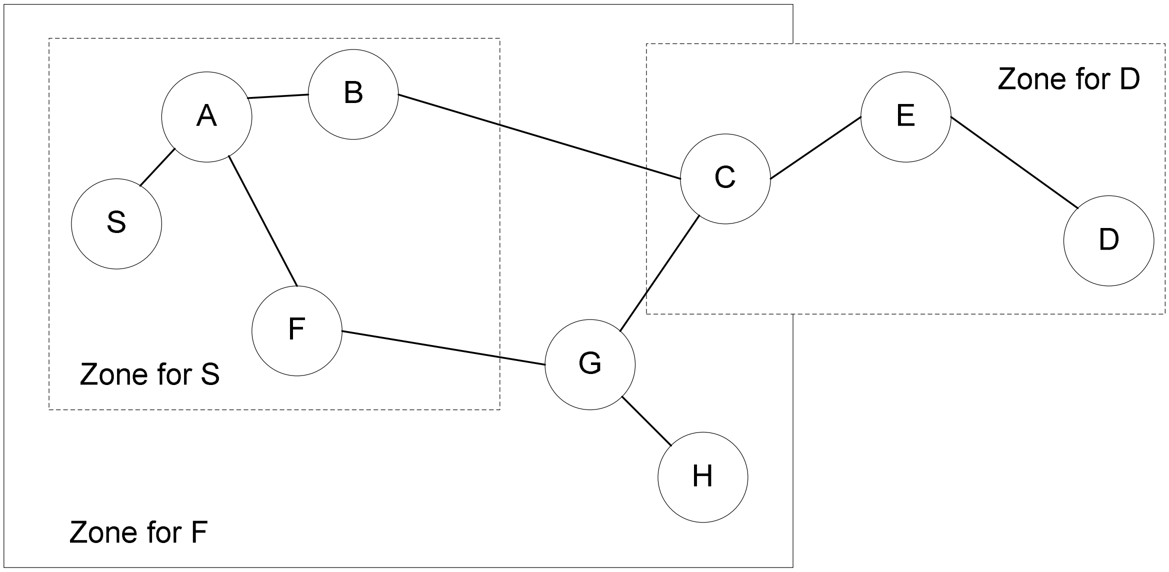


Figure 10: Network using ZRP

The dashed squares show the routing zones for nodes S and D. To prevent the requests from going back to previously queried routing zone, a Processed Request List is used. This list stores reviously processed requests and if a node receives a request that it already has processed, it is simply dropped.

3.7.4.3. Properties

ZRP is a very interesting protocol and can be adjusted of its operation to the current network operational conditions (e.g. change the routing zone diameter). However this is not done dynamically, but instead it is suggested that this zone radius should be set by the administration of the network or with a default value by the manufacturer. The performance of this protocol depends quite a lot on this decision.

Since this is a hybrid between proactive and reactive schemes, this protocol use advantages from both. Routes can be found very fast within the routing zone, while routes outside the zone can be found by efficiently querying selected nodes in the network. One problem is however that the proactive intrazone routing protocol is not specified. The use of different intrazone routing protocols would mean that the nodes would have to support several different routing protocols. This is not a good idea when dealing with thin clients. It is better to use the same intrazone routing protocol in the entire network.

ZRP also limits propagation of information about topological changes to the neighborhood of the change only (as opposed to a fully proactive scheme, which would basically flood the entire network when a change in topology occurred). However, a change in topology can affect several routing zones.

3.7.5. Temporally-Ordered Routing Algorithm – TORA

3.7.5.1. Description

Temporally Ordered Routing Algorithm (TORA) [20][21] is a distributed routing protocol. The basic underlying algorithm is one in a family referred to as link reversal algorithms. TORA is designed to minimize reaction to topological changes. A key concept in its design is that control messages are typically localized to a very small set of nodes. It guarantees that all routes are loop-free (temporary loops may form), and typically provides multiple routes for any source/destination pair. It provides only the routing mechanism and depends on Internet MANET Encapsulation Protocol (IMEP [30]) for other underlying functions.

TORA can be separated into three basic functions: creating routes, maintaining routes, and erasing routes. The creation of routes basically assigns directions to links in an undirected network or portion of the network, building a directed acyclic graph (DAG) rooted at the destination (See Figure 11).

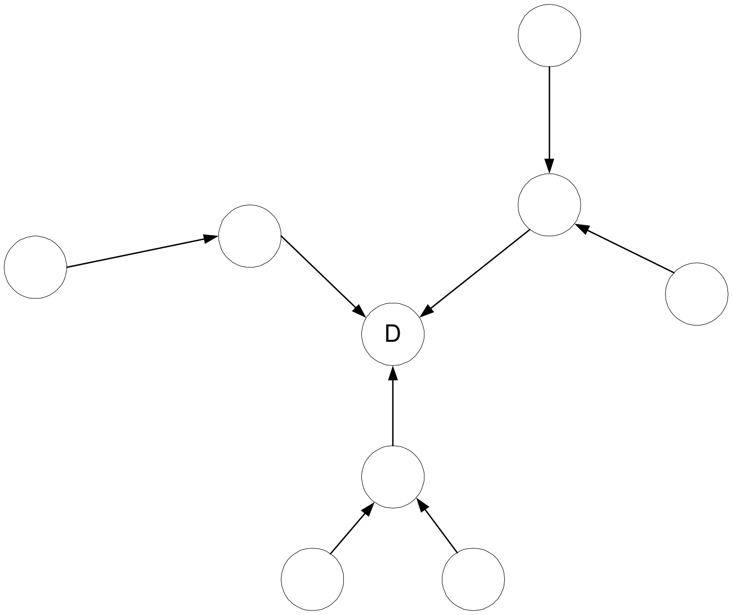


Figure 11: Directed acyclic graph rooted at destination

TORA associates a height with each node in the network. All messages in the network flow downstream, from a node with higher height to a node with lower height. Routes are discovered using Query (QRY) and Update (UPD) packets. When a node with no downstream links needs a route to a destination, it will broadcast a QRY packet. This QRY packet will propagate through the network until it reaches a node that has a route or the destination itself. Such a node will then broadcast a UPD packet that contains the node height. Every node receiving this UPD packet will set its own height to a larger height than specified in the UPD message. The node will then broadcast its own UPD packet. This will result in a number of directed links from the originator of the QRY packet to the destination. This process can result in multiple routes.

Maintaining routes refers to reacting to topological changes in the network in a manner such that routes to the destination are re-established within a finite time, meaning that its directed portions return to a destination-oriented graph within a finite time. Upon detection of a network partition, all links in the portion of the network that has become partitioned from the destination are marked as undirected to erase invalid routes. The erasing of routes is done using clear (CLR) messages.

3.7.5.2. Properties

The protocols underlying link reversal algorithm will react to link changes through a simple localized single pass of the distributed algorithm. This prevents CLR packets to propagate too far in the network. A comparison made by the CMU Monarch project has however shown that the overhead in TORA is quite large because of the use of IMEP.

The graph is rooted at the destination, which has the lowest height. However, the source originating the QRY does not necessarily have the highest height. This can lead to the situation, where multiple routes are possible from the source to the destination, but only one route will be discovered. The reason for this is that the height is initially based on the distance in number of hops from the destination.

3.7.6. Internet MANET Encapsulation Protocol –IMEP

3.7.6.1. Description

IMEP [10] is a protocol designed to support the operation of many routing protocols in Ad-hoc networks. The idea is to have a common general protocol that all routing protocols can make use of (see Figure 12). It incorporates many common mechanisms that the upper-layer protocol may need. These include:

- ❖ Link status sensing
- ❖ Control message aggregation and encapsulation
- ❖ Broadcast reliability
- ❖ Network-layer address resolution
- ❖ Hooks for inter router security authentication procedures

IMEP also provides architecture for MANET router identification, interface identification and addressing. IMEPs purpose is to improve overall performance by reducing the number of control messages and to put common functionality into one unified, generic protocol useful to all upper-level routing protocols.

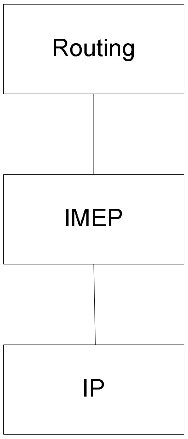


Figure 12: IMEP in the protocol stack

Of the currently proposed protocols, only TORA and OLSR use IMEP. It must however be noted that TORA and IMEP were designed by the same author.

3.7.6.2. Properties

The idea to have a general protocol for common basic feature is good, but from the performance point of view this is not a good idea. It adds another layer to the protocol stack as shown in the figure. As the work by the CMU Monarch project has shown [31], IMEP generates a lot of overhead mainly because of IMEP's neighbor discovery mechanism that generates at least one hello message per second in contrast it delivers the packets reliably.

3.7.7. Cluster Based Routing Protocol – CBRP

3.7.7.1. Description

The idea behind CBRP [24] is to divide the nodes of an ad-hoc network into a number of overlapping or disjoint clusters. One node is elected as cluster head for each cluster. This cluster head maintains the membership information for the cluster. Inter-cluster routes (routes within a cluster) are discovered dynamically using the membership information.

CBRP is based on source routing, similar to DSR. This means that intra cluster routes (routes between clusters) are found by flooding the network with Route Requests (RREQ). The difference is that the cluster structure generally means that the number of nodes disturbed is much less. Flat routing protocols, i.e. only one level of hierarchy, might suffer from excessive overhead when scaled up. CBRP is like the other protocols fully distributed. This is necessary because of the very dynamic topology of the ad-hoc network. Furthermore, the protocol takes into consideration the existence of unidirectional links.

3.7.7.2. Link Sensing

Each node in CBRP knows its bi-directional links to its neighbors as well as unidirectional links from its neighbors to itself. To handle this, each node must maintain a Neighbor Table (see Table 1).

Nighbor ID	**Link Status**	**Role**
Neighbor 1	Bi/unidirectional link to me	Is 1 a cluster head or member
Nighbor 2	Bi/unidirectional link to me	Is 2 a cluster head or member
…	…	…
Neighbor n	Bi/unidirectional link to me	Is n a cluster head or member

Table 1: Neighbor Table CBRP

Each node periodically broadcasts its neighbor table in a hello message. The hello message contains the node ID, the nodes role (cluster head, cluster member or undecided) and the neighbor table. The hello messages are used to update the neighbor tables at each node. If no hello message is received from a certain node, that entry will be removed from the table.

3.7.7.3. Clusters

The cluster formation algorithm is very simple; the node with lowest node ID is elected as the cluster head. The nodes use the information in the hello messages to decide whether or not they are the cluster heads. The cluster head regards all nodes it has bi-directional links to as its member nodes. A node regards itself as a member node to a particular cluster if it has a bi-directional link to the cluster head. It is possible for a node to belong to several clusters.

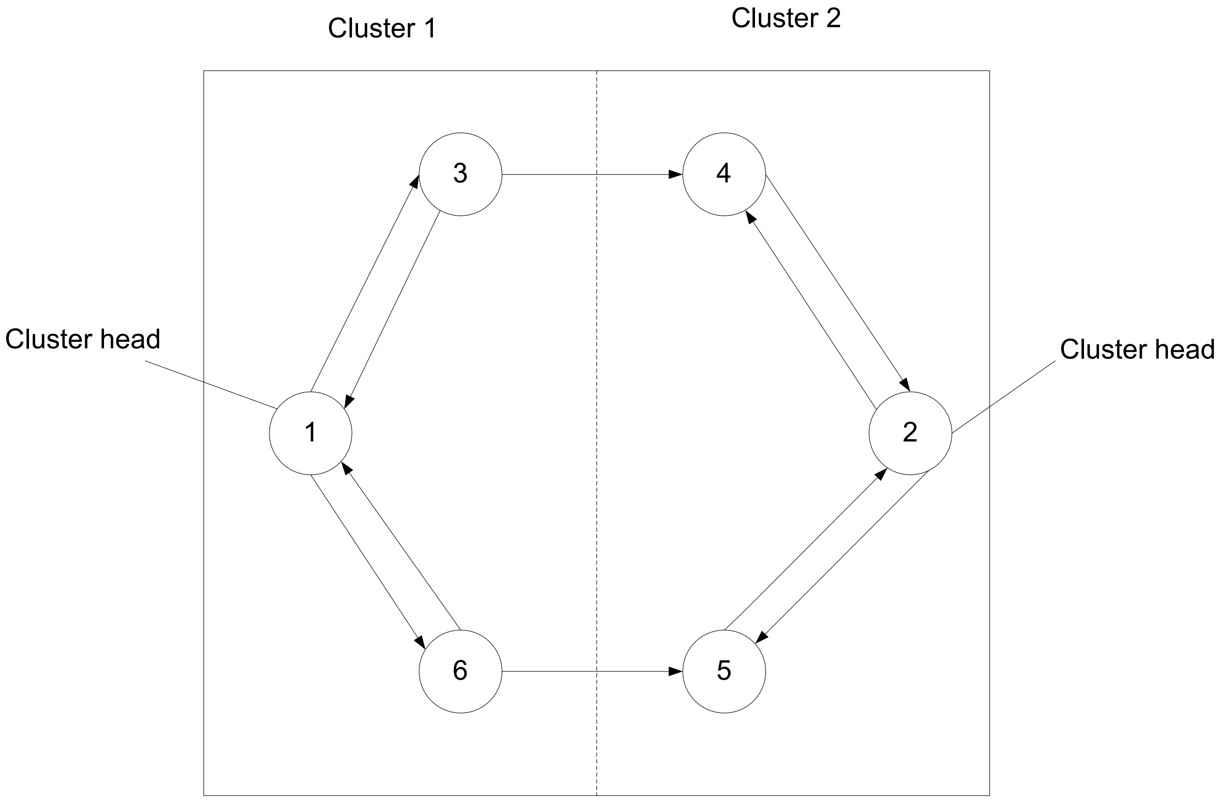


Figure 13: Bi-directional linked clusters

Clusters are identified by their respective cluster heads, which means that the cluster head must change as infrequently as possible. The algorithm is therefore not a strict "lowest ID" clustering algorithm. A non-cluster head never challenges the status of an existing cluster head. Only when two luster-heads move next to each other, will one of them lose the role as cluster head. In Figure 13 node 1 is cluster head for cluster 1 and node 2 is cluster head for cluster 2.

3.7.7.4. Routing

Routing in CBRP is based on source routing and the route discovery is done, by flooding the network with Route Requests (RREQ). The clustering approach however means that fewer nodes are disturbed. This is due to because, only the cluster heads are flooded. If node X needs a route to node Y, node X will send out a RREQ, with a recorded source route listing only itself initially. Any node forwarding this packet will add its own ID in this RREQ. Each node forwards a RREQ only once and it never forwards it to node that already appears in the recorded route.

In CBRP, a RREQ will always follow a route with the following pattern:

Source->Cluster head->Gateway->Cluster head->Gateway-> ...->Destination

A gateway node for a cluster is a node that knows that it has a bi-directional or a unidirectional link to a node in another cluster. In Figure 13, node 6 is gateway node for cluster 1 and node 4 is gateway node for cluster 2.

The source unicasts the RREQ to its cluster head. Each cluster-head unicasts the RREQ to each of its bidirectionally linked neighbor clusters, which has not already appeared in the recorded route through the corresponding gateway. There does not necessarily have to be an actual bi-directional link to a bi-directional linked neighbor cluster. For instance, in Figure 13 cluster 1 has a unidirectional link to cluster 2 through node 3 and cluster 2 has a unidirectional link to cluster 1 through node 5, and the clusters are therefore bidirectional linked neighbor clusters. This procedure continues until the target is found or another node can supply the route. When the RREQ reaches the target, the target may chose to memorize the reversed route to the source. It then copies the recorded route to a Route Reply packet and sends it back to the source.

3.7.7.5. Properties

This protocol has a lot of common features with the earlier discussed protocols. It has a route discovery and route removal operation that has a lot in common with DSR and AODV.

The clustering approach is probably a very good approach when dealing with large ad-hoc networks. The solution is more scalable than the other protocols, because it uses the clustering approach that limits the number of messages that need to be sent. CBRP also has the advantage that it utilizes unidirectional links. One remaining question is however how large each cluster should be. This parameter is critical for how the protocol will behave.

3.8. Comparison of protocols

So far, the protocols have been analyzed theoretically. Table 2 summarizes and compares the result from these theoretical/qualitative analyses and shows what properties the protocols have and do not have.

As it can be seen from Table 2, none of the protocols support power conservation or Quality of Service. This is however research question in progress and will probably be added to the protocols. All protocols are distributed, thus none of the protocols is dependent on a centralized node and can therefore easily reconfigure in the event of topology changes.

	DSDV	AODV	DSR	ZRP	TORA/IMEP	CBRP
Loop-free	Yes	Yes	Yes	Yes	No, short lived loops	Yes
Multiple routes	No	No	Yes	No	Yes	Yes
Distributed	Yes	Yes	Yes	Yes	Yes	Yes
Reactive	No	Yes	Yes	Partially	Yes	Yes
Unidirectional link support	No	No	Yes	No	No	Yes
QoS support	No	No	No	No	No	No
Multicast	No	Yes	No	No	No	No
Security	No	No	No	No	No	No
Power conservation	No	No	No	No	No	No
Periodic broadcast	Yes	Yes	No	Yes	Yes (IMEP)	Yes
Requires reliable or sequence data	No	No	No	No	Yes	No

Table 2: Comparison among ad-hoc routing protocols

DSDV is the only proactive protocol in this comparison. It is also the protocol that has most in common with traditional routing protocol in wired networks. The sequence numbers were added to ensure loop-free routes. DSDV will probably be good enough in networks, which allows the protocol to converge in reasonable time. This however means that the mobility cannot be too high. The authors of DSDV came to the same conclusions and designed AODV, which is a reactive version of DSDV. They also added multicast capabilities, which will enhance the performance significantly when one node communicates with several nodes. The reactive approach in AODV has many similarities with the reactive approach of DSR. They both have a route discovery mode that uses request messages to find new routes. The difference is that DSR is based on source routing and will learn more routes than AODV. DSR also has the advantage that it supports unidirectional links. DSR has however one major drawback and it is the source route that must be carried in each packet. This can be quite costly, especially when QoS is going to be used.

ZRP and CBRP are two very interesting proposals that divide the network into several zones/clusters. This approach is probably a very good solution for large networks. Within the zones/clusters they have a more proactive scheme and between the zones/clusters they have a reactive scheme that has many similarities with the operation of AODV and DSR. They have for instance a route discovery phase that sends request through the network. The difference between ZRP and CBRP is how the network is divided. In ZRP all zones are overlapping and in CBRP clusters can be both overlapping and disjoint.

None of the presented protocols are adaptive, meaning that the protocols do not take any smart routing decisions when the traffic load in the network is taken into consideration. As a route selection criteria the proposed protocols use metrics such as shortest number of hops and quickest response time to a request. This can lead to the situation where all packets are routed through the same node even if there are better routes where the traffic load is not huge.

Chapter 4: Simulation Environment

The simulator we have used to simulate the ad-hoc routing protocols in is the Network Simulator 2 (ns) [32] from Berkeley.

4.1 Network Simulator

Network simulator 2 is the result of an on-going effort of research and development that is administrated by researchers at Berkeley. It is a discrete event simulator targeted at networking research. It provides substantial support for simulation of TCP, routing, and multicast protocols.
The simulator is written in C++ and a script language called OTcl[2]. Ns use an Otcl interpreter towards the user. This means that the user writes an OTcl script that defines the network (number of nodes, links), the traffic in the network (sources, destinations, type of traffic) and which protocols it will use. This script is then used by ns during the simulations. The result of the simulations is an output trace file that can be used to do data processing (calculate delay, throughput etc) and to visualize the simulation with a program called Network Animator (NAM). See Appendix I for a screenshot of NAM. NAM is a very good visualization tool that visualizes the packets as they propagate through the network. An overview of how a simulation is done in ns is shown in Figure 14.

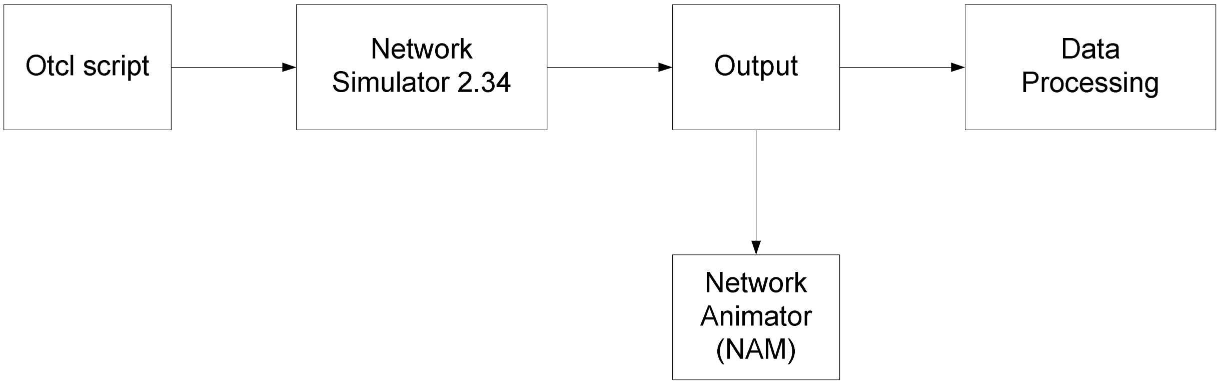


Figure 14: Network simulator 2.35

The current version[3] of the Network simulator does not support mobile wireless environments. The Network simulator alone is only intended for stationary networks with wired links. This caused us some problems in the beginning of this master thesis. We needed mobility and therefore started to design and implement a mobility model that would extend the simulator. We also started to implement the AODV protocol. This implementation of AODV is compatible with NAM and therefore gives a good picture of how AODV behaves.

4.2 Node mobility

Each mobile node is an independent entity that is responsible for computing its own position and velocity as a function of time. Nodes move around according to a movement pattern specified at the beginning of the simulation.

4.3 Physical layers

Propagation models are used to decide how far packets can travel in air. These models also consider propagation delays, capture effects and carrier sense [33].

4.4 MAC 802.11

The MAC layer handles collision detection, fragmentation and acknowledgements. This protocol may also be used to detect transmission errors. 802.11 is a CSMA/CA (Carrier Sense Multiple Access with Collision Avoidance) protocol. It avoids collisions by checking the channel before using it. If the channel is free, it can start sending, if not, it must wait a random amount of time before checking again. For each retry an exponential backoff algorithm will be used. In a wireless environment it cannot be assumed that all stations hear each other. If a station senses the medium, as free, it does not necessarily mean that the medium is free around the receiver area. This problem is known as the hidden terminal problem and to overcome these problems the Collision Avoidance mechanism together with a positive acknowledgement scheme is used. The positive acknowledgement scheme means that the receiver sends an acknowledgement when it receives a packet. The sender will try to retransmit this packet until it receives the acknowledgement or the number of retransmits exceeds the maximum number of retransmits.

802.11 also support power saving and security. Power saving allows packets to be buffered even if the system is asleep. Security is provided by an algorithm called Wired Equivalent Privacy (WEP). It supports authentication and encryption. WEP is a Pseudo Random Number Generator (PRNG) and is based on RSAs RC4.

One of the most important features of 802.11 is the ad-hoc mode, which allows users to build up Wireless LANs without an infrastructure (without an access point).

4.5 Address Resolution Protocol

The Address Resolution Protocol, ARP [34] is implemented. ARP translates IP-addresses to hardware MAC addresses. This takes place before the packets are sent down to the MAC layer.

4.6 Radio network interfaces

This is a model of the hardware that actually transmits the packet onto the channel with a certain power and modulation scheme [35].

4.7 Transmission power

The radius of the transmitter with an omni-directional antenna is about 250 meters in this extension.

4.8 Antenna gain and receiver sensitivity

Different antennas are available for simulations.

4.9 Ad-hoc routing protocols

Both DSR and DSDV have been implemented

4.10 Shared media

The extension is based on a shared media model (Ethernet in the air). This means that all mobile nodes have one or more network interfaces that are connected to a channel (see Figure 15). A channel represents a particular radio frequency with a particular modulation and coding scheme. Channels are orthogonal, meaning that packets sent on one channel do not interfere with the transmission and reception of packets on another channel. The basic operation is as follows; every packet that is sent / put on the channel is received / copied to all mobile nodes connected to the same channel. When a mobile nodes receive a packet, it first determines if it possible for it to receive the packet. This is determined by the radio propagation model, based on the transmitter range, the distance that the packet has traveled and the amount of bit errors.

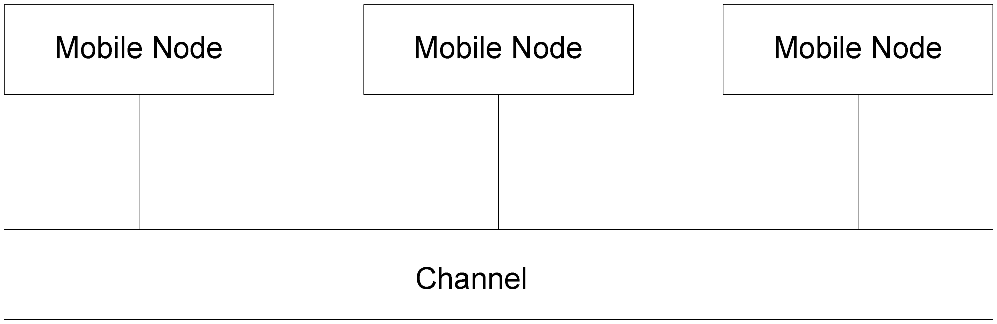


Figure 15: Shared media model

4.11 Mobile node

Each mobile node (Figure 16) makes use of a routing agent for the purpose of calculating routes to other nodes in the ad-hoc network. Packets are sent from the application and are received by the routing agent. The agent decides a path that the packet must travel in order to reach its destination and stamps it with this information. It then sends the packet down to the link layer. The link layer level uses an Address Resolution Protocol (ARP) to decide the hardware addresses of neighboring nodes and map IP addresses to their correct interfaces. When this information is known, the packet is sent down to the interface queue and awaits a signal from the Multiple Access Control (MAC) protocol. When the MAC layer decides it is ok to send it onto the channel, it fetches the packet from the queue and hands it over to the network interface which in turn sends the packet onto the radio channel. This packet is copied and is delivered to all network interfaces at the time at which the first bit of the packet would begin arriving at the interface in a physical system. Each network interface stamps the packet with the receiving interfaces properties and then invokes the propagation model.

The propagation model uses transmit and receive stamps to determine the power with which the interface will receive the packet. The receiving network interfaces then use their properties to determine if they actually successfully received the packet, and send it to the MAC layer if appropriate. If the MAC layer receives the packet error- and collision- free, it passes the packet to the mobiles entry point. From there it reaches a demultiplexer, which decides if the packet should be forwarded again, or if it has reached its destination node. If the destination node is reached, the packet is sent to a port demultiplexer, which decides to what application the packet should be delivered. If the packet should be forwarded again the routing agent will be called and the procedure will be repeated.

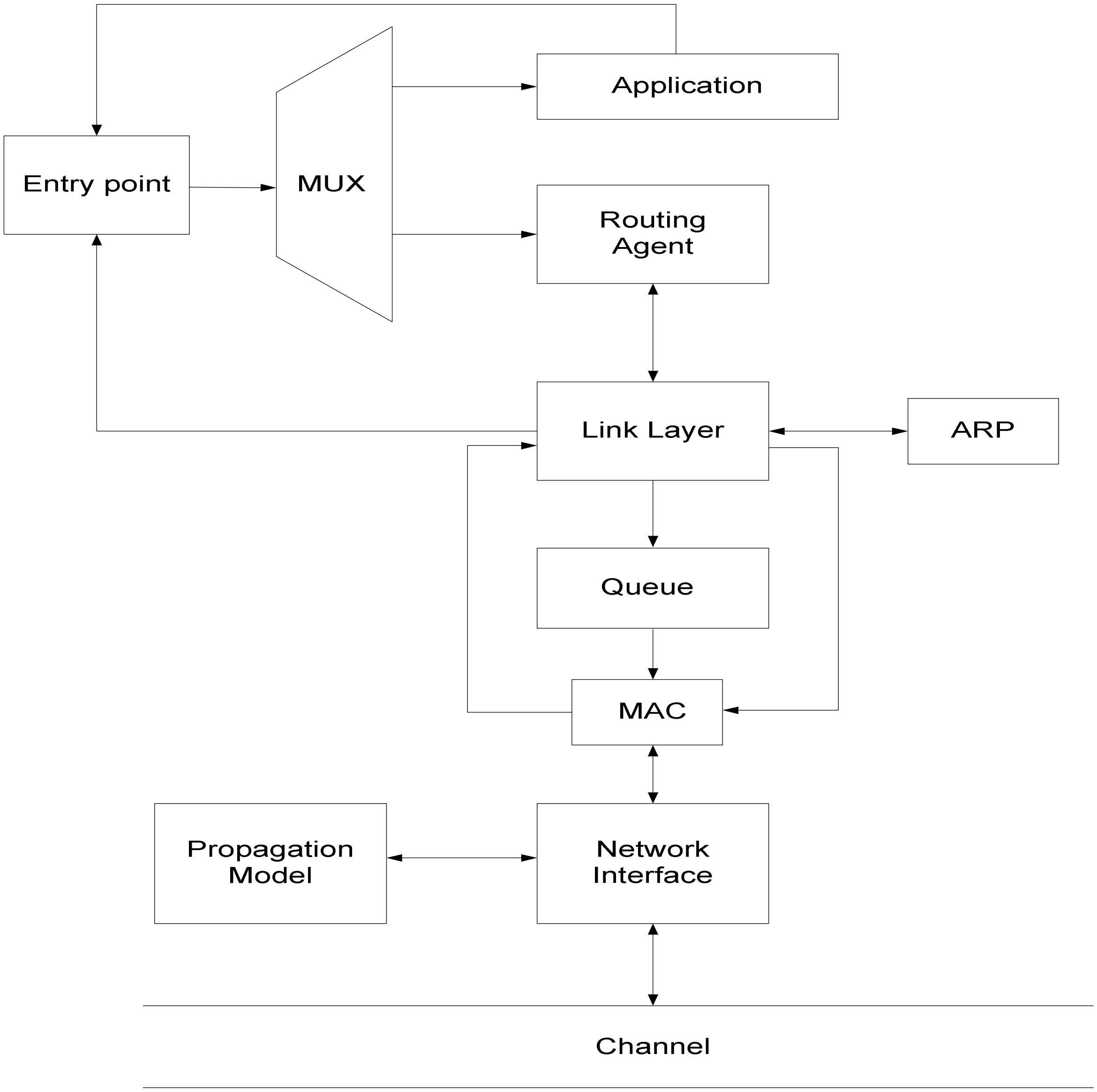


Figure 16: A mobile node

4.12 Simulation in NS-2

To successfully carry out simulation, we must first tell ns-2, things it may need from us for simulation. The necessary parameters to configure are outlined as below.

a) Appearance of the network: the whole topology view of sensor network or mobile network, this includes the position of nodes with (x, y, z) coordinate, the node movement parameters, the movement starting time, the movement direction, and the node movement speed with pausing time between two recurrent movement.
b) Internal of the network: Since the simulation is on the network traffic, so it is important to tell ns2 about which nodes are the sources, about the connections, and links we want to use.
c) Configuration of the layered structure of each node in the network, this includes the detail configuration of network components on wireless node, from driving a simulation to obtaining the simulation results in the form of trace file, and organization of a simulation process.

4.7.1. Simple simulation using tcl

<u>Step by step procedure</u>

Step 1. Create an instance of the simulator:
set ns_ [new Simulator]
Step.2. Setup trace support by opening file ”tracefile.tr” and call the procedure trace-all
set tracefd [open tracefile.tr w]
$ns_ trace-all $tracefd
Step 3. Create a topology object that keeps track # of all the nodes within boundary
set topo [new Topography]
Step 4. The topography is broken up into grids and the default value of grid resolution is 1. A different value can be passed as a third parameter to load_flatgrid { }.
$topo load_flatgrid $val(x) $val(y)

Step 5. Create the object God, "God (General Operations Director) is the object that is used to store global information about the state of the environment, network or nodes. The procedure create-god is defined in $ns-2.35/tcl/mobility/com.tcl, which allows only a single global instance of the God object to be created during a simulation. God object is called internally by MAC objects in nodes, so we must create god in every cases.

```
set god_ [create-god $val(nn)]
```

Step 6. Before we can create node, we first needs to configure them. Node configuration API may consist of defining the type of addressing (flat/hierarchical etc), for example, the type of adhoc routing protocol, Link Layer, MAC layer, IfQ etc.

```
$ns_ node-config       -adhocRouting $opt(adhocRouting) \
                        -llType $opt(ll) \
                        -macType $opt(mac) \
                        -ifqType $opt(ifq) \
                        -ifqLen $opt(ifqlen) \
                        -antType $opt(ant) \
                        -propType $opt(prop) \
                       -phyType $opt(netif) \
                       -channelType $opt(chan) \
                       -topoInstance $wtopo \
                       -agentTrace ON \
                       -routerTrace ON \
                       -macTrace OFF
```

Step 7. Create nodes and the random-motion for nodes is disabled here, as we are going to provide node position and movement (speed & direction) directives through the command line discussed next

```
for {set i 0} {$i < $val(nn) } {incr i} {
set node_($i) [$ns_ node]
$node_($i) random-motion 0 # Disable random motion
}
```

Step 8. Give nodes positions to start with, Provide initial (X,Y, for now Z=0) co-ordinates for node_(0) and node_(1). Node0 has a starting position of (5,2) while Node 1 starts off at location (390,385).

```
$node_(0) set X_ 5.0
$node_(0) set Y_ 2.0
$node_(0) set Z_ 0.0
$node_(1) set X_ 390.0
```

```
$node_(1) set Y_ 385.0
$node_(1) set Z_ 0.0
```

Step 9. Setup node movement as the following example, at time 50.0s, node 1 starts to move towards the destination (x=25, y=20) at a speed of 15m/s. This API is used to change direction and speed of movement of nodes.

```
$ns_ at 50.0 "$node_(1) setdest 25.0 20.0 15.0”
```

Step 10. Setup traffic flow between the two nodes as follows: TCP connections between node_(0) and node_(1)

```
set tcp [new Agent/TCP]
$tcp set class_ 2
set sink [new Agent/TCPSink]
$ns_ attach-agent $node_(0) $tcp
$ns_ attach-agent $node_(1) $sink
$ns_ connect $tcp $sink
set ftp [new Application/FTP]
$ftp attach-agent $tcp
$ns_ at 10.0 "$ftp start"
```

Step 11. Define stop time when the simulation ends and tell nodes to reset which actually resets their internal network components. In the following case, at time 100.0, the simulation shall stop. The nodes are reset at that time and the "$ns_ halt" is called at 100.0002s, a little later after resetting the nodes. The procedure stop{} is called to flush out traces and close the trace file.

```
for {set i 0} {$i < $val(nn) } {incr i} {
$ns_ at 100.0 "$node_($i) reset";
}
$ns_ at 100.0001 "stop"
$ns_ at 100.0002 "puts \"NS EXITING...\" ; $ns_ halt"
proc stop {} {
global ns_ tracefd nf
$ns_flush-trace
close $tracefd
close $nf
}
```

Step 12. Finally the command to start the simulation

```
puts "Starting Simulation...\n" $ns_ run
```

4.7.2. Network scenario generating mobility

For nodes positions and their movement, we can generate a file with the statements which set nodes' positions and nodes movement using CMU generator. It is under ns-2.35/indep-utils/cmu-scen-gen/setdest/

The usage of this executable command is:

```
/setdest [-n num_of_nodes] [-p pausetime] [-s maxspeed]

[-t simtime] [-x maxx] [-y maxy] > [scenario_output_file]
```

Example usage, We are making a scenario for mobility exactly the same way in implementation done in this work.

```
./setdest -n 50 -p $i -M 10.0 -t 100 -x 500 -y 500 > scen-20-$i
```

Where n is the number of nodes or MH
p is the pause time in between recurrent motion of mobile nodes, we have varied the pause time to test on mobility of mobile nodes and therefore $i
M is the maximum moving speed of a node measured in m/s
t Simulation stopping time
The topology boundary is 500m X 500m,
The scenario output is in scen-20-$i

Some fragments of scen-20-$i for AODV are shown in Figure 3.4 below

```
$ns_ at 0.032016276726 "$god_ set-dist 12 13 1"

$ns_ at 0.066078380917 "$god_ set-dist 33 35 1"
```

```
$ns_ at 0.129003781622 "$god_ set-dist 8 38 1"

$ns_ at 0.172803231251 "$god_ set-dist 8 34 1"

$ns_ at 0.178225011288 "$god_ set-dist 5 11 1"

$ns_ at 0.185806276691 "$god_ set-dist 17 41 1"
```

Figure 17: Fragments of scenario generating result

From figure 17, we notice that the node movement is the same pattern as we described before. That is because this scenario generating program uses GOD. Directives for GOD are present as well in node-movement file. The General Operations Director (GOD) object is used to store global information about the state of the environment, network, or nodes that an omniscient observer would have, but that should not be made known to any participant in the simulation. Currently, the god object is used only to store an array of the shortest number of hops required to reach from one node to another. The god object does not calculate this on the fly during simulation runs, since it can be quite time consuming. The information is loaded into the god object from the movement pattern file. And the setdest program generates node-movement files using the random waypoint algorithm. These files already include the lines to load the god object with the appropriate information at the appropriate time.

4.7.3. Creating random traffic-pattern for wireless scenarios

Random traffic connections of TCP and CBR can be setup between mobile nodes using a traffic-scenario generator script. This traffic generator script is available under ~ns-2.35/indep-utils/cmu-scen-gen and is called cbrgen.tcl. It can be used to create CBR and TCP traffics connections between wireless mobile nodes. In order to create a traffic-connection file, we need to define the type of traffic connection (CBR or TCP), the number of nodes and maximum number of connections to be setup between them, a random seed and incase of CBR connections, a rate whose inverse value is used to compute the interval time between the CBR pkts. So the command line looks like the following:

```
ns cbrgen.tcl [-type cbr|tcp] [-nn nodes] [-seed seed] [-mc connections] [-rate rate]
```

For example, the traffic we have generated here in our case would be:

```
ns cbrgen.tcl –type cbr –nn 50 –seed 1.0 –mc $i –rate 4.0 > cbr-50-$i
```

Here we have varied the number of maximum connections to 10, 20. 30 and 40. A sample fragment of traffic generating output is shown in figure 18.

```
# 1 connecting to 2 at time 2.5568388786897245

set udp_(0) [new Agent/UDP]

$ns_ attach-agent $node_(1) $udp_(0)

set null_(0) [new Agent/Null]

$ns_ attach-agent $node_(2) $null_(0)

set cbr_(0) [new Application/Traffic/CBR]

$cbr_(0) set packetSize_ 512

$cbr_(0) set interval_ 0.125

$cbr_(0) set random_ 1

$cbr_(0) set maxpkts_ 10000

$cbr_(0) attach-agent $udp_(0)

$ns_ connect $udp_(0) $null_(0)

$ns_ at 2.5568388786897245 "$cbr_(0) start"

# 4 connecting to 5 at time 56.333118917575632

set udp_(1) [new Agent/UDP]

$ns_ attach-agent $node_(4) $udp_(1)

set null_(1) [new Agent/Null]

$ns_ attach-agent $node_(5) $null_(1)

set cbr_(1) [new Application/Traffic/CBR]
```

```
$cbr_(1) set packetSize_ 512

$cbr_(1) set interval_ 0.125

$cbr_(1) set random_ 1

$cbr_(1) set maxpkts_ 10000

$cbr_(1) attach-agent $udp_(1)

$ns_ connect $udp_(1) $null_(1)

$ns_ at 56.333118917575632 "$cbr_(1) start"
```

And so on..

Figure 18: Traffic generating output

4.7.4. Adding mobility and traffic generating results in tcl file

a. Add two variables in parameter options in the main tcl file

```
set val(cp)            " /home/vpokhrel/NSexamples/traffic/AODV/cbr-50-10"
set val(sc)            " /home/vpokhrel/NSexamples/mobility/AODV/scen-50-0"
```

These are two instance created as depicted for traffic and mobility scenario generation.

4.7.5. Automate generation of traffic and mobility files using Shell script

a. Mobility:

```
#!/bin/bash

dest_dir="/home/vpokhrel/NSexamples/mobility/AODV"

if [ -d $dest_dir ]
then
        # Do nothing
        echo "'$dest_dir' is a directory"
```

```
else
	echo "Creating directory $dest_dir";
	mkdir --verbose $dest_dir
fi

setdest_loc="/root/ns-allinone-2.35/ns-2.35/indep-utils/cmu-scen-gen/setdest/setdest";

if [ -x $setdest_loc ]
then
	# Do nothing
	echo "$setdest_loc is executable"
else
	echo "$setdest_loc does not exist or is not executable";
	exit;
fi

# Create the scenarios

for i in 0 100 200 300 400 500 600 700 800 900
do
	$setdest_loc -v 1 -n 50 -p $i -M 20 -t 900 -x 1000 -y 1000 > $dest_dir/scen-50-$i
done

echo ""
echo "Created the following files"
echo ""
ls -la $dest_dir/scen-50*
```

b. Traffic:

```
!/bin/bash


dest_dir="/home/vpokhrel/NSexamples/traffic/AODV"
```

```
if [ -d $dest_dir ]

then

        # Do nothing

        echo "'$dest_dir' is a directory"

else

        echo "Creating directory $dest_dir";

        mkdir --verbose $dest_dir

fi


script_file="/root/ns-allinone-2.35/ns-2.35/indep-utils/cmu-scen-gen/cbrgen.tcl";


if [ -f $script_file ]

then

        # Do nothing

        echo "$script_file exists"

else

        echo "$script_file does not exist"

        exit;

fi



# Create the scenarios
```

```
for i in 10 18 32 45
do
        ns $script_file -type cbr -nn 50 -seed 1 -mc $i -rate 4.0 > $dest_dir/cbr-50-$i
done

echo "Created the following files"
ls -la $dest_dir/cbr-50*
```

4.7.6. Getting data logs from tracefile

```
#!/bin/bash

for i in 10 20 30 40;
do
        for j in 0 10 20 30 40 50 60 70 80 90 100
        do
        awk -f e2edelay.awk  temptr-$j-$i >> throughput
        done
done
```

where temptr~ is the tracefile incorporating the scenario files as mentioned above,

throughput is the <<output>> data file appending the required value out of trace files.

4.7.7. AWK script for calculating performance KPIs (Example)

```
# AWK Script for calculating:
# Average End-to-End Delay.
BEGIN {
  seqno = -1;
  count = 0;
}
{
  if($4 == "AGT" && $1 == "s" && seqno < $6) {
        seqno = $6;
  }
  #end-to-end delay
  if($4 == "AGT" && $1 == "s") {
        start_time[$6] = $2;
  } else if(($7 == "AODV") && ($1 == "r")) {
      end_time[$6] = $2;
  } else if($1 == "D" && $7 == "AODV") {
        end_time[$6] = -1;
  }
}
END {
  for(i=0; i<=seqno; i++) {
        if(end_time[i] > 0) {
            delay[i] = end_time[i] - start_time[i];
                count++;
      }
          else
          {
                delay[i] = -1;
          }
  }
  for(i=0; i<=seqno; i++) {
        if(delay[i] > 0) {
            n_to_n_delay = n_to_n_delay + delay[i];
      }
  }
 n_to_n_delay = n_to_n_delay/count;
 print "Average End-to-End Delay    = " n_to_n_delay * 1000 " ms";
}
```

4.13 Simulation Overview (summary)

A typical simulation with ns and the mobility extension is shown in Figure 17. Basically it consists of generating the following input files to ns:

- A scenario file that describes the movement pattern of the nodes.
- A communication file that describes the traffic in the network.

These files can be generated by drawing them by hand using the visualization tool Ad-hockey or by generating completely randomized movement and communication patterns with a script like shell or Perl.

These files are then used for the simulation and as a result from this, a trace file is generated as output. Prior to the simulation, the parameters that are going to be traced during the simulation must be selected. The trace file can then be scanned and analyzed for the various parameters that we want to measure. This can be used as data for plots with for instance GNUPLOT. The trace file can also be used to visualize the simulation run with either Ad-hockey or Network animator.

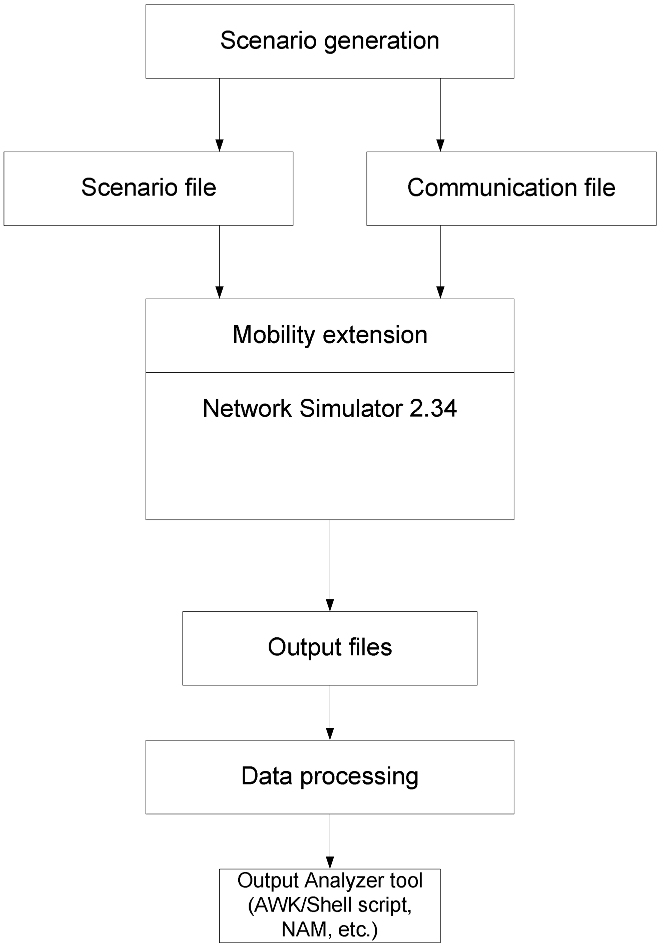


Figure 19: Simulation Overview

Chapter 5: Simulation and Result

5.1 Simulation study

The protocols that we have simulated are DSDV, AODV and DSR. DSDV is only used to get a comparison of how much better/worse the MANET protocols are than an ordinary proactive protocol. The simulations were conducted on Virtual Intel PC with a Pentium-4 processor at 1.5 GHz, 1.5GB Mbytes of RAM running CentOS 5.

5.2 Measurements

Before we go into the actual simulations, we will discuss which parameters [10] that are interesting to measure when studying routing protocols in an ad-hoc network. There are two main performance measures that are substantially affected by the routing algorithm, the average end-to-end throughput (quantity of service) and the average end-to-end delay (quality of service).

5.3 Scenario

The metrics has to be measured against some parameter that describes the characteristic behavior of an Adhoc network and can be varied in a controlled way. The parameters that we have chosen to simulate with are:

- Mobility, which probably is one of the most important characteristics of an ad-hoc network. This will affect the dynamic topology; links will go up and down. This is determined by the varying the pause time in between node movements.
- Offered network load. The load that we actually offer the network. This can be characterized by three parameters: packet size, number of connections and the rate that we are sending the packets with.
- Network size (number of nodes, the size of the area that the nodes are moving within). The network size basically determines the connectivity. Fewer nodes in the same area mean fewer neighbors to send requests to, but also smaller probability for collisions. In this work, we have keep a fixed mobile nodes and size.

5.4 Mobility

Because mobility is an important metric when evaluating ad-hoc networks we need some definition of mobility. There exist many definitions of mobility. The CMU Monarch project [31] has for instance used the pause time in the waypoints as a definition of mobility. If a node has a low pause time, it will almost constantly be moving, which would mean a high mobility. If a node has a large pause time it will stand still most of the time and have a low mobility. We have considered mobility a little differently. Our definition is based on the relative movement of the nodes. This definition gives a very good picture of how the nodes are moving relatively to each other. The definition is as follows:

- ❖ If several nodes move for a certain time, then the mobility is the average change in distance between all nodes over that period of time. This time is the simulation time T.
- ❖ Mobility is a function of both the speed and the movement pattern. It is calculated with a certain sampling rate. During the simulations, we have used 0.1 seconds as sampling rate. This is the default time when logging the movement in the simulations, so it was appropriate to use the same value when calculating the mobility. Table 3 shows all variables that are used in the equations for the mobility factor.

Variable name	**Description**
dist(n_x,n_y)t	the distance between node x and node y at time t
n	Number of nodes
i	Index
A_x(t)	Average distance for node x to all other nodes at time t
M_x	Average mobility for node x relative to all other nodes during the entire simulation time
T	Simulation time
Δt	Granularity, simulation step
Mob	Mobility for entire scenario

Table 3: Mobility variables

First of all, the average distance from each node to all other nodes has to be calculated. This has to be done at times t=0, t=0+X, t=0+2X, …, t= simulation time. For the node x at time t the formula is:

$$A_x(t)=\frac{\sum_{i=1}^{n} dist(n_x,n_i)}{n-1} \qquad (4.1)$$

After that, with the use of (4.1), the average mobility for that particular node has to be calculated. This is the average change in distance during a whole simulation. The mobility for node x is:

$$M_x = \frac{\sum_{t=0}^{T-\Delta t} |(A_x(t) - A_x(t+\Delta t))|}{T-\Delta t} \qquad (4.2)$$

Finally, the mobility for the whole scenario is the sum of the mobility for all nodes (4.2) divided with the number of nodes:

$$\text{Mob} = \frac{\sum_{i=1}^{n} Mi}{n} \qquad (4.3)$$

The unit for the mobility factor (4.3) is m/s. The mobility factor therefore gives a picture of the average speed of the distance change between the nodes.

Figure 18 shows some basic examples of how this mobility factor will reflect the actual movement. If the nodes are standing still, this will of course lead to a mobility of 0, but this would also be the case when the nodes relative movement is zero, for example when the nodes are moving in parallel with the same speed. It is only when the nodes have a movement relative to each other that the mobility factor will be greater than zero.

Our mobility definition reflects how the mobility affects the dynamic topology, without considering obstacles or surroundings.

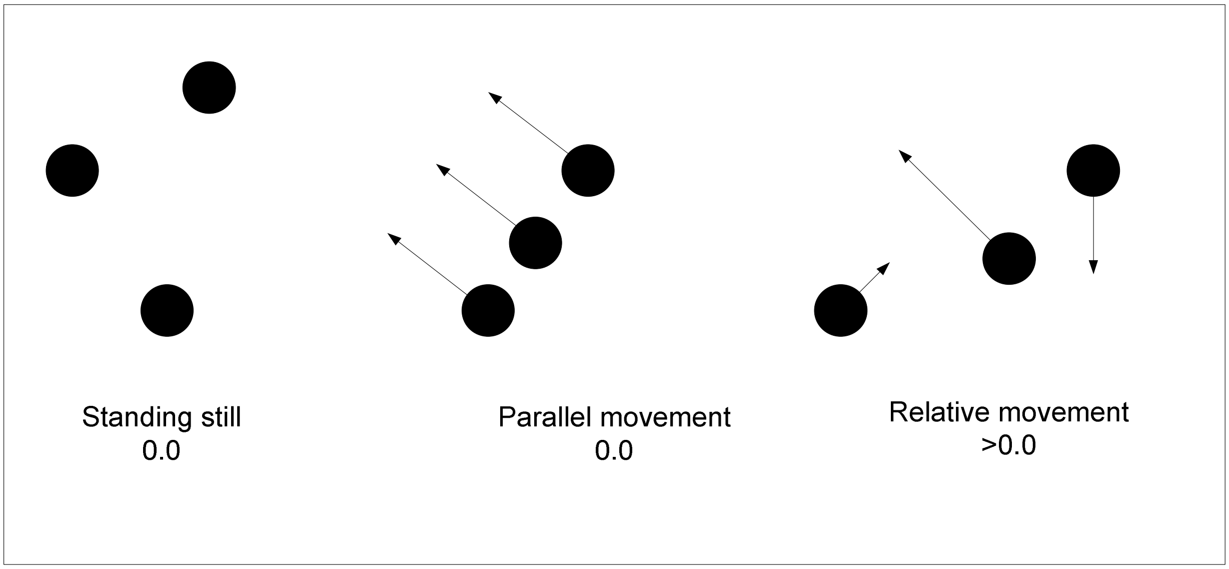


Figure 20: Example of mobility

5.5 Mobility Vs. Link change

The reason for choosing mobility as a parameter in the simulations is that the mobility is one of the most important characteristics of an ad-hoc network. Also because mobility is a parameter that is easy to grasp for people in general. Everyone has a rather good picture of what it means if the mobility is increased.

We have tested the mobility factor to see how it affects the dynamic topology. As it can be seen in Figure 19, the number of link changes is directly proportional to the mobility factor. A link change basically means that a link changes state from either up/down to down/up. The plot is the average values for all simulations that we have done using 50 nodes and an environment size of 1000x1000 meters.

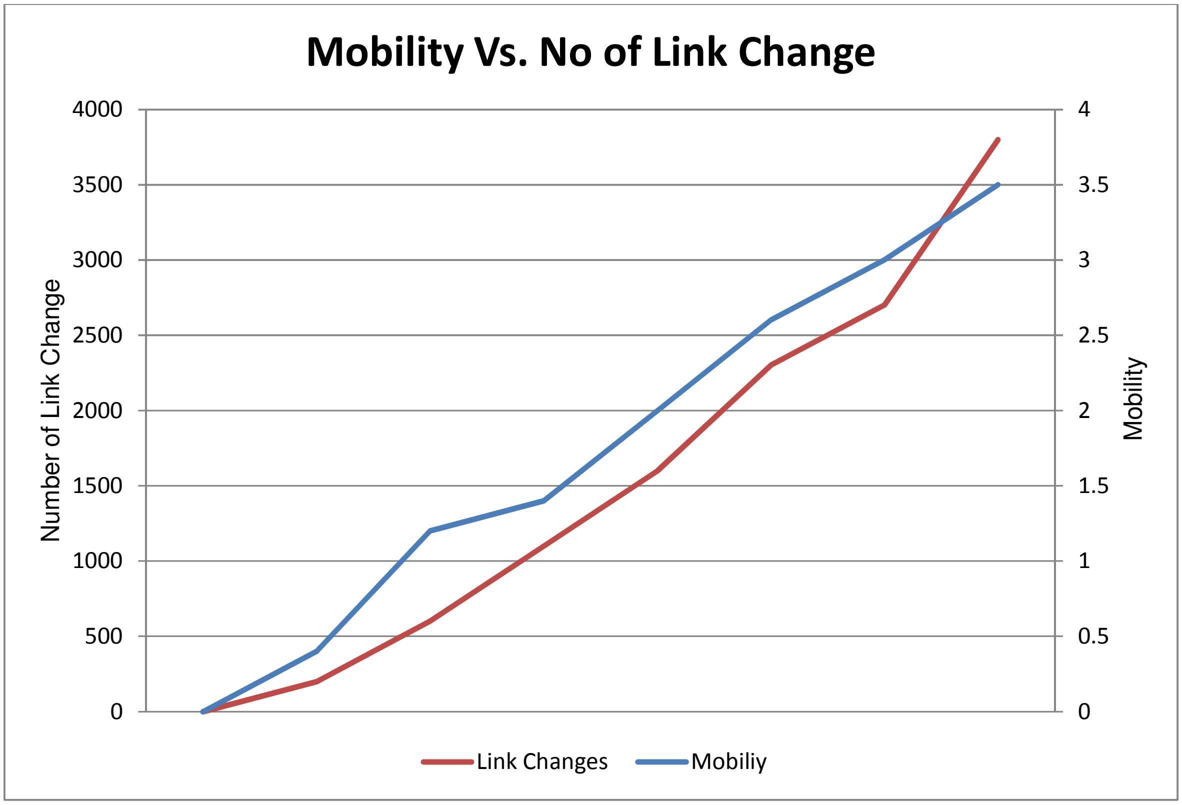


Figure 21: Mobility

5.6 Experimental setup

For our scenario based experiments, we used the ns-2 simulator which is available as an open source distribution [13]. For generating the scenarios, we used the mobility scenario generation tool, *BonnMotion*. We utilized CMU's wireless extensions to the ns-2 simulator, which is based on a two-ray ground reflection model. The radio model corresponds to the 802.11 WaveLAN, operating at a maximum air-link rate of 2 Mbps. The Media Access Control protocol used is the IEEE 802.11 Distributed Coordination Function (DCF). The traffic pattern file was generated using “cbrgen.tcl” script, which is provided along with the standard ns-2 distribution. We used CBR traffic with the following parameters for our simulations –

Traffic Pattern	
Maximum number of connections	10, 20, 30, and 40
Application data payload size	512 bytes
Packet size	4 packets/sec

Thus, effectively a bandwidth of 16 Kbps was used, which corresponds to applications such as the Combat Network Radio (CNR), which are self-forming networks comprised of highly mobile radios that can transmit voice and data for disaster/battlefield operations.

5.7 Antenna Propagation Model

In wireless communications, the channel is responsible for modifying the original electromagnetic waves by reflection, diffraction, and scattering1. Direct line-of-sight models (free space propagation) and multipath fading models have been created to represent the channel of a wireless system. Regardless of the model been used, the electromagnetic wave power in a wireless channel is a function of the distance between transmitter and receiver. As distance increases, power decreases [37].

5.7.1. Free space propagation model

The free space propagation model is implemented whenever there are no obstructions between transmitter and receiver, and assumes a single path in between them. In this model, the power received by an antenna that is separated by a distance d from the transmitter is given by Friss free space equation,

$$\Pr(d) = \frac{P_t G_t G_r \lambda^2}{(4\pi)^2 d^2 L}$$

where Pr(d) is the received power at a distance d from transmitter, P_t is the transmitted power, G_t is the transmitter antenna gain, G_r the receiver antenna gain, λ the wavelength in meters and L is the system loss factor not related to propagation. L should be equal or greater than 1.

5.7.2. Two-Ray ground reflection model

A more accurate model involves both the direct path and a ground reflected propagation path between transmitter and receiver. This model is known as the two-ray ground reflection model, and is useful to predict signal strengths over distances of several kilometers. The received power at a given distance d from transmitter is given by

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

where Pr(d) is the received power at a distance d from transmitter, P_t is the transmitted power, G_t is the transmitter antenna gain, G_r the receiver antenna gain, h_t is the transmitter height, h_r is the receiver height, and *d* is the distance between transmitter and receiver. The two-ray model shows a faster path loss than the free space model. We can observe from the last equation that at a large distance, $d >> \sqrt{h_t h_r}$, the receiver power falls off at a rate of 40dB per decade [Rapp124]. In general, it can be observed that how in the two-ray model, the power falls off faster than in the free space one.

5.8 Simulation evaluation for performance KPIs (Key Performance Indicator) and Results

The following are the metrics which we have used for the performance analysis –

a. ***Packet Delivery Fraction (PDF)***

This is the ratio of total number of packets successfully received by the destination nodes to the number of packets sent by the source nodes throughout the simulation.

$$PDF = \frac{numberofreceivedpackets}{numberofsentpackets}$$

This estimate gives us an idea of how successful the protocol is in delivering packets to the application layer. A high value of PDF indicates that most of the packets are being delivered to the higher layers and is a good indicator of the protocol performance.

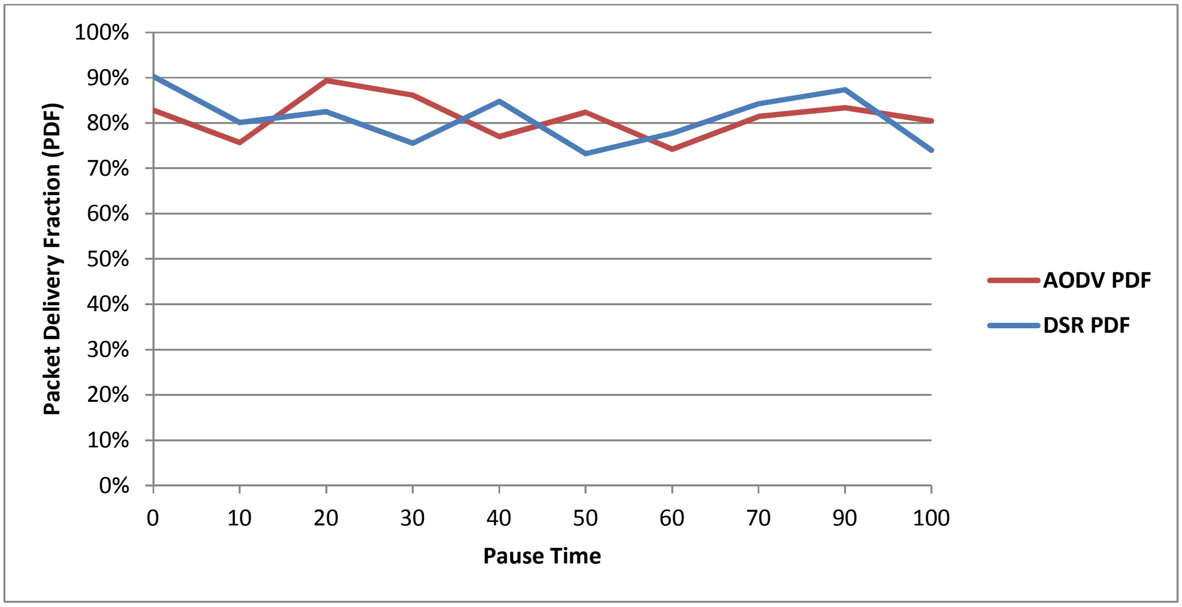


Figure 19: Delivery fractions vs. Pause Time

We varied the pause times from 0 to 100 sec for a disaster scenario and rescue operations. For the traffic, we vary the CBR sources with maximum connection of 10, 20, 30 and 40 taking an average of these connections for various pause times within the range. In this scenario, we found that for higher mobility, i.e. lower pause time, DSR has high PDF than AODV as depicted in the graph. Also for the low mobility with higher pause time AODV outperforms DSR. In summary, the results obtained for both on demand routing protocols look similar.

b. **Throughput**

The network throughput determines the amount of data that is transmitted from a source to a destination node per unit time (bits per second). While ignoring the overheads in the network, only the data of the CBR packets are considered. Node throughput is a measure of the total number of data packets successfully received at the node, having the total number of bits is computed over the simulation runtime. The network throughput is then derived from the average throughput of all nodes involved in the CBR packet transmission.

$$AvgerageThroughput = \frac{TotalBits\mathrm{Re}ceivedByNodes}{SimulationTime}$$

This estimates the number of received size from a node a to node b with respect to the observation duration. A high value of throughput indicates transmit of data packets not limited by the transmission link between the nodes. Whereas the low value throughput indicates loss of data packets in between the source and destination link due to various factors like limited bandwidth, jitter and higher signal to noise ratio.

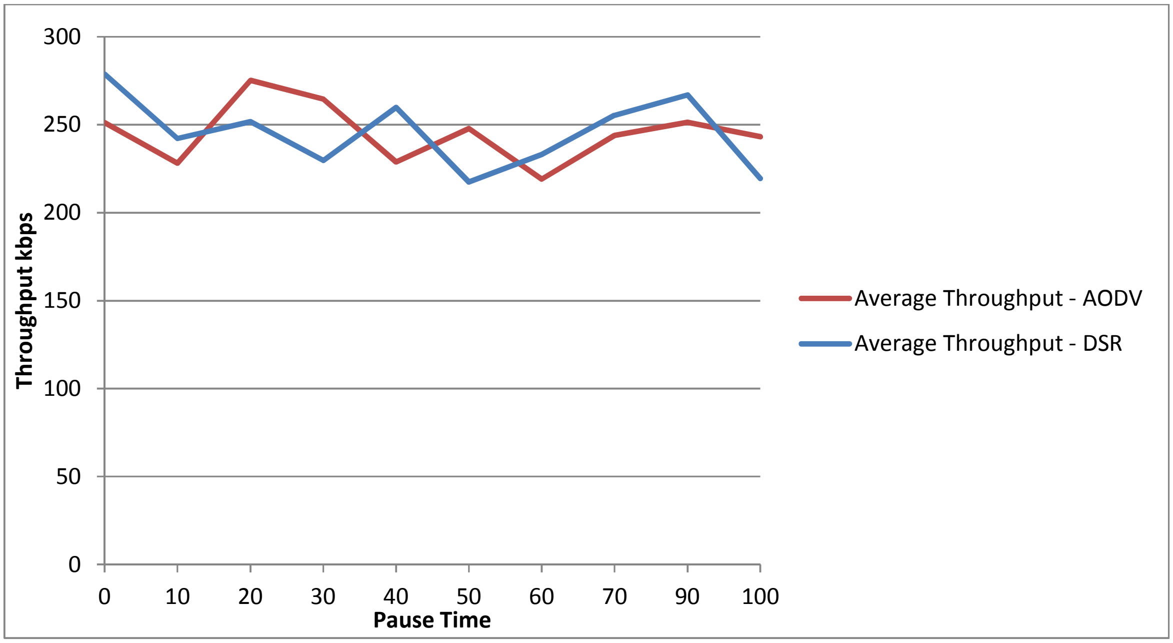


Figure 20: Throughputs vs. Pause Time

In our simulation, Figure 4.4, DSR out performs AODV when the mobility is high whereas AODV has high throughput when the mobility becomes stationary as the pause time increases. The in-between pause time shows similar behavior in both of these protocols.

c. **Normalized Routing Load**

This is calculated as the ratio between the no. of routing packets transmitted to the number of packets actually received (thus accounting for any dropped packets).

$$NRL = \frac{NumberOfRoutingPacketsSent}{NumberOfDataPackets\mathrm{Re}ceived}$$

This metric gives an estimate of how efficient a routing protocol is since the number of routing packets sent per data packet gives an idea of how well the protocol maintains the routing information updated. Higher the NRL, higher the overhead of routing packets and consequently lower the efficiency of the protocol.

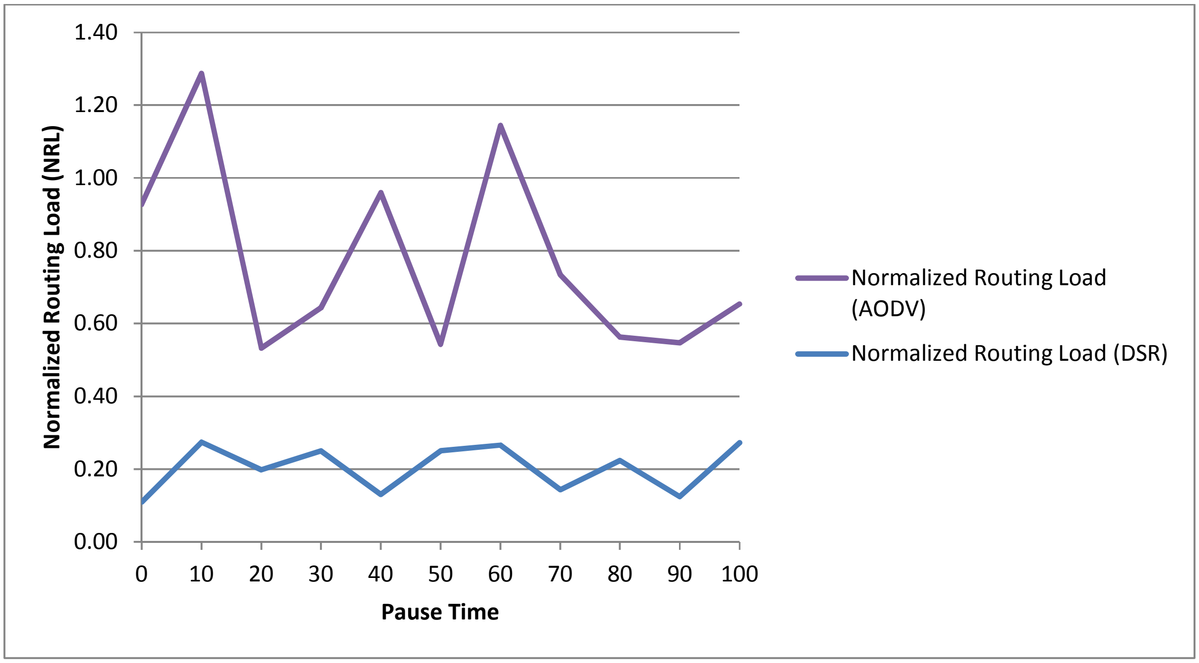


Figure 21: NRL vs. Pause Time

The normalized routing load is analyzed for both protocols by varying paused times. The values for the DSR protocol were less as compared to the AODV which show fairly stable results even if we increase the number of sources. If normalized routing load is stable, the protocol is considered to be scalable. The routing overhead for AODV is mainly from the route requests. DSR finds the route in the cache as a result of aggressive caching. This helps to avoid a frequent route discovery process in DSR thereby decreasing the routing overhead for DSR when compared to AODV.

d. **End to end delay**

Average End-to-End delay (seconds) is the average time it takes a data packet to reach the destination. This metric is calculated by subtracting "time at which first packet was transmitted by source" from "time at which first data packet arrived to destination". This includes all possible delays caused by buffering during route discovery latency, queuing at the interface queue, retransmission delays at the MAC, propagation and transfer times. This metric is significant in understanding the delay introduced by path discovery.

$$Avg.EndToEndDelay = \frac{\sum_{i-0}^{n} Packet\mathrm{Re}ceivedTime - PacketSentTime}{TotalNumberOfPackets\mathrm{Re}ceived}$$

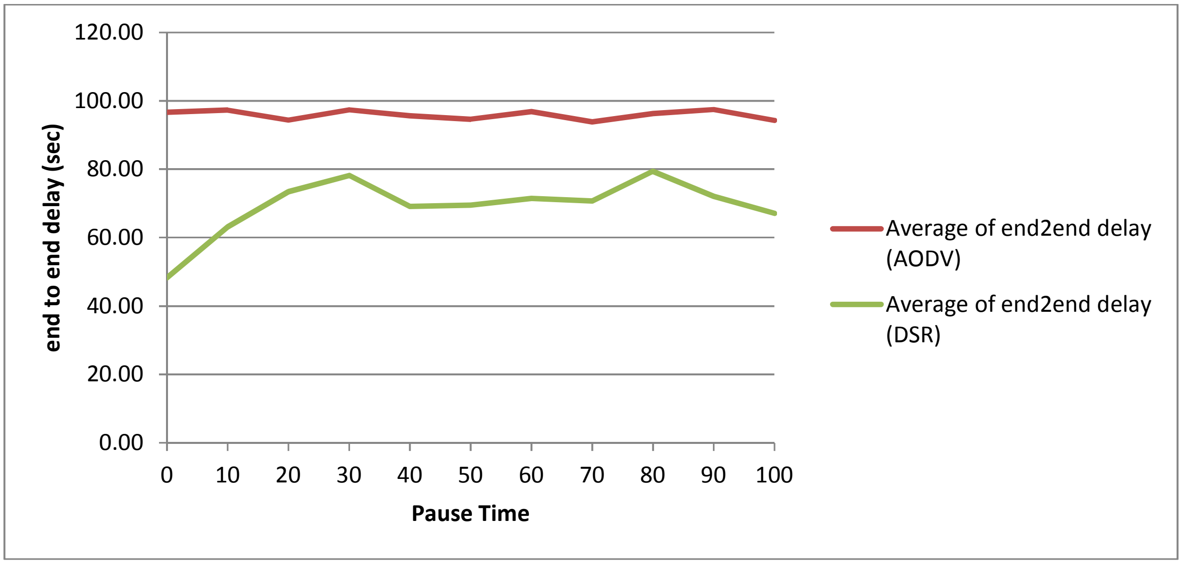


Figure 22: Average end to end delay vs Pause Time

Figure 4.6 illustrates the average end to end delay for 50 nodes with CBR traffic of 4 maximum numbers of connections, 10, 20, 30, 40 varying pause time starting from 0 to 100 seconds. As route breaks, nodes have to discover new routes which lead to longer end-to-end delay (packets are buffered before the route discovery). In this instance of simulation, AODV perform stable irrespective of mobility with high average end-to-end delay. While DSR outperforms AODV throughout the simulation time. At pause time 0, DSR has the lowest end-to-end delay and the delay get increased as the mobility starts but remains below the AODV performance.

Chapter 6: Conclusion Summary and Future work

6.1 Summary

The simulations presented here clearly show that there is a need for routing protocols specifically tuned to the characteristics of ad-hoc networks. The mobility metric used throughout the study explicitly shows how the examined protocols behave for various degrees of relative node motion. The mobility metric is explicitly designed to capture the kind of motion important for an ad-hoc network – the relative motion of nodes. It can be used for any continuous node motion.

In network with a dynamic topology, proactive protocols such as DSDV, though we haven't considered the proactive measures in this thesis, have considerable difficulties in maintaining valid routes, and lose many packets because of that. With increasing mobility, it strives to continuously maintain routes to every node increases network load as updates become larger.

This study clearly indicates that reactive routing protocol is superior to a proactive one. The principle of focusing only on explicitly needed connectivity, and not all connectivity, seems to be excellent when the network consists of moving nodes. In addition, the protocol should be able to detect link failures as quickly as possible to avoid use of invalid routes.

Overall, the proactive protocols under study (AODV and DSR) behaved similarly in terms of delay (PDF) and throughput. On the basis of this study both should be considered suitable for mobile ad-hoc networks. However, a number of differences between the protocols do exist.

The source routes used by DSR gives increased byte over-head compared to AODV when routes have many hops and packet rates are high. DSR is, on the other hand, efficient in finding (learning) routes in terms of the number of control packets used, and does not use periodic control messages.

Data packets in AODV carry the destination address only, and not source routes. Therefore, the byte overhead for AODV is the lowest of the examined protocols. The over-head is however high in terms of packets since AODV broadcast periodic Hello messages to its neighbors, and needs to send control message more frequently than DSR to find and repair routes.

The simulations in this work show that DSR performs better than AODV. This is because in low traffic loads, DSR discovers route more efficiently. At higher traffic loads, however, AODV performs better than DSR due to less additional load being imposed by source routes in data packets.

To conclude, both DSR and AODV performed quite well for almost all examined scenarios. As a preliminary recommendation, DSR should be considered for ad-hoc networks where path have a limited number of hops and where it is crucial to limit packet overhead. AODV on the other hand appears to perform better in networks where paths have many hops and low byte overhead is preferred over low packet overhead.

6.2 Further work

The work presented herein is the first of a series of simulation studies within the area of mobile ad-hoc networking. These studies will include,

- Additional analysis of other proposed protocols (e.g. TORA, ZRP and CBRP)
- Measurements and estimation of power consumption and processing costs,
- Other traffic than CBR (e.g. TCP transfers),
- Inclusion of QoS and encryption (to get away with eavesdropping) mechanisms for real-time and non real-time traffic,
- Evaluation of proposed multicast routing protocols,
- Analysis of interworking function for Mobile IP

Chapter 7: References

[37] B. Tavli, and W. Heinzelman, Mobile Ad Hoc Networks: Energy-Efficient Real-Time Data Communications, Springer, 2006.

[26] Bommaiah, McAuley and Talpade. AMRoute, “Adhoc Multicast Routing Protocol”, Internet draft, draft-talpade-manet-amroute-00.txt, August 1998.

[5] C. A. Santiváñez, R. Ramanathan, and I. Stavrakakis, “Making link-state routing scale for ad hoc networks,” Proc. of the 2nd ACM international symposium on Mobile ad hoc networking & computing, Long Beach, CA, USA, 2001.

[2] C. Perkins and E. Royer, “Ad hoc On Demand Distance Vector (AODV) Routing,” RFC 3561, The Internet Society, July, 2003 and Internet draft, draft-ietf-manet-aodv-01.txt, November 2001.

[14] C. Shen, C. Srisathapornphat and C. Jaikaeo, “Sensor Information Networking Architecture and Applications,” IEEE Pers. Commun., pp. 52-59, Aug. 2001.

[28] "Charles E. Perkins and Pravin Bhagwat, “Highly dynamic Destination-Se quenched Distance-Vector routing (DSDV) for mobile computers”. In proceedings of the SIGCOM ’94 Conference on Communications Architechture, protocols and Applications, pages 234-244, August 1994. A revised version of the paper is available from http://www.cs.umd.edu/projects/mcml/papers/Sigcomm94.ps. (1998-11-29)"

[29] D. B. Johnson and D. A. Maltz, Dynamic Source Routing in Ad Hoc Wireless Networks,Kluwer,v1996.

[7] D. Johnson, D. A. Maltz, and Y.-C. Hu, “The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks (DSR),” Internet Draft, April, 2003, http://www.ietf.org/internet-drafts/draft-ietfmanet-dsr-09.txt.

[23] David B. Johnson and David A. Maltz, “Protocols for adaptive wireless and mobile computing”. In IEEE Personal Communications, 3(1), February 1996.

[9] Dimitri Bertsekas and Robert Gallager, “Data Networks- 2nd ed”. Prentice Hall, New Jersey, ISBN 0-13-200916-1

[17] IETF MANET Working Group, “Mobile Ad Hoc Networks (MANET)”. Working Group charter, available at http://www.ietf.org/html.charters/manet-charter.html.

[31] Josh Broch, David A. Maltz, David B. Johnson, Yih-Chun Hu and Jorjeta Jetcheva, “A performance Comparison of Multi-hop Wireless Ad Hoc Network Routing Protocols”. Mobicom'98, Dallas Texas, 25–30 October, 1998.

[36] K. Fall and K. Varadhan, The NS Manual, The VINT Project, UC Berkeley, January 2002.

[32] Kevin Fall and Kannan Varadhan, “ns notes and documentation”. The VINT project, UC Berkeley, LBL, USC/ISI, and Xerox PARC, May 1998.

[8] Larry L. Peterson and Bruce S. Davie, “Computer Networks – A Systems Approach”. San Francisco, Morgan Kaufmann Publishers Inc. ISBN 1-55860-368-9.

[3] M. Gerla, X. Hong, and G. Pei, “Fisheye State Routing Protocol (FSR) for Ad Hoc Networks,”Internet Draft, June, 2002, http://www.ietf.org/proceedings/02jul/I-D/draft-ietf-manet-fsr-03.txt.

[6] M. Gerla, X. Hong, L. Ma, and G. Pei, “Landmark Routing Protocol (LANMAR) for Large Scale Ad Hoc Networks”, Internet Draft, Nov. 2002, http://www.ietf.org/proceedings/01dec/I-D/draftietf-manet-lanmar-02.txt.

[13] M. S. Corson, J.P. Maker and G.H. Cirincione, “Internet-Based Mobile Ad Hoc Networking,” IEEE Internet Computing, Vol. 3, no.4, July-August 1999, pp. 63-70.

[30] M.Scott Corson, S. Papademetriou, Philip Papadopolous, Vincent D. Park and Amir Qayyum, “An Internet MANET Encapsulation Protocol (IMEP) Specification”. Internet draft, draft-ietf-manet-imepspec01. txt, August 1998.

[33] Martha Steenstrup, “Routing in communication networks”. New Jersey, Prentice Hall. ISBN 0-13- 010752-2.

[24] Mingliang Jiang, Jinyang Li and Yong Chiang Tay, “Cluster Based Routing Protocol (CBRP) Functional specification”. Internet draft, draft-ietf-manet-cbrp-spec-00.txt, August 1998.

[19] Mobile Ad-hoc Networks (MANET) URL: //www.ietf.org/html.charters/manet-charter.html. (1998-11-29)

[18] P Calhoun and C Perkins, “Mobile IP Network Access Identifier Extension for IPv4,” RFC 2794, March 2000

[27] Philippe Jacquet, Paul Muhlethaler and Amir Qayyum, “Optimized Link State Routing Protocol” . Internet draft, draft-ietf-manet-olsr-00.txt, November 1998.

[34] Plummer, D., "An Ethernet Address Resolution Protocol - or - Converting Network Protocol Addresses to 48.bit Ethernet Address for Transmission on Ethernet Hardware", RFC-826, MIT, November 1982.

[4] R. Ogier, M. Lewis, and F. Templin, “Topology Broadcast Based on Reverse Path Forwarding (TBRPF),” Internet Draft, April. 2003, http://www.potaroo.net/ietf/old-ids/draft-ietf-manet-tbrpf-08.txt.

[25] Raghupathy Sivakumar, Prasun Sinha and Vaduvur Bharghavan, "Core Extraction Distributed Ad hoc Routing (CEDAR) Specification", Internet draft, draft-ietf-manet-cedar-spec-00.txt, October 1998.

[15] RL Pickholtz, LB Milstein and DL Schilling, "Spread spectrum for mobile communications, " IEEE Transactions, 1991.

[12] Sarngapani, Jagannathan, "Wireless ad hoc and sensor networks: Protocols, Performance and Control", CRC Press.

[10] Scott Corson and Joseph Macker, "Mobile Ad Hoc Networking (MANET): Routing Protocol Performance Issues and Evaluation Considerations". Internet-Draft, draft-ietf-manet-issues-01.txt, March 1998.

[11] Stephen Kent and Randall Atkinson, "security Architecture for the Internet Protocol", Internet draft, draft-ietf-ipsec-arch-sec-07.txt, July 1998.

[1] T. Clausen, Ed., and P. Jacquet, Ed., "Optimized Link State Routing Protocol (OLSR)," RFC 3626, The Internet Society, Oct., 2003.

[35] Theodore S. Rappaport, "Wireless Communications: Principles and Practice". New Jersey,Prentice Hall. ISBN 0-13-375536-3.

[22] Vincent D. Park and M. Scott Corson, "A performance comparision of the Temporally-Ordered Routing Algorithm and Ideal Link-state routing". In Proceedings of IEEE Symposium on Computers and Communication '98, June 1998.

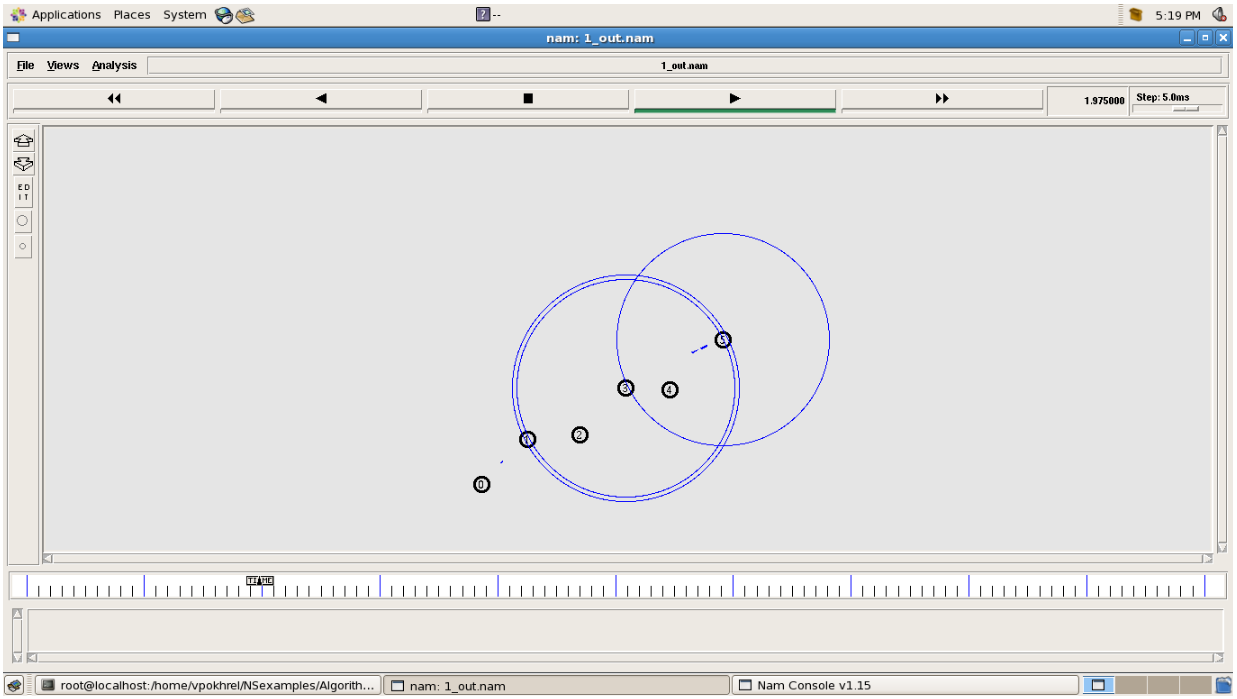
[21] Vincent D. Park and M. Scott Corson, "Temporally-Ordered Routing Algorithm (TORA) Version 1: Functional specification". Internet draft, draft-ietf-manet-tora-spec-01.txt, August 1998.

[16] W. Richard Stevens, "TCP/IP Illustrated, Volume 1: The Protocols", Addison Wesley Professional 1994 / 0-201-63346-9 /

[20] Zygmunt J. Haas and Marc R. Pearlman, "The Zone Routing Protocol (ZRP) for Ad Hoc Networks", Internet draft, draft-ietf-manet-zone-zrp-01.txt, August 1998.

Annex I

I.1 Simulation screenshot visualization



The circle in the figure shows the wireless range of a particular node.

I.2 Trace Sample (AODV)

```
r 1.000000000 _0_ RTR  --- 0 cbr 1000 [0 0 0 0] ------- [0:2 5:0 32 0] [0] 0 0
s 1.000000000 _0_ RTR  --- 0 AODV 48 [0 0 0 0] ------- [0:255 -1:255 30 0] [0x2 1 1 [5 0] [0 4]] (REQUEST)
s 1.000235000 _0_ MAC  --- 0 AODV 106 [0 ffffffff 0 800] ------- [0:255 -1:255 30 0] [0x2 1 1 [5 0] [0 4]] (REQUEST)
r 1.001083471 _1_ MAC  --- 0 AODV 48 [0 ffffffff 0 800] ------- [0:255 -1:255 30 0] [0x2 1 1 [5 0] [0 4]] (REQUEST)
```

```
r 1.001083745 _2_ MAC  --- 0 AODV 48 [0 ffffffff 0 800] ------- [0:255 -1:255 30 0] [0x2 1 1 [5 0] [0 4]] (REQUEST)
r 1.001108471 _1_ RTR  --- 0 AODV 48 [0 ffffffff 0 800] ------- [0:255 -1:255 30 0] [0x2 1 1 [5 0] [0 4]] (REQUEST)
r 1.001108745 _2_ RTR  --- 0 AODV 48 [0 ffffffff 0 800] ------- [0:255 -1:255 30 0] [0x2 1 1 [5 0] [0 4]] (REQUEST)
```

I.3 Trace Sample (DSR)

```
r 1.000000000 _0_ RTR  --- 0 cbr 1000 [0 0 0 0] ------- [0:2 5:0 32 0] [0] 0 0
s 1.004599737 _0_ RTR  --- 1 DSR 32 [0 0 0 0] ------- [0:255 5:255 32 0] 1 [1 1] [0 1 0 0->0] [0 0 0 0->0]
s 1.004674737 _0_ MAC  --- 1 DSR 90 [0 ffffffff 0 800] ------- [0:255 5:255 32 0] 1 [1 1] [0 1 0 0->0] [0 0 0 0->0]
r 1.005395209 _1_ MAC  --- 1 DSR 32 [0 ffffffff 0 800] ------- [0:255 5:255 32 0] 1 [1 1] [0 1 0 0->0] [0 0 0 0->0]
r 1.005395483 _2_ MAC  --- 1 DSR 32 [0 ffffffff 0 800] ------- [0:255 5:255 32 0] 1 [1 1] [0 1 0 0->0] [0 0 0 0->0]
r 1.005420209 _1_ RTR  --- 1 DSR 32 [0 ffffffff 0 800] ------- [0:255 5:255 32 0] 1 [1 1] [0 1 0 0->0] [0 0 0 0->0]
r 1.005420483 _2_ RTR  --- 1 DSR 32 [0 ffffffff 0 800] ------- [0:255 5:255 32 0] 1 [1 1] [0 1 0 0->0] [0 0 0 0->0]
```

I.4 Trace definition

.Event	Abbreviation	Type	Value
Wireless Event	s: Send r: Receive d: Drop f: Forward	%.9f %d (%6.2f %6.2f) %3s %4s %d %s %d [%x %x %x %x]	
		%.9f _%d_ %3s %4s %d %s %d [%x %x %x %x]	
		double	Time
		int	Node ID

		double	X Coordinate (If Logging Position)
		double	Y Coordinate (If Logging Position)
		string	Trace Name
		string	Reason
		int	Event Identifier
		string	Packet Type
		int	Packet Size
		hexadecimal	Time To Send Data
		hexadecimal	Destination MAC Address
		hexadecimal	Source MAC Address
		hexadecimal	Type (ARP, IP)

I.5 AODV and DSR trace definition

DSR Trace	%d [%d %d] [%d %d %d %d->%d] [%d %d %d %d->%d]	
	Int	Number Of Nodes Traversed
	Int	Routing Request Flag
	int	Route Request Sequence Number
	int	Routing Reply Flag
	int	Route Request Sequence Number
	int	Reply Length
	int	Source Of Source Routing
	int	Destination Of Source Routing
	int	Error Report Flag (?)
	int	Number Of Errors
	int	Report To Whom
	int	Link Error From

	int	Link Error To
AODV Trace	[0x%x %d %d [%d %d] [%d %d]] (REQUEST)	
	hexadecimal	Type
	int	Hop Count
	int	Broadcast ID
	int	Destination
	int	Destination Sequence Number
	int	Source
	int	Source Sequence Number
	[0x%x %d [%d %d] %f] (%s)	
	hexadecimal	Type
	int	Hop Count
	int	Destination
	int	Destination Sequence Number
	double	Lifetime
	string	Operation (REPLY, ERROR, HELLO)

I.6 TCL script (AODV)

```
# ================================================================
# Define options
#
=================================================================
set opt(chan)   Channel/WirelessChannel
set opt(prop)   Propagation/TwoRayGround
set opt(netif)  Phy/WirelessPhy
set opt(mac)    Mac/802_11
set opt(ifq)    Queue/DropTail/PriQueue
set opt(ifq)    CMUPriQueue
set opt(ll)                 LL
set opt(ant)        Antenna/OmniAntenna
set opt(x)                  500                         ;# X dimension of the topography
set opt(y)                  500                         ;# Y dimension of the topography
set opt(ifqlen)             50                          ;# max packet in ifq
set opt(seed)               1.0
set opt(tr)                 aodv-25-0-5.tr              ;# trace file
set opt(adhocRouting)   AODV
set opt(ifq)      Queue/DropTail/PriQueue
#set opt(rpr)               1                           ;#1 for DSR and anything else for AODV
set opt(nn)             50                      ;# how many nodes are simulated
set opt(scen)               "movement/scen-25-0"
set opt(tfc)                "traffic/cbr-25-5"
set opt(stop)               100.0                       ;# simulation time

#
========================================================================
# Main Program
#
========================================================================

if { $argc != 6 } {
        puts "Wrong no. of cmdline args."
            puts "Usage: ns compare.tcl -scen <scen> -tfc <tfc> -tr <tr>"
        exit 0
}



# proc getopt {argc argv} {

        for {set i 0} {$i < $argc} {incr i} {
                set arg [lindex $argv $i]
                if {[string range $arg 0 0] != "-"} continue
                set name [string range $arg 1 end]
#                       puts $name
```

```
            set opt($name) [lindex $argv [expr $i+1]]
        }
          set opt(scen) [lindex $argv 1]
          set opt(tfc) [lindex $argv 3]

#       if {$opt(rpr) == 1} {
#         set opt(adhocRouting)   DSR
#         set opt(ifq)      CMUPriQueue
#         set opt(ifq)      Queue/DropTail/PriQueue
#       } else {
#         set opt(adhocRouting)   AODV
#         set opt(ifq) Queue/DropTail/PriQueue
#       }

#         set val(mov) $opt(scen)
#         set val(traf) $opt(tfc)
#         set opt(trace) $opt(tr)

          puts $opt(scen)
          puts $opt(tfc)
          puts $opt(tr)
# }


# getopt $argc $argv



          puts $opt(adhocRouting)
#         puts $val(mov)
#         puts $val(traf)
#         puts $opt(trace)

# Initialize Global Variables
# create simulator instance
```

```
set ns_         [new Simulator]

# set wireless channel, radio-model and topography objects
set wtopo       [new Topography]

# create trace object for ns and nam
set tracefd     [open $opt(tr) w]
$ns_ trace-all $tracefd
# use new trace file format
$ns_ use-newtrace

# define topology
$wtopo load_flatgrid $opt(x) $opt(y)

# Create God
set god_ [create-god $opt(nn)]

#set chan_1_ [new $opt(chan)]
#set chan_2_ [new $opt(chan)]

# define how node should be created
#global node setting
$ns_ node-config -adhocRouting $opt(adhocRouting) \
                 -llType $opt(ll) \
                 -macType $opt(mac) \
                 -ifqType $opt(ifq) \
                 -ifqLen $opt(ifqlen) \
                 -antType $opt(ant) \
                 -propType $opt(prop) \
                 -phyType $opt(netif) \
                 -channelType $opt(chan) \
                 -topoInstance $wtopo \
                 -agentTrace ON \
             -routerTrace ON \
                -macTrace OFF
#           -channel $chan_1_

#  Create the specified number of nodes [$opt(nn)] and "attach" them
#  to the channel.
for {set i 0} {$i < $opt(nn) } {incr i} {
            set node_($i) [$ns_ node]
            $node_($i) random-motion 0              ;# disable random motion
}

# Define node movement model
puts "Loading connection pattern..."
source $opt(scen)

# Define traffic model
puts "Loading traffic file..."
source $opt(tfc)

# Define node initial position in nam
for {set i 0} {$i < $opt(nn)} {incr i} {

    # 20 defines the node size in nam, must adjust it according to your scenario
    # The function must be called after mobility model is defined
   $ns_ initial_node_pos $node_($i) 20
}

# Tell nodes when the simulation ends
for {set i 0} {$i < $opt(nn) } {incr i} {
    $ns_ at $opt(stop).000000001 "$node_($i) reset";
}

# tell nam the simulation stop time
#$ns_ at  $opt(stop)     "$ns_ nam-end-wireless $opt(stop)"
$ns_ at  $opt(stop).000000001 "puts \"NS EXITING...\" ; $ns_ halt"
```

```
puts "Starting Simulation..."
$ns_ run
```