Optimization Of Multicast Routing For Multimedia Traffic Over MANETs

BY

Abhishek Munshaw

10MICT22

Nirma University

AHMEDABAD

May 2012

Optimization Of Multicast Routing For Multimedia Traffic Over MANETs

Major Project

Submitted in partial fulfillment of the requirements

For the degree of

Master of Technology in Information & Communication Technology

BY

Munshaw Abhishek H.

(10MICT22)

Guided by:

Prof. Priyanka Sharma

Nirma University, Ahmedabad

May 2012

Declaration

This is to certify that

i) The thesis comprises my original work towards the degree of Master of Technology in Information and Communication Technology at Nirma University and has not been submitted elsewhere for a degree.

ii) Due acknowledgement has been made in the text to all other material used.

Munshaw Abhishek H.

Certificate

This is to certify that the Major Project Part - II Report entitled **"Optimization of Multicast Routing For Multimedia Traffic Over MANETs"** submitted by **Munshaw Abhishek H. (Roll No : 10MICT22)**, towards the partial fulfillment of the requirements for degree of Master of Technology in Information and Communication Technology from Institute of Technology, Nirma University, Ahmedabad is the record of work carried out by him under my supervision and guidance. In my opinion, the submitted work has reached a level required for being accepted for examination. The results embodied in this major project, to the best of my knowledge, haven't been submitted to any other university or institution for award of any degree.

Prof. Priyanka Sharma
Guide & Associate Professor,
Institute of Technology,
Nirma University, Ahmedabad.

Prof. Gaurang Raval
PG Coordinator(ICT),
Institute of Technology,
Nirma University, Ahmedabad.

Prof. D. J. Patel
Professor and Head,
CSE Department,
Institute of Technology,
Nirma University, Ahmedabad.

Dr. K. Kotecha
Director,
Institute of Technology,
Nirma University,
Ahmedabad.

Acknowledgements

First of all I would like to express my deep gratitude towards my guide **Prof. Priyanka Sharma**, Associate Professor, Institute of Technology, Nirma University, Ahmedabad without whose constant guidance and support this work would not have been possible. Her extensive knowledge of the subject gave me in depth understanding of the topic and helped me get through various technical difficulties.

I would like to thank **Prof. Gaurang Raval** , PG Coordinator(ICT), Computer Engineering Department, Institute of Technology, Nirma University, Ahmedabad for providing constant support and encouragement throughout the Major Project.

A special thanks to **Prof. D. J. Patel** , Head of Department, Computer Engineering, Institute of Technology, Nirma University, Ahmedabad for providing healthy research environment and basic infrastructure.

It gives me immense pleasure to thank **Dr. Ketan Kotecha** , Director, Institute of Technology, Nirma University, Ahmedabad for his motivation throughout the Major Project.

I would also like to thank the Institution, all the faculty members in Department of Computer Science and my colleagues without whom this projet would not have been possible.

Last but not the least I would like to thank God, family members and friends for their patience and unmentionable support throughout the Major Project.

- Munshaw Abhishek H.

10MICT22

Abstract

Mobile Ad-Hoc Network (MANET) is a network formed by the mobile devices without any preinstalled infrastructure. MANET provides a cheap and instantaneous solution to share data among mobile nodes. Nowadays, data hugely comprises of multimedia which is sent to many receivers by a single source. In such scenario when unicast is used, same data is sent as many times as there are receivers, making inept use of network resources. Multicast provides a solution to this problem. It duplicates a data packet only when necessary. But still there is a scope of improvement in current multicast routing protocols. In this work, a cluster based multicast ad hoc on demand vector protocol is proposed which provides an improvement in throughput and reduces control overhead in dense network making it highly scalable.

Keywords: MANET, Clustering, Multicasting

Contents

List of Figures

List of Tables

Abbreviations

AAMRP ..... Ant based adaptive multicast routing for mobile ad hoc network
ABAM .... On demand associativity based multicast routing for Ad hoc mobile networks
AMRIS ..... Ad Hoc multicast routing protocol utilizing increasing id-numbers
AMRoute ..... Ad-Hoc multicast routing protocol
AODV ..... Ad hoc On-Demand Distance Vector
AQM ..... Ad-Hoc quality of service multicast routing
AQOR ..... Ad-Hoc QoS on demand routing in mobile ad hoc network
ASTM ..... Adaptive shared tree multicast in mobile wireless networks
BEMR ..... Bandwidth efficient multicast routing for multihop, Ad Hoc wireless networks
CBMMRP ..... Core Based Multipath Multicast Routing Protocol
CBR ..... Constant Bit Rate
CBRP ..... Cluster Based Routing Protocol
CDMA ..... Code Division Multiple Access
CEDAR ..... Core Extraction Distributed Ad-Hoc routing
CM ..... Continuous Media
CMAODV ..... Cluster Based Multicast Ad hoc On-Demand Distance Vector
DCMP ..... Dynamic core based multicast routing protocol for ad hoc wireless networks
DDMR ..... Differential Destination Multicast Routing Protocol
DM ..... Discrete Media
DMAC ..... Distributed Mobility Adaptive Algorithm
DVMRP ..... Distance Vector Multicast Routing Protocol
FGMP .... Forwarding Group Multicast Protocol for Multihop, Mobile wireless Networks
FTP ..... File Transfer Protocol
Gbps ..... Giga bits per second
GDB ..... GNU debugger
GDMAC ..... Generalized Distributed Mobility Adaptive Algorithm
GIF ..... Graphic Interchange Format
GloMoSim ..... Global Mobile Information System Simulator
GUI ..... Graphical User Interface
HC ..... Highest Connectivity
HQMRP ..... Hierarchical QoS Multicast Routing Protocol
HTTP ..... Hyper Text Transfer Protocol
IP ..... Internet Protocol
ITAMAR ..... Independent-Tree Ad hoc Multicast Routing
JPEG ..... Joint Photographics Expert Group
Kb ..... Kilo bits

Kbps ..... Kilo bits per second
LID ..... Lowest ID
LTM ..... A Lantern-Tree based QoS Multicast Protocol
MAC ..... Media Access Control
MANET ..... Mobile Ad-Hoc Network
MANHSI ..... Novel Multicast Routing Protocol for Mobile Ad Hoc Networks
MANSI ..... Ad-Hoc multicast routing algorithm with swarm intelligence
MAODV ..... Multicast Ad hoc On-Demand Distance Vector
Mbps ..... Mega bits per second
MCEDAR ..... Multicast Core Extraction Distributed Ad-Hoc routing
MOBIC ..... Mobility Metric Based Algorithm
MZRP ..... Mulitcast Zone Routing Protocol in Wireless Mobile Ad Hoc Networks
NAMP ..... Neighbour Aware Multicast Routing Protocol for Mobile Ad Hoc Networks
NRT ..... Non real Time
NS ..... Network Simulator
NSMP ..... Neighbor Supporting Ad hoc Multicast Routing Protocol
ODMRP ..... On-Demand Multicast Routing Protocol
OPHMR .... An Optimized Polymorphic Hybrid Multicast Routing Protocol for MANET
OPNET ..... Optimized Network Engineering Tools
OTCL ..... Objective Tool Command Language
PARSEC ..... Parallel Simulation Enviroment for Complex Systems
PNG ..... Portable Network Graphics
QoS ..... Quality of Service
RFC ..... Request For Comments
RP ..... Rendezvous Point
RT ..... Real Time
RTI ..... Real Time Intolerant
RTT ..... Real Time Tolerant
SMTP ..... Simple Mail Transfer Protocol
TCL ..... Tool Command Language
TDMA ..... Time Division Multiple Access
WBCA ..... Weight Based Clustering Algorithm
WCA ..... Weighted Clustering Algorithm

Chapter 1

Introduction

1.1 General Overview

The widespread use of mobile and hand held computing devices increases the popularity of mobile ad hoc networks, which are self-organizing communication groups formed impromptu by wireless mobile hosts. They make their administrative decisions in a distributed manner without any centralized control. They are free from the boundaries of any pre-existing infrastructure and can be deployed anytime, anywhere. The nodes in a mobile ad hoc network move arbitrarily. Thus, the network topology changes frequently and unpredictably. The routing functionality possessed by the nodes enables them to communicate with each other through multihop paths made of intermediate nodes that relay the packets from the source towards the destination, even if these reside beyond the transmission range of each other. Due to their quick and economically less demanding deployment, mobile ad hoc networks are considered for many commercial applications, including home networks, nomadic computing, wireless local area, mesh or sensor networks and an increasing number of collaborative and distributed applications such as short-term communication for emergency operations, search and rescue, disaster relief, public events, game playing and temporary offices. The requirement of a temporary network for instantaneous communication among a group of people makes mobile ad hoc networks an excellent solution for these cases. A major factor that favors them for such tasks is the self-configuration ability of the system with minimal overhead.

There are many applications of mobile ad hoc networks that involve point-to multipoint or multipoint-to-multipoint communication patterns, which makes the efficient support of group communications a critical issue. The multicast communications model can facilitate effective and collaborative communication among groups. Multicast routing is a promising technique to provide a subset of network nodes with the group oriented service they demand while not jeopardizing the resource requirements of others. This is important for mobile ad hoc networks. The advantage of multicast routing is that packets are only replicated when it is necessary to reach two or more receivers on disjoint paths. This way, bandwidth consumption, router processing and delivery delay can be minimized. In wireless networks, it is particularly important to reduce transmission overhead. Thus, multicast routing can improve wireless link efficiency by exploiting the inherent broadcast property of the wireless medium. Combining the features of mobile ad hoc networks with the usefulness of multicast routing, a number of group oriented applications with close collaborative efforts can be realized. However, node mobility, with constraints of delay, loss and bandwidth, makes multicast routing very challenging.[1]

In this work, the protocol has been proposed which tries to improve the throughout and reduce the control overhead. The protocol addresses this issue by forming the clusters in the network and changing the way the control packets are flooded in multicast ad hoc on demand protocol (MAODV). Clustering is used as it limits the flooding of control packets in the network thus making a more efficient use of bandwidth. The routing protocol MAODV has been selected for routing of the packets as it is an on demand protocol and forms a tree structure for multicast routing. The on demand nature of the protocol sends control packets only when necessary and the tree structure sends data only to the nodes which are members of the tree in comparison to the mesh architecture which floods the data packets to send them to the receiving nodes.

1.2 Motivation

According to [2],there will be nearly one mobile device per capita by 2015. Global mobile data traffic will increase 26-fold between 2010 and 2015.Two-thirds of the world's mobile data traffic will be video by 2015.

Looking at the increase in the generation of multimedia data along with increase in number of mobile devices, the demand to transfer multimedia data between mobile devices will increase. To transfer the data between two devices, a network is required. Mobile ad hoc network provides a cheap, flexible and fast solution to create this network.

When same multimedia data has to be sent from one device to many, multicast routing provides an efficient way of transmitting data.The use of multicasting within MANETs has many benefits. It can reduce the cost of communication and improve the efficiency of the wireless channel when sending multiple copies of the same data by exploiting the inherent broadcasting properties of wireless transmission. But still there is a scope of improvement in current protocols specifically in providing better throughput for transmission of multimedia traffic.

1.3 Scope of the work

The scope of the work is to propose a multicast routing protocol for transmission of multimedia traffic over MANETs. The protocol will try to improve the throughput and reduce the control overhead.

1.4 Thesis Organization

Rest of the thesis is organized as follow:

Chapter 2 *(Literature Survey)*: This chapter initially looks at various types of multimedia traffic and their bandwidth requirements. It then discusses various multicast routing protocols that are currently present for MANETs and they are classified according to there

various features. The chapter also includes the survey of different one hop clustering algorithms along with their strength and weaknesses.

Chapter 3 *(Protocols Supporting Quality of Service)*: This chapter covers the survey of multicast routing protocols over MANET which support quality of service and then certain protocols which were felt as important are discussed along with their advantages and disadvantages.

Chapter 4 *(Proposed Protocol)*: This chapter discusses in detail the protocol cluster based multicast ad hoc on demand protocol which is proposed to meet the aim of the thesis.

Chapter 5 *(Simulation)*: This chapter includes the details of the implementation of proposed protocol. It comprises of the comparison of results obtained after simulating CMAODV, MAODV and AODV protocols for various scenarios.

Chapter 6 *(Conclusion and Future Scope of Work)*: Concluding remarks and future work is discussed in this chapter.

Chapter 2

Literature Survey

2.1 Introduction to MANET

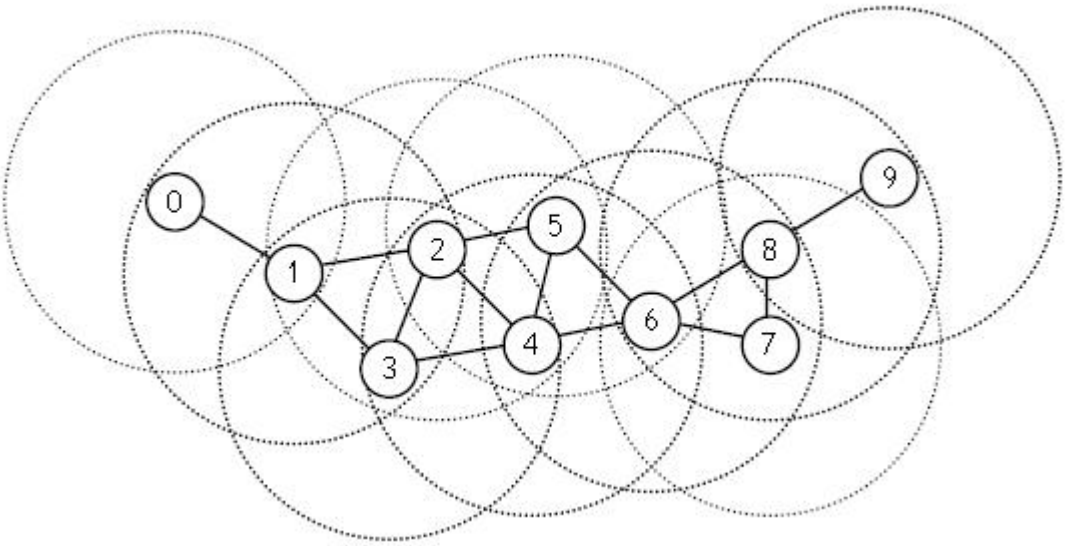

Figure 2.1: MANET[1]

Mobile ad hoc networks (MANETs) comprise either fixed or mobile nodes connected wirelessly without the support of any fixed infrastructure or central administration. The nodes are self-organized and can be deployed on the fly anywhere, any time to support a particular purpose. Two nodes can communicate if they are within each other's transmission range; otherwise, intermediate nodes can serve as relays (routers) if they are out of range (multihop routing). These networks have several salient features: rapid deployment,

robustness, flexibility, inherent mobility support, highly dynamic network topology (device mobility, changing properties of the wireless channel, that is, fading, multipath propagation, and partitioning and merging of ad hoc networks are possible), the limited battery power of mobile devices, limited capacity, and asymmetric/unidirectional links. MANETs are envisioned to support advanced applications such as military operations (formations of soldiers, tanks, planes), civil applications (e.g., audio and video conferencing, sport events, telematics applications (traffic)), disaster situations (e.g., emergency and rescue operations, national crises, earthquakes, fires, floods), and integration with cellular systems.[3] Figure 2.1 shows an example for a mobile ad hoc network of ten nodes.

Following are the salient features of MANETs:-

- Rapid deployment
- Robustness
- Flexibility
- Mobility support
- Highly dynamic network topology
- Limited battery power of mobile devices
- Limited bandwidth

2.2 Multimedia Traffic

The term multimedia refers to diverse classes of media employed to represent information. Multimedia traffic refers to the transmission of data representing diverse media over communication networks. Media is classified into three groups: (i) text, (ii) visuals, and (iii)sound. As illustrated in Figure 2.2, the symbolic textual material may include not only the traditional unformatted plain text, but also formatted text with numerous control characters, mathematical expressions, phonetic transcription of speech, music scores, and other symbolic representations such as hypertext. The visual material may include line

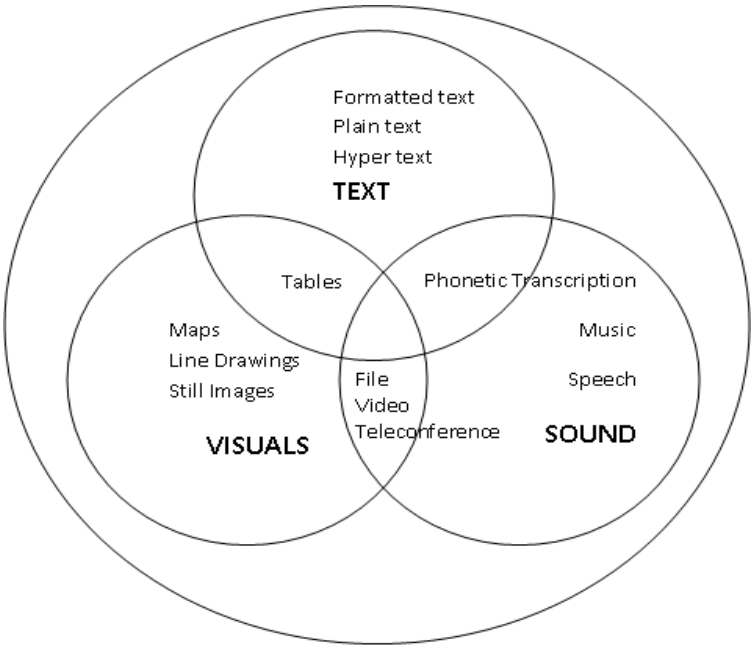

Figure 2.2: Diverse classes of media[4]

drawings, maps, gray-scale or colored images and photographs, as well as animation, simulation, virtual reality objects, video- and tele-conferencing. The sound material may include telephone/broadcast-quality speech to represent voice, wideband audio for music reproduction, and recordings of sounds such as electrocardiograms or other biomedical signals.[4]

2.2.1 Classification of Media Types

From a networking perspective, all media types can be classified as either Real-Time (RT) or Non Real-Time (NRT), as shown in Figure 2.3. RT media types require either hard or soft bounds on the end-toend packet delay/jitter, while NRT media types, like text and image files, do not have any strict delay constraints, but may have rigid constraints on error. The RT media types are further classified as Discrete media (DM) or Continuous media (CM), depending on whether the data is transmitted in discrete quantum as a file or message, or continuously as a stream of messages with inter-message dependency. The real time discrete type of media has recently gained high popularity because of ubiquitous applications like MSN/Yahoo messengers (which are error intolerant) and instant messaging services like stock quote updates (which are error tolerant). The RT continuous type of

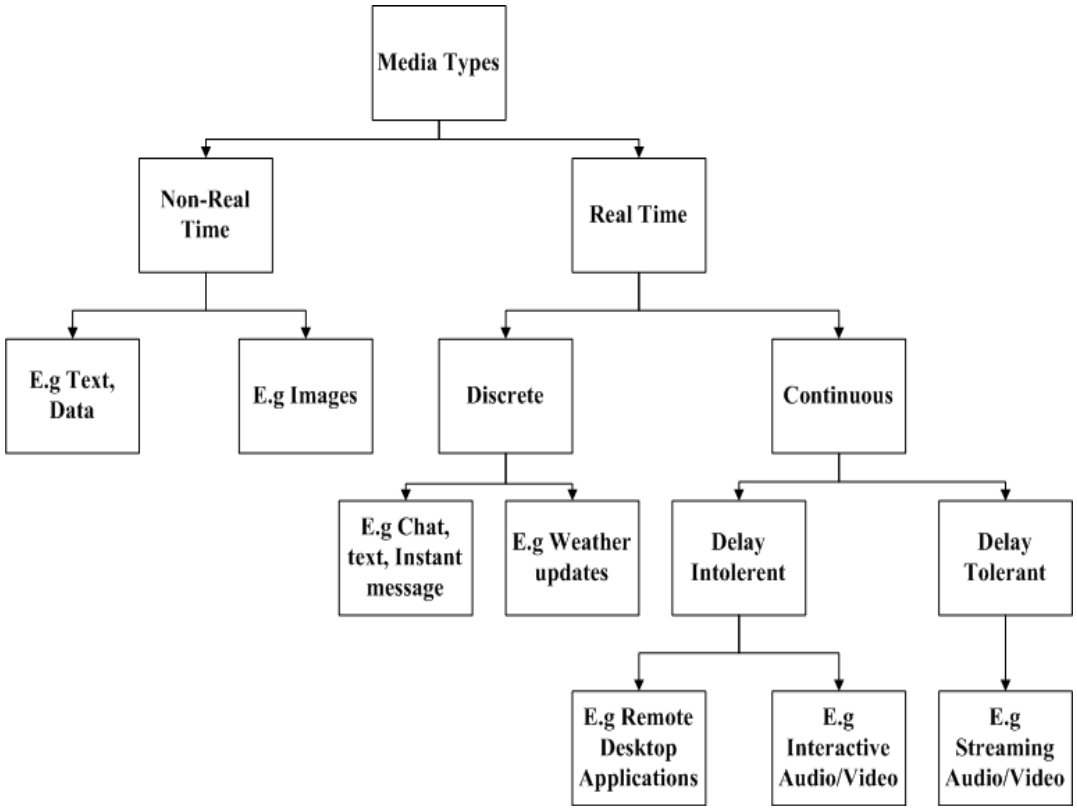

Figure 2.3: Classification[4]

media can further be classified as delay tolerant or delay intolerant. The term delay tolerant to signify that such media type can tolerate higher amounts of delay than their delay intolerant counterparts, without significant performance degradation. Examples of RT, continuous, and delay-intolerant media are audio and video streams used in audio or video conference systems, and remote desktop applications. Streaming audio/video media, used in applications like Internet webcast, are examples of delay-tolerant media types. Their delay-dependency is significantly diminished by having an adaptive buffer at the receiver that downloads and stores a certain portion of the media stream before starting playout.

2.2.2 Text

Text is the most popular of all the media types. It is distributed over the Internet in many forms including files or messages using different transfer protocols such as FTP (File Transfer Protocol: used to transfer binary and ASCII files over the Internet), HTTP (Hyper Text Transfer Protocol: used to transmit HTML pages) or SMTP (Simple Mail Transfer

Protocol: Used for exchanging e-mails). Text is represented in binary as either 7-bit US-ASCII, 8-bit ISO-8859, 16-bit Unicode or 32-bit ISO 10646 character sets, depending on the language of choice and the country of origin. Bandwidth requirements of text media mainly depend on its size, which can be easily reduced using common compression schemes such as Shannon-Fano Coding, Huffman Coding, LZW, Unix Compress, etc.[5]

2.2.3 Graphics

This includes static media types like digital images and dynamic media types like flash presentations. An uncompressed, digitally encoded image consists of an array of pixels, with each pixel encoded in a number of bits to represent luminance and color[5]. Compared to text or digital audio, digital images tend to be large in size. Thus some form of compression schemes are always used that cash on the property of high spatial redundancy in digital images. Some popular compression schemes are Graphics Interchange Format (GIF), Portable Network Graphics (PNG), Joint Photographic Experts Group(JPEG), JPEG2000,etc. Most modern image compression schemes are progressive, which have important implications to transmission over the communication networks. Images are error-tolerant and can sustain packet loss, provided the application used to render them knows how to handle lost packets. Moreover images, like text files, do not have any real-time constraints.[5]

2.2.4 Audio

Audio media is sound/speech converted into digital form using sampling and quantization. Digitized audio media is transmitted as a stream of discrete packets over the network. The bandwidth requirements of digitized audio depend on its dynamic range and/or spectrum. A number of compression schemes are Pulse code Modulation (G.711), GSM, MPEG layer III (MP3), etc. The audio media type has loose requirements on packet loss/errors (or loss/error-tolerant), in the sense that it can tolerate up to 1 to 2 percent packet loss/error without much degradation. Today, most multimedia applications that use audio, have in-built mechanisms to deal with the lost packets using advanced interpolation techniques.

The real-time requirements of audio strictly depend on the expected interactivity between

the involved parties. Some applications like Internet-Telephony, which involves two-way communication, are highly interactive and require shorter response times. The audio media, in this case, requires strong bounds on end-to-end packet delay/jitter to be of acceptable/decipherable quality. Applications that use this media type are called Real-Time Intolerant (RTI) applications. In most RTI applications the end-toend delay must be limited to 200 msec to get an acceptable performance. Other applications like Internet webcast, which involves one-way communication, have relatively low interactivity. Interactivity, in this case, is limited to commands that allow the user to change radio channels (say), which can tolerate higher response times. Consequently, it requires weaker bounds on delay/jitter and the applications that use such kind of media are termed as Real-Time Tolerant (RTT) applications. Streaming Audio is also used to refer to this media type.[5]

2.2.5 Video

Video is a sequence of images/frames displayed at a certain rate, e.g., 24 or 30 frames/second. Digitized video, like digitized audio, is also transmitted as a stream of discrete packets over the network. The bandwidth requirements for digitized video depend on the spatial redundancy present within every frame, as well as the temporal redundancy present in consecutive frames. Both these redundancies can be exploited to achieve efficient compression of video data. Some common compression schemes that are used in video are MPEG-I, MPEG-II, MPEG-IV, etc. The error and real-time requirements of video media are similar to the audio media type.

2.2.6 Bandwidth Requirement

Table 2.1: Bandwidth Requirement of Audio Sources[4]

Audio Source	**Sampling Rate**	**Bits/Sample**	**Bit Rate**
Telephone Grade Voice	8000 samples/sec	12	96 Kbps
Wideband Speech	1600 samples/sec	14	224 Kbps
Wideband Audio Two Channels	44.1 samples/sec	16 per channel	1.412 Mbps for both channels

Table 2.2: Bandwidth Requirement of Uncompressed Image Sources[4]

image Source	**Pixels**	**Bits/pixel**	**Bit Rate**
Color Image	512x512	24	6.3 Mbps
CCIR TV	720x576x30	24	300 Mbps
HDTV	1280x720x60	24	1.327 Gbps

2.3 Introduction to Multicasting

Multicasting plays a crucial role in MANETs to support the above applications. It involves the transmission of a datagram to a group of zero or more hosts identified by a single destination address, and so is intended for grouporiented computing. A multicast datagram is delivered to all members of its destination host group with the same best effort reliability as regular unicast IP datagrams, that is, the datagram is not guaranteed to arrive intact at the destinations of all members of the group, or in the same order relative to other datagrams [6]. The use of multicasting within MANETs has many benefits. It can reduce the cost of communication and improve the efficiency of the wireless channel when sending multiple copies of the same data by exploiting the inherent broadcasting properties of wireless transmission. Instead of sending data via multiple unicasts,multicasting minimizes channel capacity consumption, sender and router processing, energy consumption, and delivery delay, which are considered important MANET factors. In addition,multicasting provides a simple yet robust communication method whereby a receiver's individual address remains unknown to the transmitter or changeable in a transparent manner by the transmitter. Multicasting in MANETs is much more complex than in wired networks and faces several

challenges. Multicast group members move, which precludes the use of a fixed infrastructure multicast topology, wireless channel characteristics can vary over time, and there are restrictions on node energy and capacity. The issues associated with these protocols that should be taken into consideration are:-

- Topology, Mobility, and Robustness
- Capacity and Efficiency
- Energy Consumption
- Quality of Service and Resource Management
- Security and Reliability
- Scalability

2.4 Classification of Multicast Routing Protocols

2.4.1 Overview

The protocols can be classified based on topology, initialization, topology maintenance, core and coreless approach, dependency on unicast routing protocols. Topology is defined as how multicast session's nodes are arranged in a known topology shape. Multicast routing protocols can be divided into two main categories: Tree-based protocols and Mesh-based protocols. Another view point of multicast routing protocols classification is which node will initiate the multicast session that is will it be the client or the source. Under topology maintenance, soft state approach and hard state approach are different in the way of maintaining connectivity between multicast session nodes. In soft state approach, periodic transmission of control packets is used to keep topology connections between nodes. But in hard state approach, control packets will be transmitted only when link break occur. The difference between core and coreless is that in coreless approach all nodes take part in maintaining the network while in core approach a single node or some number of nodes will be responsible for the control of the multicast session.

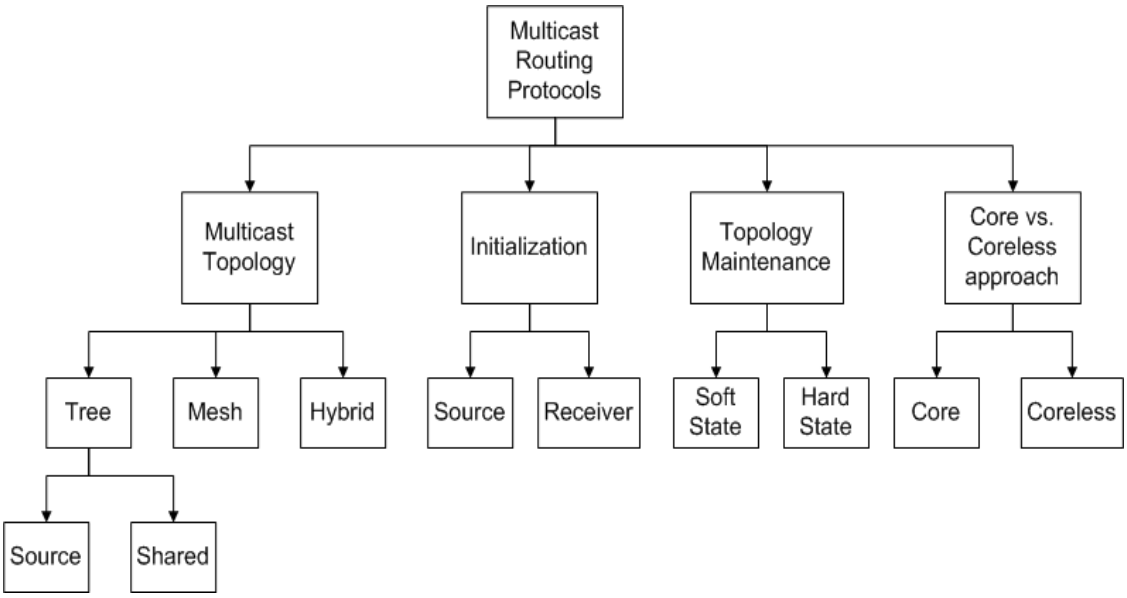

Figure 2.4: Classification of protocols[7]

2.4.2 Classification based on topology

Topology is defined as how multicast session's nodes are arranged in a known topology shape. Multicast routing protocols can be divided into two main categories: Tree-based protocols and Mesh-based protocols. Multicast tree-based concept was appeared firstly in several wired networks. It means that, only a single shortest path (i.e. route) must be established between source-receiver pair. Although multicast tree-based routing protocols are efficient and satisfy scalability issue, they have several drawbacks in ad hoc wireless networks due to mobile nature of nodes that participate during multicast session.

Tree-based proposals are also divided into two subcategories: source-based tree and shared-based tree approaches. In source-based tree approaches, each source builds its single tree. In shared-based tree approaches, all sources share only a single tree that is controlled only by one or more specific nodes.

There are other proposals that try to use source-based and shared-based tree topologies as Adaptive shared tree multicast routing protocol. Each source builds its own tree and multiple sources share one that is rooted at a rendezvous point (RP).

There are two basic drawbacks of tree-based protocols. The first drawback is that the tree structure is fragile because of unpredictable topology changes due to mobility of nodes. The

second drawback is tree reconstruction delay. So, a new topology concept called Mesh-based was established. It has the possibility to provide multiple paths between any source-receiver pair.
Ad Hoc Multicast Routing Protocol (AMRoute) and Multicast Core- Extraction Distributed Ad Hoc Routing (MCEDAR)[8] try to gather both treebased and mesh-based advantages in one protocol by constructing shared-based tree topology over mesh topology. These protocols are called Hybrid topology protocols.

2.4.3 Classification based on session initialization

Another view point of multicast routing protocols classification is which node will initiate the multicast session. MZRP, ABAM, AMRIS, ODMRP, AQOR, DCMP, and NSMP give this responsibility to source but other protocols give it to receiver as BEMRP, ASTM, DDM, PLBM and MAODV. Adaptive Shared Tree Multicast Protocol, and FGMP. CAMP, AMRoute, and MCEDAR[8] don't care who will initiate a session.Generally, there are no pros or cons about who have the rights to initiate the multicast sessions except in the case of receiver initialization because this approach has many benefits if number of sources will be greater than number of receivers.

2.4.4 Classification based on topology maintenance

Soft state approach and hard state approach are different in the way of maintaining connectivity between multicast session nodes. In soft state approach, periodic transmission of control packets is used to keep topology connections between nodes. But in hard state approach, control packets will be transmitted only when links break occur. Although soft state approach causes an inefficient utilization of limited bandwidth due to large number of control packets that flooded over network, it offers a high packet delivery ratio. In the other side, hard state approach saves bandwidth but it suffers from low packet delivery ratio.

2.4.5 Classification based on core/coreless

Multicast routing protocols in MANETs can be classified according to who is responsible of providing network information. In coreless or distributed approach, all nodes in multicast

session cooperate with each others to build an initial topology whether it is mesh based or tree based and also to maintain that constructed topology.

Another approach is core based or centralized control. It means that the responsibility of managing, storing, and controlling all information of multicast session is performed by one or some of multicast nodes. Core-based approach can be divided into two different approaches. They are dynamic core and static core approaches. Dynamic core approach means that if the current core node is failed, member nodes of a multicast session elect or search for another one to be a new core as DCMP and AMRoute. Static core approach is the contrast of dynamic core concept. It means that a group of nodes or just one node controls all network tasks. Network will be dropped due to any failure of these core nodes. Generally, core based routing protocols is used to reduce control overhead messages and to make a best utilization of bandwidth; however, they have a risk of a single node failure. Coreless based protocols solve the last problem but large overheads resulting of periodic announcements.

Table 2.3: Comparison of Multicast Routing Protocols

Sr. No.	Protocols	Type	Topology	Initiation
1	On demand associativity based multicast routing for Ad hoc mobile networks (ABAM)[9]	On Demand	Tree	Source
2	Ad Hoc multicast routing protocol utilizing increasing id-numbers (AMRIS)[10]	On Demand	Shared tree	Source
3	Ad-Hoc multicast routing protocol (AMRoute)[11]	Proactive	Shared tree (Hybrid)	Source
4	Adaptive shared tree multicast in mobile wireless networks (ASTM)[12]	Proactive	Shared tree	Receiver
5	Bandwidth efficient multicast routing for multihop, Ad Hoc wireless networks (BEMR)[13]	On Demand	Source tree	Receiver
6	Dynamic core based multicast routing protocol for ad hoc wireless networks (DCMP)[14]	On Demand	Shared Mesh	Source
7	Differential Destination Multicast A MANET Multicast Routing Protocol for Small Groups" (DDMR)[15]	Proactive	Tree	Source

Continued on next page

Table 2.3 – continued from previous page

Sr. No.	Protocols	Type	Topology	Initiation
8	Distance Vector Multicast Routing Protocol (DVMRP)	Proactive	Tree	Source
9	Forwarding Group Multicast Protocol for Multihop, Mobile wireless Networks (FGMP)[16]	Proactive	Mesh	Receiver
10	Mulitcast Zone Routing Protocol in Wireless Mobile Ad Hoc Networks (MZRP)[17]	On Demand	Mesh	Source
11	Neighbor Supporting Ad hoc Multicast Routing Protocol (NSMP)[18]	Proactive	Mesh	Source
12	On-Demand Multicast Routing Protocol (ODMRP)[19]	On Demand	Mesh	Source
13	Multicast Ad hoc On Demand Vector Routing Protocol[20]	On Demand	Tree	Receiver
14	Neighbour Aware Multicast Routing Protocol for Mobile Ad Hoc Networks (NAMP)[21]	Hybrid	Tree	Source

Continued on next page

Table 2.3 – continued from previous page

Sr. No.	Protocols	Type	Topology	Initiation
15	Novel Multicast Routing Protocol for Mobile Ad Hoc Networks (MANHSI)[22]	On Demand	Mesh	Source
16	Ant based adaptive multicast routing for mobile ad hoc network (AAMRP)[23]	Proactive	Hybrid	Source
17	Ad-Hoc multicast routing algorithm with swarm intelligence (MANSI)[24]	On Demand	Mesh	Source

2.5 Clustering Techniques

Clustering can be defined as dividing the network into various non over lapping or partially overlapping group of nodes. This group of nodes are considered as one cluster. Each cluster selects one cluster head. This cluster head is given the job to communicate with other cluster heads and act as a coordinator for other nodes of the cluster. The clusters are mostly formed with respect t proximity of the nodes. For example, in a one hop clustering, the cluster head is selected in such a way that all the other nodes can reach it in a single hop. Two nodes are considered as one hop neighbor if they can directly communicate with each other. The cluster heads use gateway nodes to communicate with other cluster heads. Gateways are the nodes which are present in the overlapping part of two clusters hence they can communicate with both cluster heads, thus forming a link between two clusters. Clusters are also formed with more than one hop which are known as k-hop cluster where k is greater than one. But the There are various clustering algorithms that are based on location, mobility, neighbor, power and artificial intelligence. These algorithms are comprehensively presented in various survey papers as [25] [26] [27]. As the one hop clustering provides shorter paths than the

multi hop clusters [27], only one clustering algorithms are looked into. Following table 2.4 presents various clustering algorithms and their strengths and weaknesses as presented in [27].

Table 2.4: Comparison of Clustering Techniques

Algorithm	**Method**	**Strength**	**Weakness**
LID	Lowest Identity based node among the neighbors	Simple;Faster cluster setup	Biasness of low ID nodes to become heads
HC	Highest Connectivity based node among the neighbors	Simple; Minimum cluster heads at high transmission range	Highly unstable; A single change in connectivity may lead to reelection; Avg. cluster changes by the nodes are very high
MOBIC	Lowest aggregate mobility of a node w.r.t. its neighbors. Received signal strength is the key parameter to measure the relative mobility	Highly stable as two heads are allowed to be neighbors for Cluster Contention Interval (CCI) period	Requires large cluster setup time; Very high reaffiliation rate increases computation and communication overhead; May increase routing delay due to more cluster heads
Continued on next page			

Table 2.4 – continued from previous page

Algorithm	Method	Strength	Weakness
DMAC	Lowest weighted node within its local topology. Weight may be the node mobility/transmission power	Distributive and mobility adaptive; Does not freeze node mobility during setup	Does not allow two heads to be neighbors increasing reaffiliation and reelection rate to some extend
WCA	Lowest weighted node in the entire network. Weight is the function of degree of connectivity, mobility, available battery power and utilized transmission power of individual nodes	Reelection is done on-demand, So better cluster stability; Reaffiliation is very low reducing the overhead	Many parameters are considered for setup, this increases setup delay; Selecting the global minima in a distributed fashion needs more message exchanges between the nodes and channel bandwidth consumption is increased
GDMAC	Enhancement over DMAC with the introduction of threshold parameter H and K	Highly stable as K heads are allowed to be neighbors; Unlike DMAC a node raffiliates to another head only when the weight difference of heads is more than H	Routing delay may be increased as the number of cluster heads or the routing points are more.

Continued on next page

Table 2.4 – continued from previous page

Algorithm	**Method**	**Strength**	**Weakness**
WBCA	Lowest weighted node in the entire network. Weight is the function of mean connectivity degree of the node and its consumed battery power	Less no. of parameters is considered than WCA; Re-election is done on-demand; So better cluster stability; Reaffiliation is very low reducing the overhead	Computation cost is more as calculation of mean connectivity degree of a node needs the degree of connectivity of neighbor nodes. Thus cluster setup is delayed; Getting the global minima in a distributed fashion needs more message exchanges between the nodes and channel bandwidth consumption is increased

Chapter 3

Protocols Supporting Quality of Service

3.1 Introduction

The increasing popularity of using multimedia and real time in different potential commercial applications in MANETs makes it a logical step to support Quality of Service (QoS) over wireless network. QoS is defined as a guarantee given by the network to give a performance level to satisfy a set of predefined service parameters requested by the network user, these parameters including bandwidth, end-to-end delay, delay-jitter and packet to loss ratio. QoS support is tightly related to resource allocation and reservation to satisfy the application requirement[28]. The role of QoS routing protocol is to find a suitable loop-free paths that have enough resources available from the source to the destination to satisfy the desired QoS requirements. There are two QoS strategies used to search for routes and maintain the information state in QoS multicast routing. The first is source routing, where a feasible route is computed locally at the source node, which is not scalable for large area networks. The second one is distributed routing, the path computation is distributed over the intermediate nodes which makes it more scalable than source routing [29]. It is not easy task to combine QoS to multicast ad hoc networks. So, supporting QoS for multicast protocols has to be designed in a way different from unicast protocols. The difference is that in unicast QoS protocols the resource reservation is done between a source and a des-

tination. While multicast QoS routing protocols should provide suitable QoS paths to all destinations of the multicast group. Also, the heterogeneous nature of paths to the destinations adds extra challenges to the design of QoS multicasting protocols[30]. Table 3.1 summarizes various multicast routing protocols which provide quality of service.

Table 3.1: Comparison of Multicast Routing Protocols with QoS

Sr. No.	**Protocols**	**Type**	**Topology**	**Initiation**
1	Ad-Hoc quality of service multicast routing (AQM)[31]	Proactive	Tree	Source
2	Ad-Hoc QoS on demand routing in mobile ad hoc network (AQOR)[32]	On Demand	N/A	Source
3	Independent-Tree Ad hoc Multicast Routing (ITAMAR)[33]	Proactive	Tree	Source
4	A Lantern-Tree based QoS Multicast Protocol (LTM)[34]	On Demand	Lantern Tree	Source
5	An Optimized Polymorphic Hybrid Multicast Routing Protocol for MANET (OPHMR)[35]	Hybrid	Mesh	Source
6	Hierarchical QoS Multicast Routing Protocol (HQMRP)[36]	On Demand	Tree	Receiver
7	A hexagonal-tree TDMA-based QoS multicasting protocol[37]	Proactive	Tree	Source
8	Multicast Core Extraction Distributed Ad-Hoc routing (MCEDAR)[8]	On Demand	Hybrid	Source

3.2 Related Protocols

3.2.1 Lantern tree based QoS On-Demand Multicast Protocol[34]

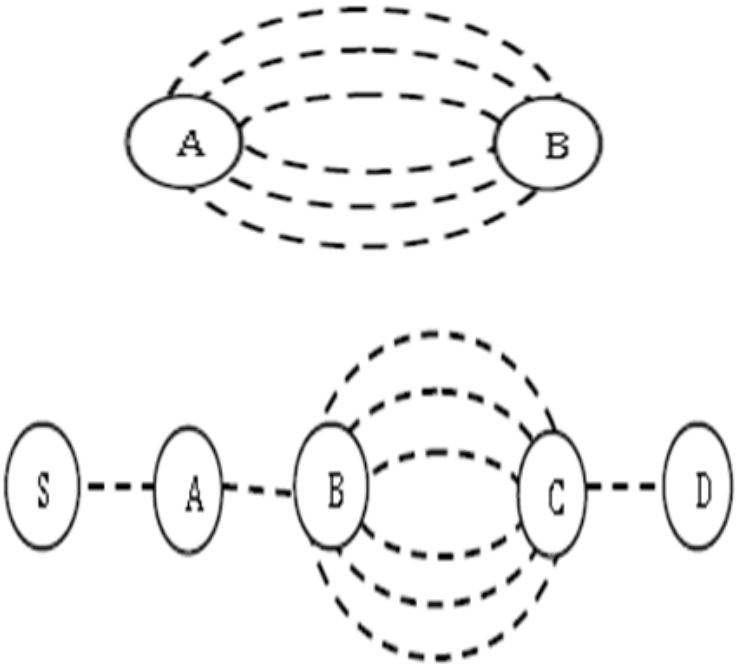

Figure 3.1: Lantern Tree[34]

The Lantern-Tree-based QoS On-Demand Multicast Protocol (LTM) first searches for lantern paths from a source to a set of destination nodes and then merges them together to construct the lantern tree. The QoS path is a path which satisfies end-to-end bandwidth requirement under CDMA-over-TDMA channel model at the MAC layer. Available bandwidth in this model is measured in terms of the number of free time slots. LTM defined a lantern as a path or more sub-paths that exists between two nodes to satisfy the required bandwidth.To identify a lantern, each node periodically maintains information about all one hop and two hop neighboring nodes with the free time slots of these nodes. After collecting the link state of all the nodes, the lantern is identified. Lantern tree construction is started after time slot reservation is finished. The source node floods the network with lantern-path request packet to check if one or more lantern exists in the source. If a lantern exists, the lantern path is constructed. This process repeated until a possible lantern path arrives at a destination node. An Ack message is sent to acknowledge the two parties of the

lantern about the successful transmission. The construction of the multicast tree is done by replacing all the spiral-fat-tree paths of the spiral-fat-tree protocol into lantern-path to provide QoS capability.

Advantages

- Using multiple paths provide high aggregate of the network bandwidth.
- High access rates and high stability due to using multiple path lantern trees.
- Efficient utilization of the network bandwidth especially when the bandwidth is limited.

Disadvantages

- Increasing the number of links increase the contention at the MAC layer.
- It takes a long time to find all the paths and shares the time slots between the neighboring nodes.
- Nodes have to process and store more information about sub-paths which wastes the resources.

3.2.2 MCEDAR-Multicast Core-Extraction Distributed Ad Hoc Routing[8]

MCEDAR [10] is a multicast extension to the CEDAR [11] architecture which provides the robustness of mesh structures and the efficiency of tree structures. MCEDAR uses a mesh as the underlying infrastructure, but the data forwarding occurs only on a sender-rooted tree. MCEDAR is particularly suitable for situations where multiple groups coexist in a MANET. At first, MCEDAR partitions the network into disjoint clusters. Each node exchanges a special beacon with its one hop neighbors to decide that it becomes a dominator or chooses a neighbor as its dominator. A dominator and those neighbors that have chosen it as a dominator form a cluster. A dominator then becomes a core node and issues a message to nearby core nodes for building virtual links between them. All the core nodes form a core graph. When a node intends to join a group, it delegates its dominating core node P to join the appropriate mgraph instead of itself. An mgraph is a subgraph of the core graph and is composed of those core nodes belonging to the same group. P joins the mgraph

by broadcasting a join message which contains a joinID. Only those members with smaller joinIDs reply an ACK message to P. Other nodes receiving the join message forward it to their nearby core nodes. An intermediate node Q only accepts at most R ACK messages where R is a robustness factor. Q then puts the nodes from which it receives the ACK message into its parent set and the nodes to which it forwards the ACK message into its child set. When a node has less than R/2 parents, it periodically issues new join messages to get more parents. When a data packet arrives at an mgraph member, the member only forwards the packet to those nearby member core nodes that it knows.

Advantages

- The underlying mesh structure is robust to high mobility.
- When multiple groups coexist, the core graph can work as a backbone and hence reduces the total control overhead for these groups.

Disadvantages

- High control overhead is incurred on the partitioning procedure.
- Since the data forwarding tree is built among core nodes (not actual group members), it has non-optimal paths.
- If a node leaves its dominating core node and attaches to another core node, it may miss interested packets.
- The failure of a core node affects those nodes that delegate it to join the group.

3.2.3 Hierarchical QoS Multicast Routing Protocol (HQMRP)[36]

A protocol called Hierarchical QoS Multicast Routing Protocol (HQMRP) is presented in [24]. It organizes the network into multiple domains, the 0-level represents the nodes, and several nodes form the first-level which contains at least one node and does not overlap with any first-level cluster. The upper levels are forming from grouping the down levels into domains. The clusters with the same level are connected using bridge nodes. This protocol consider the bandwidth and delay as QoS metrics and assume that each node measure periodically the delay of outgoing links and broadcast it to cluster members. Similarly, the

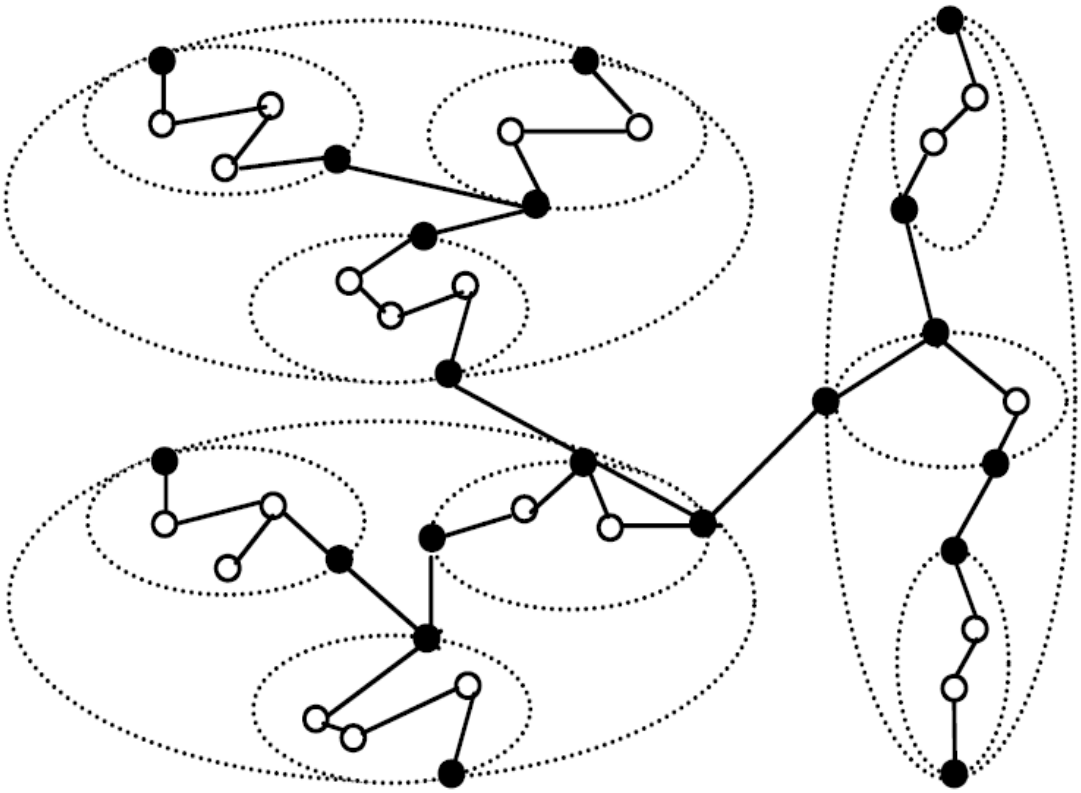

Figure 3.2: Hierarchical Topology[36]

bridge nodes measure the delay of the outgoing links and broadcast this information to bridge nodes in the same level. The multicast tree is constructed by sending a JOINReq msg with QoS metrics by the first member to its parent bridge node, and the first entry in the array is set to the address of this node. If the bridge node does not aware of the multicast tree, it adds itself to the array and forwards the request to its parent bridge node. If the multicast tree not found, the request will arrive to the bridge node at toplevel domain. The top bridge node sends multicast generation (MT generation) message towards the node. When the new member receives this message it reply by sending MT update msg to its parent bridge node to generate the tree. Each bridge node maintains the address of all on-tree nodes within the domain and bridge addresses of the lower level domains containing on-tree nodes. When a new node wants to join a multicast group, it sends a JOINReq msg to its parent bridge node. If the message arrives at a bridge node that is aware of the multicast tree, it forwards it to all on-tree nodes or bridge nodes of the sub-domains having on-tree nodes. Otherwise, the bridge node forwards the message to its parent bridge

node.When the JOINReq arrives at an on-tree node, the node initiates a SF message. This message is flooded towards the new node by sending it to some neighbors which in turn forward the message to their neighbors.

Advantages

- HQMRP is scalable for large area networks due to it hierarchy of domains.
- Reducing the message overhead by selecting fewer on-tree routers to flood towards the host router.
- Local nodes maintain local multicast routing information instead of global network states.

Disadvantages

- Using periodic messages to construct the Neighbor table introduce considerable control overhead.
- Using subscribing to handle changing the domain will requires excessive communication and message processing specially in high mobility networks.

3.2.4 A hexagonal-tree TDMA-based QoS multicasting protocol for wireless mobile ad hoc networks[37]

In Hexagonal-tree QoS multicast protocol proposes a new multicast tree structure, namely a hexagonaltree, to serve as the QoS multicasting tree, where the MAC sub-layer adopts the TDMA channel model. TDMA model is more practical and less costly than CDMA over TDMA model. But, using a TDMA model needs to face the challenge of radio interference problems. In this protocol, both the hidden-terminal and exposed-terminal problems are taken into consideration to possibly exploit the timeslot reuse capability. The hexagonal-based scheme offers a higher success rate for constructing the QoS multicast tree due to the use of the hexagonal-tree. A hexagonal-tree is a tree whose sub-path is a hexagonal-path. A hexagonal-path is a special two-path structure. To find the hexagonal path, The source node floods a hexagonalpath request packet into a MANET to check if there is a uni-path from source node which satisfies the required bandwidth b, then a hexagonal

sub-path is identified. If no uni-path exists, then node further checks to see if one or more hexagonal-block exist from the node. If a hexagonal-block exists which satisfies the required bandwidth b, then a hexagonal sub-path is constructed. If no hexagonal-block exists, then the node further checks to see if one or more hexagonal-branches exist from it. If a hexagonal-branch exists which satisfies the required bandwidth b, then it is constructed. If even above step fails, then the searching operation of a hexagonal-path is stopped and the procedure is exited.Each destination node replies to all possible hexagonalpaths to the source node. Many hexagonal-paths will be collected at the source node, and then a constructing hexagonal-tree operation is eventually performed at the source node.This protocol greatly improves the success rate by means of multi-path routing.

Advantages

- Multi paths ensure better success rate, robustness and the stability.
- While allocating the time slots, both hidden node and exposed node problems are addressed.

Disadvantages

- It is not highly scalable.
- The computation time for allocating the time slots increases with the increase in number of nodes in the path.

Chapter 4

Proposed Protocol

4.1 Introduction

The protocol, cluster based multicast ad hoc on demand vector protocol optimizes the performance of the current ad hoc on demand vector protocol with multicast with respect to control overhead and throughput. The protocol forms clusters to achieve the goal of higher throughput with lower control overhead. The protocol uses Lowest ID algorithm to choose the cluster heads and uses a multicast ad hoc on demand vector protocol for the routing of the packets. The protocol uses the LID algorithm for clustering as it is simple and forms clusters quickly with lower number of control packets and lesser amount of data to process and stored by the nodes. As the routing protocol, MAODV is chosen as the shared tree structure that is devised by it provides better bandwidth utilization than mesh architectures.

4.2 Clustering Mechanism

In this protocol, clustering has three steps.

a. Cluster Formation

b. Cluster Contention

c. Cluster Maintenance

4.2.1 Cluster Formation

This protocol uses the Lowest ID algorithm for the formation of clusters through cluster heads. Following are the steps that are followed while forming the clusters initially.

- First, all the nodes are assigned a distinct index number.
- All the nodes send a hello packet with one hope time to live along with their index number.
- The nodes save the details of their neighbors and then based on it decide whether they can be cluster head or not.
- If the node's index number is the lowest among all the one hop neighbors than it will assign itself as cluster head and send another hello packet with its index number and its status as cluster head.
- All the other nodes in the one hop vicinity of the cluster head change their status to cluster member.
- The nodes which are in vicinity to more than one cluster heads, become members of both the clusters and act as gateways.

Thus, after following above steps, all the nodes in the network are either cluster heads or cluster members and the cluster members who have more than one cluster head become gateway nodes.

4.2.2 Cluster Contention

The protocol does not allow two cluster heads to be one hop neighbors. Hence, following are the steps that will be followed when two cluster heads in the network are one hop neighbors.

- Both the cluster heads, receive the hello packets from each other and save the details.
- The cluster head whose index number is lower will remain a cluster head and will again send a hello packet which will state its status as cluster head.

- The node which has higher index number will change its status from cluster head to cluster member and also send a hello message stating the same so that the nodes which are its member can update their status.

4.2.3 Cluster Maintenance

Once the clusters are formed, due to mobile nature of the nodes, it is important to constantly be updated with status of the nodes in the network hence cluster maintenance is very important in mobile ad hoc network. Following are the steps taken to maintain the clusters in the network.

- Once all the clusters are formed, all the nodes periodically send a hello message along with their status.
- All the nodes update their neighbor information when new hello packets are received.
- If the node does not find any cluster head in its one hop neighborhood than it will declare itself as cluster head and send a hello packet stating that it is now cluster head.
- If node which has lower index number than the cluster head of a cluster which it is entering, then it will assign itself as a cluster head and the older cluster head will change its status to cluster member.
- When two cluster head come in each others one hop neighborhood, then the steps mentioned in section 4.2.2 are followed.
- If a node finds that it is a member of more than one cluster, than it will change its status to gateway and if a node which is a gateway finds that it is no more a member of more than one cluster than it will change its status simply to cluster member.

4.3 Multicast Routing

This protocol uses MAODV (Multicast Ad Hoc On Demand Protocol) as its routing protocol. The protocol is a reactive protocol, that is, it forms the multicast trees only when necessary. The protocol distributes the control of the multicast tree among its members so

that there is no single point of failure. And it provides a loop free mechanism for repairing the broken links. The reactive property of the protocol reduces the number of packets that are transmitted and hence reduces the use of bandwidth. Tree based structure provided by MAODV provides a similar advantage over a mesh architecture. But it has a drawback which has been discussed in section 4.3.1 and section 4.3.2 provides the solution to this drawback.

4.3.1 Drawback of MAODV

A major drawback of the protocol is that it has large control overhead because it broadcasts the control packets throughout the network [20]. Due to this nature of the protocol, unnecessary flooding of control packets is done which reduces the efficiency of protocol.

4.3.2 Proposed Solution

Paper [20] itself describes in its future work that the protocol might benefit from core based topology to reduce the control overhead. Thus the solution first forms the clusters throughout the network which divides the network in two tier hierarchy. At lower level of hierarchy are the nodes in the clusters and the upper level is the communication between the cluster heads of different clusters. Hence, by forming the clusters the control packets can be restricted to the cluster itself if the node required is in the cluster hence, reducing considerable number of unnecessary forwarding of the control packet in the network. Next, the nodes are not allowed to broadcast the packets, instead the nodes send the request packet with time to live as only one hop as the clusters formed are one hop clusters. If the required node is not available in the cluster, then the permission to forward the control packet is given only to the cluster heads and the gateway nodes. So, when a node is not found in the cluster, the cluster head will forward the packet with one hop and the only the gateway will forward it again with one hop which will reach the cluster head of next cluster and same steps are repeated till the requested node is not reached.

Thus forming the clusters, reducing the number of hops of a control packet to one and limiting the nodes who can forward the packets should reduce the traffic of control packets

in the network.

4.3.3 Tables Maintained

Each node in the network maintains four tables:- Neighbor Table, Unicast Route Table, Multicast Route Table, Group Leader Table and Node Status Table.

- **Neighbor Table :-** Neighbor table maintains the list of all the address of all one hop neighbors of the node and the information whether they are cluster head or cluster member.

- **Unicast Route Table :-** This table records the next hope for routes to other destinations for unicast traffic. Usually, the destination is one of the other nodes in the network. A special case is when the destination is a multicast address, which happens when the node is not a multicast tree member but has multicast data packets to send to that multicast group.

- **Multicast Route Table :-** It lists the next hops for the tree structure of each multicast group. Each entry represents one group tree structure. Every node that belongs to that group tree should maintain such entries, with its own identity as group leader, group member, or router (non-multicast member that is in the tree to provide connectivity). Every next hop is associated with direction either downstream or upstream. If the next hop is one-hop nearer to the group leader, the direction is upstream; otherwise, the direction is downstream. The group leader has no upstream, while other nodes in the tree should have one and only one upstream.

- **Group Leader Table :-** It records the currently-known multicast group address with its group leader address and the next hop towards that group leader when a node receives a periodic GRPH message.

- **Node Status Table :-** This is the table where, each node store their own status as either cluster head or cluster member or undecided. It also maintains the number of cluster heads it is member of. If the node is a cluster head than this value should be zero. The node also maintains the status of whether it is a gateway node or not based on the number of cluster head it is member of. If the node is a member of more

than one cluster head only then its status is set to gateway node, else it is set to not a gateway node.

4.3.4 Route Discovery and Route Maintenance

For discovering the route, the nodes two control packets, RREQ for the request and RREP for reply. Following are the steps for discover of route:

- The data source node initiates a RREQ to ask for a route to that multicast address. This packet was initially broadcasted with ttl for whole network diameter. But in the proposed scheme it has been limited to only one. Also once the source node sends the request, only cluster head and gateways can forward the packet. Hence, if the requested node is in another cluster, the path will be formed only through cluster head and cluster gateway.

- If the node has the group leader information in its group leader table and if it is the first time it is sending RREQ, then it unicastly sends it towards the group leader.

- Any node with the information of the multicast address or any tree member with known group leader can respond through RREP.

- While the RREP is sent back to the source node along the reverse route, every intermediate node and the source node updates the route to that tree member with the destination address set to the multicast group address, thus the forwarding route is established in their Unicast Route Tables.

Once the route is formed, RREP packets are used for the maintenance of the route. Following are the steps for the maintenance of the route:-

- During multicast data packet forwarding, each node first checks if itself is in the multicast tree. If the node itself is a tree member, it will follow its Multicast Route Table to forward the packets.

- If it is not a tree member, it will check its Unicast Route Table to find the next hop for the multicast address. If it has the information, the data packets are forwarded

towards the next hop; otherwise, it will send an unsolicited RREP back to the source node, in order to let the source node initiate a new route discovery.

- MAC layer detection is used to detect the link breakage on the route and uses RREP to inform the data source node to discover another route if necessary.

4.3.5 Multicast Tree Creation

For the multicast tree construction, same RREQ and RREP packtes are used with different flags. MACT control packet is also used the packet to initialize the multicast tree. For the formation of the multicast tree, following are the steps followed.

- When a non tree member wants to join a tree first it creates an entry in a multicast route table with itself as a group member , group leader address as unknown and any upstream and downstream next hop. Then it sends a RREQ packet with join flag.
- In normal protocol, the RREQ-J packet is broadcasted in entire network, but in this protocol it is just broadcasted with one hop ttl. If the node has the information about the group leader, then it unicastly sends the RREQ-J packet.
- The nodes with same or larger multicast sequence number reply to the RREQ-J with RREP-J.
- When the RREP-J travels back along the reverse route, the nodes on the path insert the information about the upstream next hop towards the tree in multicast route table. If in future any RREP-J is received with better route, that is, with larger sequence number or smaller number of hop count with same sequence number, then the multicast entry is updated in the table.
- Once the source node receives the RREP-J and waits for specific time, it sends MACT with a join flag MACT-J towards the upstream node and adds next hop in multicast route table.
- All the nodes which receive the MACT-J updates the multicast table with next hop and indicates it as downstream. If the node is a tree member then a branch is formed in a tree.

- If a node is in a tree but not a group member wants to become a group member, it simply changes its identity from router to group member.

4.3.6 Multicast Tree Maintenance

The maintenance of multicast tree includes tasks such as group leader selection, periodic group hello propagation, neighbor connectivity maintenance, tree merge and membership revocation.

4.3.6.1 Group Leader Selection

When there is a partition in the tree or when a group leader revokes its group membership, a new leader must be selected. The duty of the group leader is to periodically send the group hello packets GRPH and maintain the multicast tree. Following are the steps for selection of new group leader:

- When there is a partition in a tree and the current node is a group member, it will become new group leader, else it will force one of the tree neighbors to be the leader.

- If the current node only has one downstream node, it cancels its entry in multicast route table, indicating that it no longer belongs to the tree and sends a MACT with a prune flag MACT-P to a downstream node indicating it will leave the tree and tree will need new group leader.

- If the current node has more than one downstream node, it selects one downstream, change its direction from downstream to upstream, and sends a MACT with a group-leader flag (MACT-GL) towards that node, indicating that it has other branch(es) in the tree and the tree needs a leader. If the node is a group member it becomes the group leader else it sends the packet ahead till the group member is not reached.

- Once the group leader is selected, it will send GRPH-U to every node downstream unicastly so that the nodes can update the group information about new group leader.

4.3.6.2 Periodic Group Hello Propagation

Because the nodes in MANET are mobile, to maintain the multicast tree and to keep the status of the tree updated, group leader of the tree periodically sends the group hello message GRPH throughout the network. In this protocol, this broadcast is a controlled broadcast, as the GRPH is forwarded only by the cluster heads and gateway nodes. Thus stopping the flooding if GRPH messages in the network and yet all the nodes in the network receive the information. Following are the steps that are taken when a GRPH message is received.

- The node updates its group leader table with information about group leader and the route towards it.Then the node if it is a cluster head or gateway retransmits the first-time received GRPH.
- A node that is a tree member and receives GRPH from its own upstream uses it to update its current group sequence number, current group leader and the current distance from the group leader.
- If the tree member receives GRPH from other upstream and the group leader is same, then it discards the packet and waits for GRPH from its own upstream.
- If the GRPH indicates there is another group leader with same multicast group address, then this two trees can be connected and tree merge is initialized by the tree member with a group leader with smaller leader address. If its leader address is larger than in the GRPH then it discards it.

4.3.6.3 Neighbor Connectivity Maintenance

For the maintenance of the neighbor connectivity, periodic one-hop Neighbor-Hello messages are used. To reduce this one-hop Neighbor-Hello overhead, if a node already sent a broadcast message including data packets, it can delay the transmission of the Neighbor-Hello. The node realizes that the link is broken by not receiving any broadcast messages from that neighbor in a specific time which is usually three times of Neighbor-Hello interval. Following are the steps which takes place when the node detects that there is breakage in the link.

- Once the breakage is detected by the downstream node, it deletes the information about the upstream node from the multiast route table and the sends a RREQ-J packet with an extension to find new branch. The RREQ-J packet includes the information about the hop count to the group leader.

- Only those nodes can reply to RREQ-J with RREP-J which has a larger sequence number and equal or smaller hop count to the group leader.

- If in the process the requesting node changes the information such as group leader or group sequence or hop count, it sends GRPH-U to its downstream nodes so that they can update the information.

- If the upstream node that realizes link breakage, the upstream node deletes that next hop in its multicast route table. If this upstream node is not a group member and a leaf node without any downstream, it stays for a while in the tree and after that if it still is a leaf node, it begins self-prune.

4.3.6.4 Member Revocation

A group member including the group leader can revoke its group membership at any time. Following are the steps when a node revokes its membership.

- If a group leader revokes its membership, it changes its identity to router and a new group leader is selected.

- If the node is not a group leader, then it first changes its identity to router and then if it has downstream nodes, it stays in the tree to connect a group member. The node can also completely remove itself from the tree through pruning.

- When a node prunes itself, it removes the entry of that multicast address from its multicast route table. Then it sends MACT-P to its upstream node.

- If receiving a MACT-P makes the upstream node a leaf and it is also not a group member, it can similarly prune itself from the tree with the same action. The procedure terminates when a group member or non-leaf tree member is met.

4.3.6.5 Tree Merge

Tree merge is detected when a tree member with a smaller group leader address receives a GRPH generated by another group leader with a larger address for the same group. Once the tree merge is detected following steps take place to merge two trees.

- The node which has detected the tree merge sends RREQ with repair flag (RREQ-R) upstream to the group leader, for the permission to rebuild the tree.
- If the group leader has not given permission to any other node, it gives the permission to the node by sending RREP-R through the same route.
- The request node sends the RREQ join and repair (RREQ-JR) to the group leader with larger address. The group leader sends the RREP-JR to the request node. During the travel of RREP-JR, the group information, such as group leader address, group sequence number, and hop count to group leader, is updated.
- When the tree member with smaller group leader address is reached, the node adds a next hop from which it received the RREP-JR as upstream. This node then sends the RREP-JR upstream towards the group leader with smaller address.
- At each node, the nodes change the status from where they received the RREP-JR as upstream and the nodes which send the RREP-JR upstream as the status of those nodes as downstream.
- When the group leader is reached, it changes the downstream nodes to upstream nodes and changes its identity from group leader to group member. And then it sends GRPH-U to its downstream to indicate change of information.
- If the group leader itself detects the tree merge, then the RREQ-R and RREP-R steps are omitted and it itself starts building the new tree.

Chapter 5

Simulation

5.1 Introduction

In the network research area, it is very costly to deploy a complete test bed containing multiple networked computers, routers and data links to validate and verify a network protocol or a specific network algorithm. The network simulators in these circumstances save a lot of money and time in accomplishing this task. Network simulators are also particularly useful in allowing the network designers to test new networking protocols or to change the existing protocols in a controlled and reproducible manner. Thus we use a network simulator to test the proposed protocol.[38]

5.2 Study of various simulators

There are many network simulators available. Various protocols provide various facilities. Thus, it is very important to choose the network simulator which is suitable for the implementation of our protocol.
Simulators can be classified on the basis of various factors. One of them is commercial and open source. OPNET and QUALNET are commercial network simulators, while NS2, NS3 and OMNeT++ are open source simulators.

- **OPNET**:- OPNET is based on a mechanism called discrete event system which means that the system behavior can simulate by modeling the events in the system in the

order of the scenarios the user has set up. Hierarchical structure is used to organize the networks. As other network simulators, OPNET also provides programming tools for users to define the packet format of the protocol. The programming tools are also required to accomplish the tasks of defining the state transition machine, defining network model and the process module.[38]

- **QualNet**:- QualNet is the first commercial simulator in this comparison. It is based on GloMoSim developed at the University of California, Los Angeles (UCLA). GloMoSim uses the Parallel Simulation Enviroment for Complex Systems (PARSEC) for basic operations.QualNet also has a graphical user interface for creating the model and its specification. So it is by far easier to specify small to medium networks by using the GUI compared to specifying all connections in a special model file manually.QualNet is a network simulator targeting at wireless solutions, however it also has support for wired networks. Its environment and library is very sophisticated, which makes it very easy to simulate a real network with QualNet. This however makes the simulation of logical networks a little bit more difficulty, but it is possible as well.Since it uses primarily Java for the GUI it is available for Linux as well as for Windows. The simulator itself is a for the specified target system optimized C program.[39]

- **NS-2**:- The network simulator 2 (ns-2) is a frequently used discrete event simulator mainly for networking research. For implementing the model, the object orientated extension of TCL, OTcl is used. The functionality of modules can also be coded in OTcl or for speed optimizations in C++. ns-2 includes the most common network technologies and applications, for very easy and fast network specification and simulations.ns-2 is also free available for research and education issues. It is designed for Unix-Systems but runs under Windows CygWin as well.[39]

- **NS-3**:- Similar to NS2, NS3 is also an open sourced discrete-event network simulator which targets primarily for research and educational use. NS3 is licensed under the GNU GPLv2 license , and is available for research and development. NS3 is designed to replace the current popular NS2 . However, NS3 is not an updated version of NS2 since that NS3 is a new simulator and it is not backward-compatible with NS2.[38]

- **OMNET++**:- OMNet++ is a public-source simulation environment, which main goal is the simulation of communication networks. However its design is quite open, which enables also other target applications. It has a sophisticated GUI support and common used models like IPv4, IPv6, Ethernet, MPLS etc. are available. OMNet++ is free for academic and non-profit use, however for commercial use a license must be obtained.OMNet++ modules are structured by an own network definition language NED, while the functionality is coded by using C++ classes. OMNet++ is available for Unix-Systems (like Linux) as well as for Windows supporting Visual C++ 6.0, Visual Studio .NET 2003 and CygWin.[39]

We have chosen NS-2 as the simulator to be used for the simulation of the proposed protocol. Following are the reason for choosing NS-2 as the simulator:-

- It is open source, hence it is free.
- Complex scenarios can be easily tested.
- Many modules and protocols are already implemented in NS-2.
- The protocols are implemented according to RFC. If theory and implementation does not match, it is not included.

5.3 Features of NS-2 [38]

First and foremost, NS2 is an object-oriented, discrete event driven network simulator which was originally developed at University of California-Berkely. The programming it uses is C++ and OTcl (Tcl script language with Object-oriented extensions developed at MIT). The usage of these two programming language has its reason. The biggest reason is due to the internal characteristics of these two languages. C++ is efficient to implement a design but it is not very easy to be visual and graphically shown. It's not easy to modify and assembly different components and to change different parameters without a very visual and easy-to-use descriptive language. Moreover, for efficiency reason, NS2 separates control path implementations from the data path implementation. The event scheduler and the basic network component objects in the data path are written and compiled using C++

to reduce packet and event processing time. OTcl happens to have the feature that C++ lacks. So the combination of these two languages proves to be very effective. C++ is used to implement the detailed protocol and OTcl is used for users to control the simulation scenario and schedule the events. A simplified user's view of NS2 is shown in figure. The OTcl script is used to initiate the event scheduler, set up the network topology, and tell traffic source when to start and stop sending packets through event scheduler. The scenes can be changed easily by programming in the OTcl script. When a user wants to make a new network object, he can either write the new object or assemble a compound object from the existing object library, and plumb the data path through the object. This plumbing makes NS2 very powerful. Another feature of NS2 is the event scheduler. In NS2, the event scheduler keeps track of simulation time and release all the events in the event queue by invoking appropriate network components. All the network components use the event scheduler by issuing an event for the packet and waiting for the event to be released before doing further action on the packet.

5.3.1 Features to be used

For the development of the protocol following are the steps which should be carried out:-

- Develop protocol in C++ language.
- Design various packets that are to be used by the protocol.
- Integrate the protocol with NS-2.

Once the protocol is designed and integrated, it has to be checked for various scenarios. Thus following are the steps that have to be followed to check the protocol for a particular scenario:-

- Design the scenario using the TCL script.
- As it is a MANET, we need to consider the movements of the nodes as well.

- Various parameters of the nodes should be defined for their configuration:-
  - Ad Hoc routing protocol

- Link Layer Type
- MAC Layer
- Type of Queue
- Length of Queue
- Antenna Type
- Wireless Propogation Model
- Type of physical interface
- Type of channel
- Define the transmission and receiving range of antenna.
- Type of traffic
- Bandwidth of channel
- Rate at which traffic is sent
- Speed of node
- Type of motion

5.4 Comparison of unicast and multicast routing protocols

Table 5.1: Parameters for nodes

Parameter	**Value**
Link Layer Type	LL
MAC Layer	MAC/802.11
Type of Queue	DropTail/PriQueue
Antenna Type	OmniAntenna
Wireless Propogation Model	TwoRayground
Type of physical interface	Wireless
Type of channel	Wireless channel
Transmission and Receiving range of antenna	250 m

Table 5.2: Scenario

Parameter	**Value**
Area	1000x1000
Wireless Propogation Model	TwoRayGround
Type of channel	Wireless channel
Speed of node	20 m/s
Type of motion	RandomWayPoint
Pause Time	1.0 s
Number of nodes	50
Number of receivers	15,20,30,40
Type of traffic	CBR
Packet size	512 Kb
Rate	20 packets/s
Maximum Packets	10000

Automation of Traffic creation:-

Table 5.3: Difference between new and old version

Older version	**New Version**
No way of controlling number of sources	Number of sources can be specified
No way to specify exact number of connections	Exact number of connections can be specified
Maximum only two receivers connected to a source	Random number of receivers can be connected to a source
Traffic file only for unicast routing protocols	Traffic file support both unicast and multicast routing protocols

Example of traffic file generated:-

Here, Figure 5.1 shows a list of random sources that are generated and figure 5.2 shows an example of code that is generated for making connection between two nodes, in this case node 0 and node 33 by running the tcl file that is being made to generate the traffic connections.

```
# nodes: 50, max conn: 26, send rate: 0.05000000000000003, seed: 31
#
#Source /0: 0
#
#Source /1: 11
#
#Source /2: 22
#
#Source /3: 36
#
#Source /4: 33
#
#Source /5: 38
#
#Source /6: 35
#
#Source /7: 15
#
#Source /8: 13
#
#Source /9: 40
#
#Source /10: 29
#
#Source /11: 46
#
#Source /12: 4
#
#Source /13: 16
#
#Source /14: 42
```

Figure 5.1: Traffic file

```
# 0 connecting to 33 at time 52.791335691135068
#
set udp_(0) [new Agent/UDP]
$udp_(0) set dst_addr_ 0xE000000
$udp_(0) set dst_port_ 100
$ns_ attach-agent $node_(0) $udp_(0)
set null_(0) [new Agent/Null]
$node_(0) attach $null_(0) 100
set cbr_(0) [new Application/Traffic/CBR]
$cbr_(0) set packetSize_ 512
$cbr_(0) set interval_ 0.050000000000000003
$cbr_(0) set random_ 1
$cbr_(0) set maxpkts_ 10000
$cbr_(0) set dst_ 0xE000000
$cbr_(0) attach-agent $udp_(0)
$ns_ at 52.791335691135068 "$cbr_(0) start"
set null_(1) [new Agent/Null]
$node_(33) attach $null_(1) 100
$ns_ at 1.20000000000000 "$node_(0) join 0xE000000"
$ns_ at 52.791335691135068 "$node_(33) join 0xE000000"
$ns_ at 100.000000000000000 "$node_(0) leave 0xE000000"
$ns_ at 100.000000000000000 "$node_(33) leave 0xE000000"
```

Figure 5.2: Traffic file

Results:-

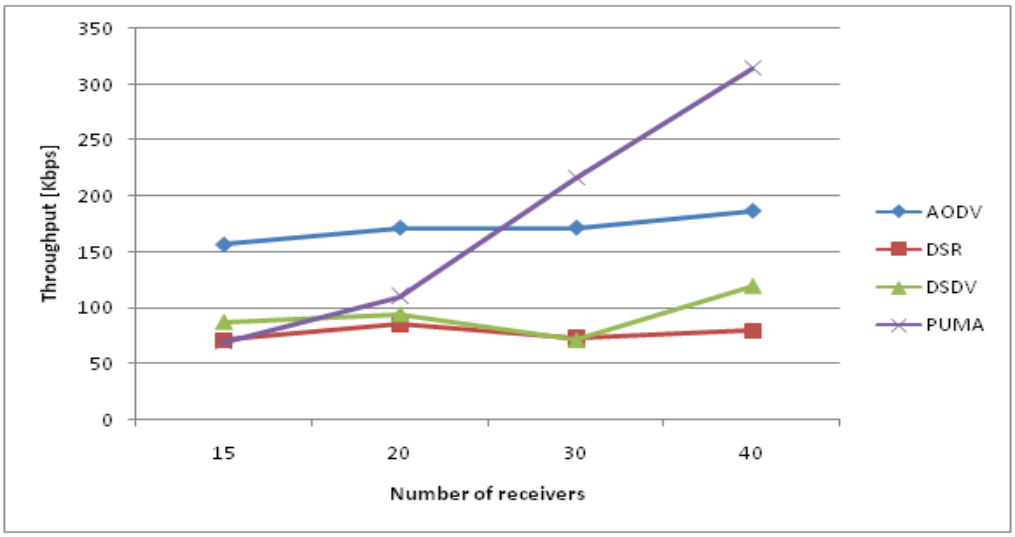

Figure 5.3: Result

Conclusion from the result:-

- As the number of receiving nodes increase, multicast routing protocol performs better than the unicast routing protocol.
- AODV has the best result among all the three unicast routing protocols taken into consideration.

5.5 Implementation of protocol

5.5.1 Implementation of cluster formation

For the formation of clusters and cluster heads, CBRP protocol is used. The code for this protocol was for older version of NS-2 and hence the integration of the code to NS-2.35 was difficult. There were differences in the syntax and the functions used. Also the code contained various logical errors which had to be solved. **GDB** debugging tool was used to overcome the errors.

```
[abhishek@Abhishek version-1-cbrp]$ gdb ns
GNU gdb (GDB) Fedora (7.2.90.20110429-36.fc15)
Copyright (C) 2011 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.  Type "show copying"
and "show warranty" for details.
This GDB was configured as "i686-redhat-linux-gnu".
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>...
Reading symbols from /usr/local/bin/ns...done.
(gdb) r cmmrp1.tcl
`/usr/local/bin/ns' has changed; re-reading symbols.
Starting program: /usr/local/bin/ns cmmrp1.tcl
num_nodes is set 5
warning: Please use -channel as shown in tcl/ex/wireless-mitf.tcl
Creating node 0
INITIALIZE THE LIST xListHead
Creating node 1
Creating node 2
Creating node 3
Creating node 4
Starting Simulation...
channel.cc:sendUp - Calc highestAntennaZ_ and distCST_
highestAntennaZ_ = 1.5,  distCST_ = 550.0
SORTING LISTS ...DONE!
mobile/tworayground.cc: TwoRayGround propagation model assume flat ground
mobile/tworayground.cc: TwoRayGround propagation model assume flat ground
mobile/tworayground.cc: TwoRayGround propagation model assume flat ground
mobile/tworayground.cc: TwoRayGround propagation model assume flat ground
mobile/tworayground.cc: TwoRayGround propagation model assume flat ground
mobile/tworayground.cc: TwoRayGround propagation model assume flat ground
[Inferior 1 (process 2423) exited with code 01]
(gdb)
```

Figure 5.4: GDB

Various breakpoints were put in the code using gdb for debugging the code. The command **rbreak** was used to put breakpoints which is shown in figure 5.5. This command enables to put breakpoints for all the functions having a particular pattern which is given as an argument.Then the tcl file was run using command **run** with filename as an argument. The debugger stops the program at the breakpoint and the command **s** runs the program step by step which is shown in figure 5.6, while the command **c** stops the program when next breakpoint is encountered.The debugger also enables us to look at the memory stack. The memory stack can be accessed using the command **bt** as shown in figure 5.7.

```
<function, no debug info> NeighborTable::printHELLOpkt(Packet*);
Breakpoint 20 at 0x8309090
<function, no debug info> NeighborTable::PrintTwoHopNeighbors();
Breakpoint 21 at 0x8309182
<function, no debug info> NeighborTable::DeleteAdjEntry(int, int);
Breakpoint 22 at 0x8315f5e
<function, no debug info> NeighborTable::insertionPoint() const;
Breakpoint 23 at 0x8315f9a
<function, no debug info> NeighborTable::networkId(unsigned int) const;
Breakpoint 24 at 0x8315fd7
<function, no debug info> NeighborTable::entryPresent(unsigned int) const;
Breakpoint 25 at 0x83175e0
<function, no debug info> NeighborTable::update(std::basic_string<char, std::char_traits<char>, std::allocator<char> >);
Breakpoint 26 at 0x831796e
<function, no debug info> NeighborTable::print(int) const;
Breakpoint 27 at 0x8317d65
<function, no debug info> NeighborTable::NeighborTable();
Note: breakpoint 27 also set at pc 0x8317d65.
Breakpoint 28 at 0x8317d65
<function, no debug info> NeighborTable::NeighborTable();
Breakpoint 29 at 0x831b1de
<function, no debug info> NeighborTable::levelId(unsigned int, int) const;
Breakpoint 30 at 0x831b22e
<function, no debug info> NeighborTable::hopNumber(unsigned int) const;
Breakpoint 31 at 0x831b260
<function, no debug info> NeighborTable::etxMetric(unsigned int) const;
Breakpoint 32 at 0x831b29e
<function, no debug info> NeighborTable::routeLog(unsigned int) const;
Breakpoint 33 at 0x831b2e6
<function, no debug info> NeighborTable::routeLog(unsigned int, unsigned int) const;
(gdb)
```

Figure 5.5: Breakpoints in GDB

```
Breakpoint 12, 0x0830841e in NeighborTable::AddUpdateEntry(ntable_ent*) ()
(gdb)
Continuing.

Breakpoint 18, 0x08308e52 in NeighborTable::printNeighbors() ()
(gdb)
Continuing.

Breakpoint 13, 0x083087ae in NeighborTable::processUpdate(Packet*) ()
(gdb)
Continuing.

Breakpoint 12, 0x0830841e in NeighborTable::AddUpdateEntry(ntable_ent*) ()
(gdb)
Continuing.

Breakpoint 1, 0x083073e2 in NeighborTable::getBroadcastPacket() ()
(gdb)
Continuing.

Breakpoint 5, 0x08307f75 in NeighborTable::InitLoop() ()
(gdb)
Continuing.

Breakpoint 6, 0x08307f8f in NeighborTable::NextLoop() ()
(gdb)
Continuing.

Breakpoint 6, 0x08307f8f in NeighborTable::NextLoop() ()
(gdb)
Continuing.

Breakpoint 6, 0x08307f8f in NeighborTable::NextLoop() ()
(gdb)
Continuing.
[Inferior 1 (process 2455) exited with code 01]
(gdb)
The program is not being run.
(gdb) bt
No stack.
(gdb)
```

Figure 5.6: Step by step execution

```
(gdb) bt
#0  0x08307f8f in NeighborTable::NextLoop() ()
#1  0x08307591 in NeighborTable::getBroadcastPacket() ()
#2  0x08307652 in NeighborTablePeriodicHandler::handle(Event*) ()
#3  0x081e24d4 in Scheduler::dispatch(Event*, double) ()
#4  0x081e2408 in Scheduler::run() ()
#5  0x081e25b5 in Scheduler::command(int, char const* const*) ()
#6  0x0840df52 in TclClass::dispatch_cmd(void*, Tcl_Interp*, int, char const**) ()
#7  0x08411a4a in OTclDispatch (cd=0x88e1770, in=0x86f2cf8, argc=3, argv=0x86f38c8) at otcl.c:455
#8  0x084181b7 in TclInvokeStringCommand ()
#9  0x0841ceae in TclEvalObjvInternal ()
#10 0x08465751 in TclExecuteByteCode ()
#11 0x0846d043 in TclCompEvalObj ()
#12 0x08465670 in TclExecuteByteCode ()
#13 0x084ab58e in TclObjInterpProcCore ()
#14 0x084abb49 in TclObjInterpProc ()
#15 0x084183d3 in TclInvokeObjectCommand ()
#16 0x08411ba2 in OTclDispatch (cd=0x88e1770, in=0x86f2cf8, argc=2, argv=0x86f37c0) at otcl.c:498
#17 0x084181b7 in TclInvokeStringCommand ()
#18 0x0841ceae in TclEvalObjvInternal ()
#19 0x08465751 in TclExecuteByteCode ()
#20 0x084ab58e in TclObjInterpProcCore ()
#21 0x084abb49 in TclObjInterpProc ()
#22 0x084183d3 in TclInvokeObjectCommand ()
#23 0x08411a4a in OTclDispatch (cd=0x88db568, in=0x86f2cf8, argc=2, argv=0x86f36b0) at otcl.c:455
#24 0x084181b7 in TclInvokeStringCommand ()
#25 0x0841ceae in TclEvalObjvInternal ()
#26 0x0841eb24 in TclEvalEx ()
#27 0x0841f1cb in Tcl_EvalEx ()
#28 0x0848b13c in Tcl_FSEvalFileEx ()
#29 0x084909a4 in Tcl_Main ()
#30 0x0840d234 in nslibmain ()
#31 0x0840d38f in main ()
(gdb)
```

Figure 5.7: Stack

Following files were added to integrate CBRP:

- cmmrp.tcl
- cmmrpagent.h
- cmmrpagent.cc
- cmmrp_packet.h
- hdr_cmmrp.h
- ntable.h
- ntable.cc

Following files were modified to integrate CBRP:-

- ns-2.35/tcl/lib/ns-packet.tcl
- ../common/packet.h
- ../trace/cmu-trace.h
- ../trace/cmu-trace.cc
- ../tcl/lib/ns-mobilenode.tcl

- ../tcl/lib/ns-default.tcl
- ../ns-lib.tcl

Cluster formation:-

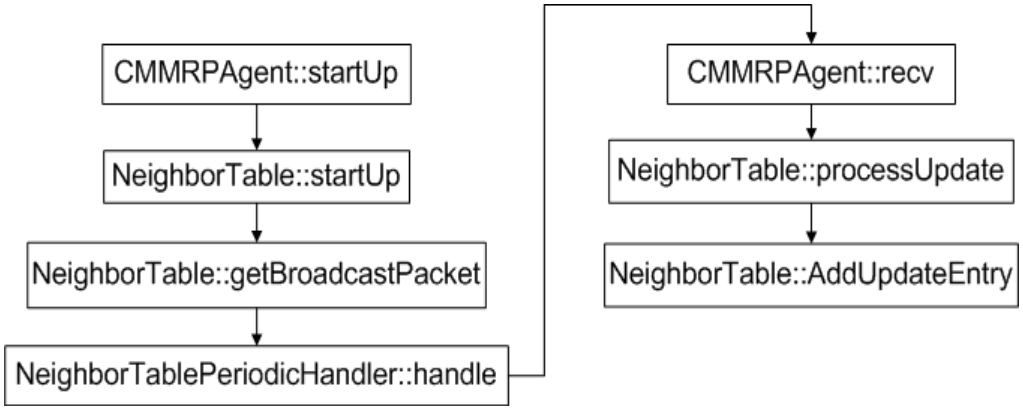

Figure 5.8: Flow through the functions

```
SUM CLUSTERS 4.00000 1
S Cluster head 4.00000 _0_
SUM CLUSTERS 4.00000 2
S Cluster head 4.00000 _1_
SUM CLUSTERS 4.00000 3
S Cluster head 4.00000 _2_
SUM CLUSTERS 4.00000 4
S Cluster head 4.00000 _4_
SUM CLUSTERS 4.00000 5
S Cluster head 4.00000 _5_
SUM CLUSTERS 4.00000 6
S Cluster head 4.00000 _7_
SUM CLUSTERS 4.00000 7
S Cluster head 4.00000 _15_
SUM CLUSTERS 4.00000 8
S Cluster head 4.00000 _18_
SUM CLUSTERS 4.00000 9
S Cluster head 4.00000 _50_
```

Figure 5.9: List of cluster heads

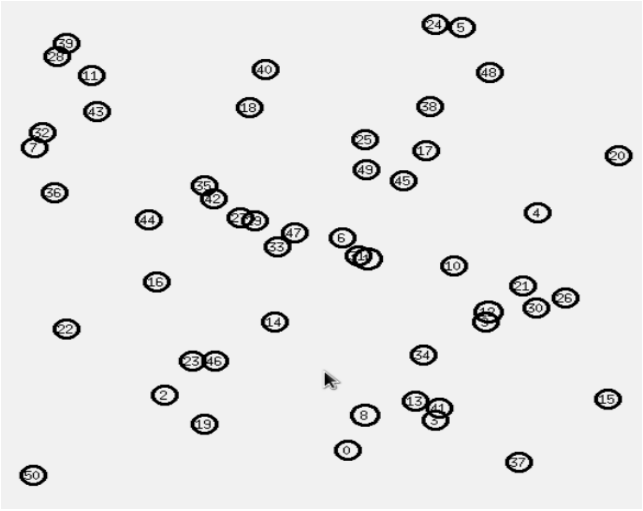

Figure 5.10: Topology

Problem in routing:- The entries in the routing tables are not proper. There are two tables, one for one hop neighbors and other for two hop neighbors, but even the two hop entries are made in one hop neighbor and when an entry is made in two hop neighbor it is incorrect.

```
s 5.313365876 _1_ RTR  --- 0 message 13 [0 0 0 0] ------- [1:0 -1:255 0 0]
r 5.314234543 _0_ RTR  --- 0 message 13 [0 ffffffff 1 800] ------- [1:0 -1:255 0 0]
[]
[2 3 1 ]
r 5.314234546 _2_ RTR  --- 0 message 13 [0 ffffffff 1 800] ------- [1:0 -1:255 0 0]
[ (3 1 4.00145) ]
[1 0 ]
r 5.314235211 _3_ RTR  --- 0 message 13 [0 ffffffff 1 800] ------- [1:0 -1:255 0 0]
[ (2 1 4.00145) ]
[1 0 ]
s 5.485138145 _0_ RTR  --- 0 message 17 [0 0 0 0] ------- [0:0 -1:255 0 0]
r 5.485958812 _1_ RTR  --- 0 message 17 [0 ffffffff 0 800] ------- [0:0 -1:255 0 0]
[]
[2 3 0 ]
```

Figure 5.11: Error in table entry

5.5.2 Implementation of clustering in multicast routing

For the implementation of clustering in multicast routing, first a multicast routing was integrated over AODV and then clustering was integrated over the multicast routing protocol and changes were made in the functions of the protocol, new contents were added in the header of the packet and new table was added. Following are the files that were changed for adding clustering mechanism:-
Files added:

- aodv_mcast.cc
- aodv_mtable_aux.cc
- aodv_mtable_aux.h
- aodv_mtable.cc
- aodv_mtable.h

Files modified:

- aodv.h
- aodv.cc
- aodv_packet.h
- aodv_rqueue.cc
- aodv_rqueue.h
- aodv_rtable.cc
- aodv_rtable.h
- cmu-trace.cc

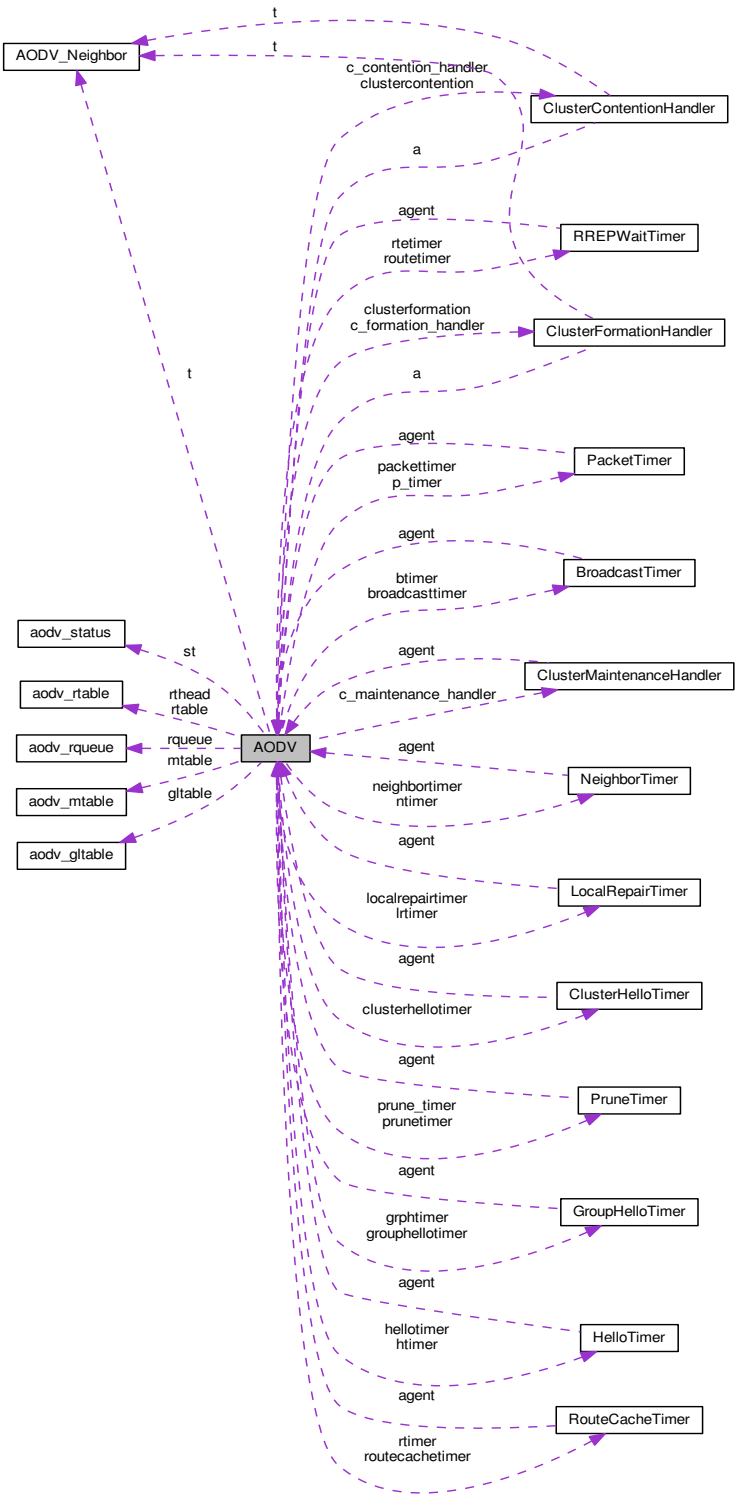

Figure 5.12: Collaboration Graph

5.5.3 Simulation Parameters

Table 5.4: Parameters for nodes

Parameter	**Value**
Link Layer Type	LL
MAC Layer	MAC/802.11
Type of Queue	DropTail/PriQueue
Antenna Type	OmniAntenna
Wireless Propogation Model	TwoRayground
Type of physical interface	Wireless
Type of channel	Wireless channel
Transmission and Receiving range of antenna	250 m

Table 5.5: Scenario

Parameter	**Value**
Area	1000m x 1000m
Wireless Propogation Model	TwoRayGround
Type of channel	Wireless channel
Speed of node	5 m/s, 10m/s, 15 m/s and 20 m/s
Type of motion	RandomWayPoint
Pause Time	0 s, 30 s, 60 s, 120 s, 300 s and 600 s
Number of nodes	25, 50, 75 and 100
Number of sources	5, 10, 15 and 20
Number of receivers	15, 30, 45 and 60
Type of traffic	CBR
Packet size	512B
Rate	512 Kbps
Maximum Packets	10000

5.5.4 Simulation Results

Table 5.6: Throughput(Kbps) for AODV

Speed - 5 m/s or 18 km/hr						
Nodes	**Pause Time (s)**					
	0	**30**	**60**	**120**	**300**	**600**
25	332	339	230	350	556	304
50	509	413	215	318	221	921
75	198	298	177	290	265	165
100	150	134	298	248	304	353
Speed - 10 m/s or 36 km/hr						
Nodes	**Pause Time (s)**					
	0	**30**	**60**	**120**	**300**	**600**
25	342	188	281	255	246	466
50	267	232	171	362	392	161
75	235	200	224	317	280	272
100	176	232	283	156	235	159
Speed - 15 m/s or 54 km/hr						
Nodes	**Pause Time (s)**					
	0	**30**	**60**	**120**	**300**	**600**
25	272	347	260	610	435	384
50	305	344	314	405	200	405
75	327	212	276	152	346	211
100	165	143	235	372	310	309
Speed - 20 m/s or 72 km/hr						
Nodes	**Pause Time (s)**					
	0	**30**	**60**	**120**	**300**	**600**
25	395	257	240	312	607	164
50	234	353	332	305	322	389
75	228	217	287	217	186	479
100	197	211	146	187	278	476

Table 5.7: Throughput(Kbps) for MAODV

Speed - 5 m/s or 18 km/hr						
Nodes	**Pause Time (s)**					
	0	**30**	**60**	**120**	**300**	**600**
25	289	285	232	233	642	32
50	364	482	183	486	338	1124
75	243	299	88	351	388	64
100	149	165	351	309	421	285
Speed - 10 m/s or 36 km/hr						
Nodes	**Pause Time (s)**					
	0	**30**	**60**	**120**	**300**	**600**
25	443	77	287	175	252	377
50	282	187	154	491	300	432
75	232	188	167	281	235	241
100	140	195	272	117	231	243
Speed - 15 m/s or 54 km/hr						
Nodes	**Pause Time (s)**					
	0	**30**	**60**	**120**	**300**	**600**
25	261	345	309	658	263	333
50	443	220	473	419	56	465
75	383	184	290	126	338	163
100	114	110	190	328	224	531
Speed - 20 m/s or 72 km/hr						
Nodes	**Pause Time (s)**					
	0	**30**	**60**	**120**	**300**	**600**
25	306	131	162	269	650	171
50	232	325	327	216	436	708
75	216	192	350	356	179	481
100	157	130	110	267	413	438

Table 5.8: Throughput(Kbps) for CMAODV

Speed - 5 m/s or 18 km/hr						
Nodes	**Pause Time (s)**					
	0	**30**	**60**	**120**	**300**	**600**
25	247	287	184	157	657	88
50	332	566	133	494	254	961
75	233	313	140	345	410	204
100	109	175	356	352	393	682
Speed - 10 m/s or 36 km/hr						
Nodes	**Pause Time (s)**					
	0	**30**	**60**	**120**	**300**	**600**
25	480	114	226	245	361	365
50	215	188	129	428	248	404
75	329	316	163	143	360	647
100	169	235	368	45	319	287
Speed - 15 m/s or 54 km/hr						
Nodes	**Pause Time (s)**					
	0	**30**	**60**	**120**	**300**	**600**
25	241	234	345	510	276	203
50	561	274	373	153	60	457
75	421	77	325	151	359	633
100	64	106	109	321	322	667
Speed - 20 m/s or 72 km/hr						
Nodes	**Pause Time (s)**					
	0	**30**	**60**	**120**	**300**	**600**
25	245	131	104	142	613	73
50	226	185	344	210	265	754
75	134	184	370	334	119	421
100	159	87	122	302	473	322

Table 5.9: Delay(ms) for AODV

Speed - 5 m/s or 18 km/hr						
Nodes	**Pause Time (s)**					
	0	**30**	**60**	**120**	**300**	**600**
25	896.925	1010.77	1315.64	853.852	572.137	860.273
50	887.553	1500.98	1927.99	1992.94	3202.51	714.902
75	1982.67	1945.19	2432.82	1537.47	2311.9	2134.35
100	2381.75	3230.26	1528.82	2061.09	1841.05	1557.55
Speed - 10 m/s or 36 km/hr						
Nodes	**Pause Time (s)**					
	0	**30**	**60**	**120**	**300**	**600**
25	1318.37	1485.52	1467.63	1485.96	1715.19	384.165
50	1936.21	1984.5	1989.49	1359.11	1442.58	2301.4
75	2016.49	1902.73	2114.89	1718.37	2171.54	2508.13
100	2439.06	2212.9	1879.885	3081.78	1969.38	1454.03
Speed - 15 m/s or 54 km/hr						
Nodes	**Pause Time (s)**					
	0	**30**	**60**	**120**	**300**	**600**
25	1384.89	1042.93	1115.74	383.159	503.843	1078.71
50	1934.2	1319.39	1332.19	1191.62	2704.56	1649
75	1539.71	1631.82	1605.78	2531	1639.08	2103.6
100	1826.78	2272.12	1952.83	1391.71	1981.46	2551.59
Speed - 20 m/s or 72 km/hr						
Nodes	**Pause Time (s)**					
	0	**30**	**60**	**120**	**300**	**600**
25	1089.2	1144.36	1085.53	800.128	525.065	1686.24
50	1742.88	1085.39	1799.72	1704.08	2293.71	1767.95
75	1989.19	1911.93	1382.52	2339.05	2656.46	1187.5
100	1777.8	1862.83	2421.19	2339.49	2226.67	1415.34

Table 5.10: Delay(ms) for MAODV

Speed - 5 m/s or 18 km/hr						
Nodes	**Pause Time (s)**					
	0	**30**	**60**	**120**	**300**	**600**
25	695	1005	956	670	298	1789
50	887	786	1063	894	1186	420
75	930	738	1262	655	806	1500
100	776	1276	1124	1126	833	1516
Speed - 10 m/s or 36 km/hr						
Nodes	**Pause Time (s)**					
	0	**30**	**60**	**120**	**300**	**600**
25	1171	1230	934	1892	1017	111
50	1256	1347	920	867	1465	730
75	1118	1054	1227	1122	1411	1437
100	1211	793	770	956	1049	1158
Speed - 15 m/s or 54 km/hr						
Nodes	**Pause Time (s)**					
	0	**30**	**60**	**120**	**300**	**600**
25	858	556	742	433	235	1197
50	711	1048	627	906	2352	910
75	880	1003	1122	1719	919	970
100	952	992	737	1007	1324	1094
Speed - 20 m/s or 72 km/hr						
Nodes	**Pause Time (s)**					
	0	**30**	**60**	**120**	**300**	**600**
25	938	1052	814	647	379	1550
50	822	627	1048	1258	1348	736
75	1229	998	629	951	1402	1122
100	970	955	1135	771	1042	959

Table 5.11: Delay(ms) for CMAODV

Speed - 5 m/s or 18 km/hr						
Nodes	**Pause Time (s)**					
	0	**30**	**60**	**120**	**300**	**600**
25	557	700	842	615	250	2116
50	806	544	912	801	1255	369
75	702	636	799	639	1099	2159
100	1112	2036	1067	1143	1177	1447
Speed - 10 m/s or 36 km/hr						
Nodes	**Pause Time (s)**					
	0	**30**	**60**	**120**	**300**	**600**
25	707	979	519	864	913	90
50	1225	1118	1363	564	1062	435
75	744	588	1049	1415	1232	929
100	1688	806	1149	1763	1175	1731
Speed - 15 m/s or 54 km/hr						
Nodes	**Pause Time (s)**					
	0	**30**	**60**	**120**	**300**	**600**
25	571	499	576	348	196	1541
50	779	812	673	1232	2110	1153
75	855	926	1043	1391	1218	689
100	1433	1357	959	999	1252	1146
Speed - 20 m/s or 72 km/hr						
Nodes	**Pause Time (s)**					
	0	**30**	**60**	**120**	**300**	**600**
25	449	563	359	380	581	1880
50	871	1050	904	1019	1634	831
75	1209	862	662	997	1795	1599
100	873	1761	1381	899	897	1425

Table 5.12: Number of control packets generated and forwarded in AODV

Speed - 5 m/s or 18 km/hr						
Nodes	**Pause Time (s)**					
	0	**30**	**60**	**120**	**300**	**600**
25	26258	20181	13323	7720	8162	14263
50	103193	104908	131705	90579	74455	48086
75	267982	229494	277000	283262	253477	258328
100	421424	427345	408346	408425	367656	370163
Speed - 10 m/s or 36 km/hr						
Nodes	**Pause Time (s)**					
	0	**30**	**60**	**120**	**300**	**600**
25	6623	22516	13420	8498	10400	13150
50	97725	112036	129682	118144	119407	117993
75	244732	290696	242402	233319	236460	238915
100	420596	389694	438638	398999	431404	145667
Speed - 15 m/s or 54 km/hr						
Nodes	**Pause Time (s)**					
	0	**30**	**60**	**120**	**300**	**600**
25	18001	9995	13412	9409	8573	3203
50	106219	113007	130295	113200	69149	80659
75	253040	298805	257547	299718	257529	259255
100	430954	435947	415093	432528	392884	355230
Speed - 20 m/s or 72 km/hr						
Nodes	**Pause Time (s)**					
	0	**30**	**60**	**120**	**300**	**600**
25	11802	13828	12632	11242	5592	2221
50	135240	117279	92046	99499	69048	59812
75	285412	256642	272856	263044	234274	216805
100	454079	446018	441176	438247	369161	332822

Table 5.13: Number of control packets generated and forwarded in MAODV

Speed - 5 m/s or 18 km/hr						
Nodes	**Pause Time (s)**					
	0	**30**	**60**	**120**	**300**	**600**
25	100906	78200	59145	81850	184773	23804
50	224749	253084	169319	224130	195852	394284
75	334575	360235	314476	374156	373214	261859
100	555626	541093	568323	540019	488955	345248
Speed - 10 m/s or 36 km/hr						
Nodes	**Pause Time (s)**					
	0	**30**	**60**	**120**	**300**	**600**
25	108261	41863	94855	62712	64796	118676
50	206093	196881	173014	253152	185744	215853
75	376558	379322	333682	368509	313708	287325
100	577496	571032	571887	515451	533031	515148
Speed - 15 m/s or 54 km/hr						
Nodes	**Pause Time (s)**					
	0	**30**	**60**	**120**	**300**	**600**
25	78859	105197	86843	191414	87671	87251
50	272558	209837	264166	225615	117800	199722
75	406876	353070	384844	344108	328386	205401
100	566470	606029	550215	553624	530639	542827
Speed - 20 m/s or 72 km/hr						
Nodes	**Pause Time (s)**					
	0	**30**	**60**	**120**	**300**	**600**
25	100816	58841	54959	73912	183851	61599
50	225446	222023	201433	165991	196235	274727
75	386849	351080	405638	380137	322318	379591
100	596857	580798	573348	583576	541821	335867

Table 5.14: Number of control packets generated and forwarded in CMAODV

Speed - 5 m/s or 18 km/hr						
Nodes	**Pause Time (s)**					
	0	**30**	**60**	**120**	**300**	**600**
25	91498	88571	55054	69116	195868	42949
50	192069	235875	132481	199645	144550	339044
75	238968	264651	184861	247854	240327	178027
100	364549	339628	376045	368467	315287	354839
Speed - 10 m/s or 36 km/hr						
Nodes	**Pause Time (s)**					
	0	**30**	**60**	**120**	**300**	**600**
25	136601	53079	86971	84896	119857	122768
50	143858	138009	113132	206242	162966	179031
75	266524	262179	193912	189303	253636	315663
100	346774	367204	395862	294891	343068	327163
Speed - 15 m/s or 54 km/hr						
Nodes	**Pause Time (s)**					
	0	**30**	**60**	**120**	**300**	**600**
25	87071	88025	106024	160898	84434	55912
50	243664	141461	192420	120507	96371	191986
75	281145	179499	261633	207137	228244	252608
100	333215	384402	345746	389405	353115	335774
Speed - 20 m/s or 72 km/hr						
Nodes	**Pause Time (s)**					
	0	**30**	**60**	**120**	**300**	**600**
25	114704	60721	50613	73632	183495	43574
50	152380	134961	184079	129146	128722	265328
75	206266	211573	273523	247033	175447	230850
100	368778	341662	355952	333992	370071	237681

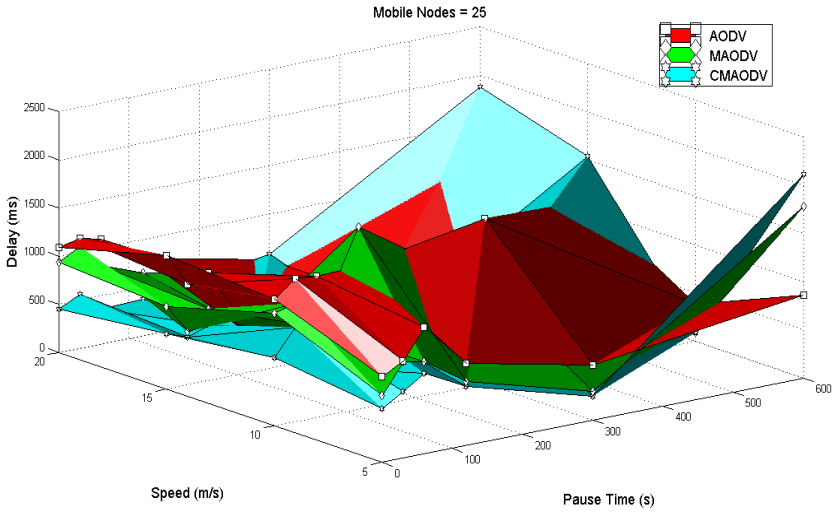

Figure 5.13: Delay for 25 nodes

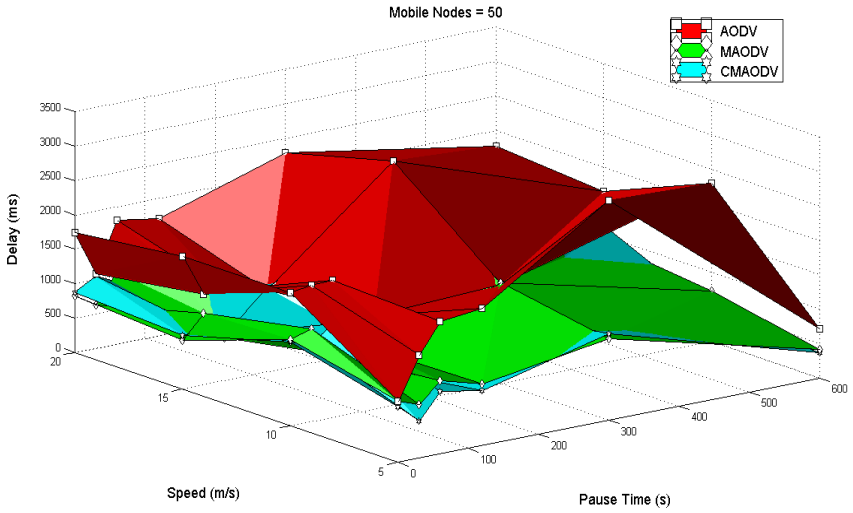

Figure 5.14: Delay for 50 nodes

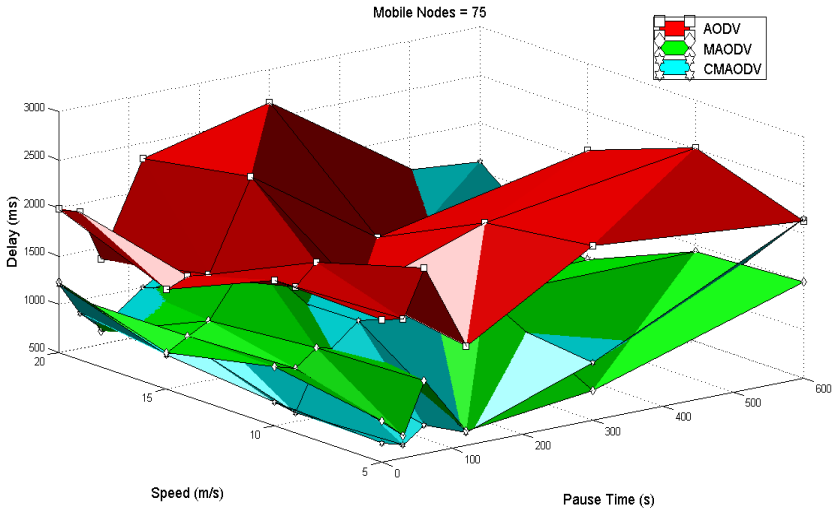

Figure 5.15: Delay for 75 nodes

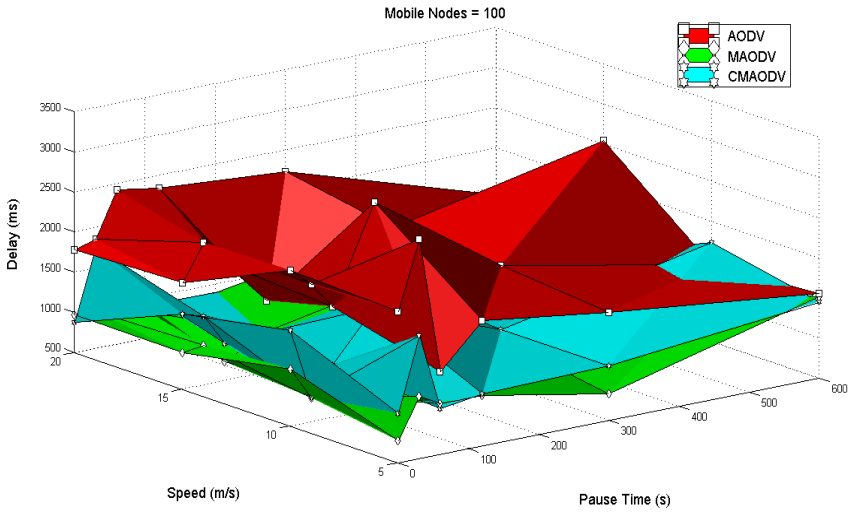

Figure 5.16: Delay for 100 nodes

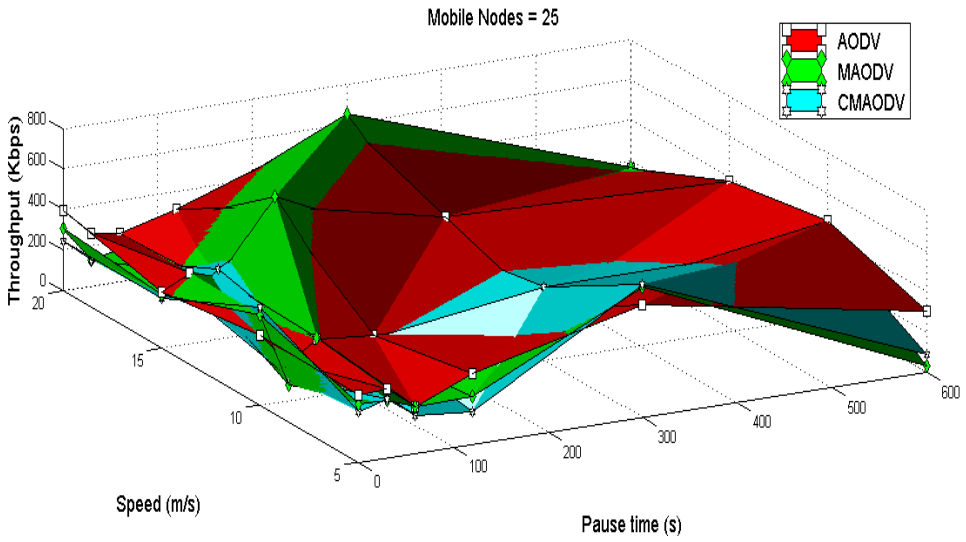

Figure 5.17: Throughput for 25 nodes

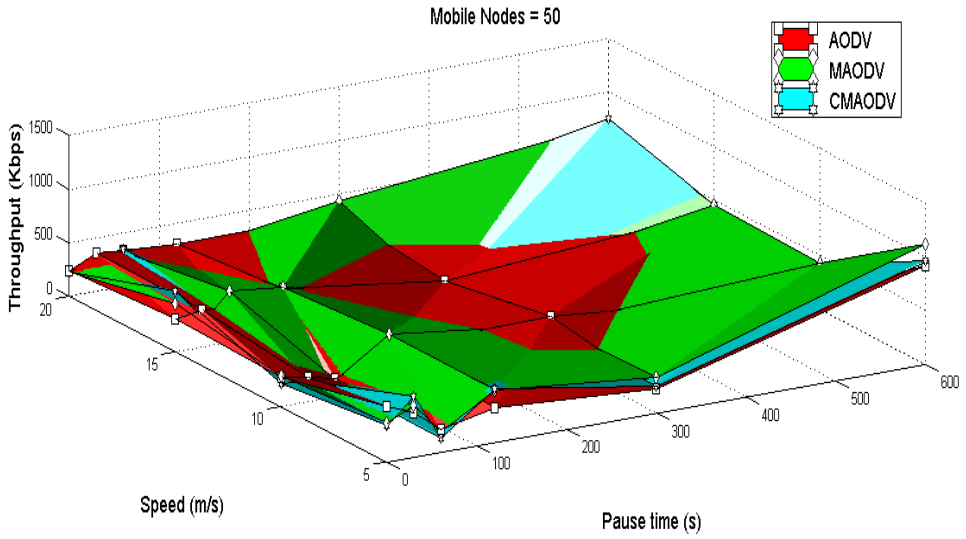

Figure 5.18: Throughput for 50 nodes

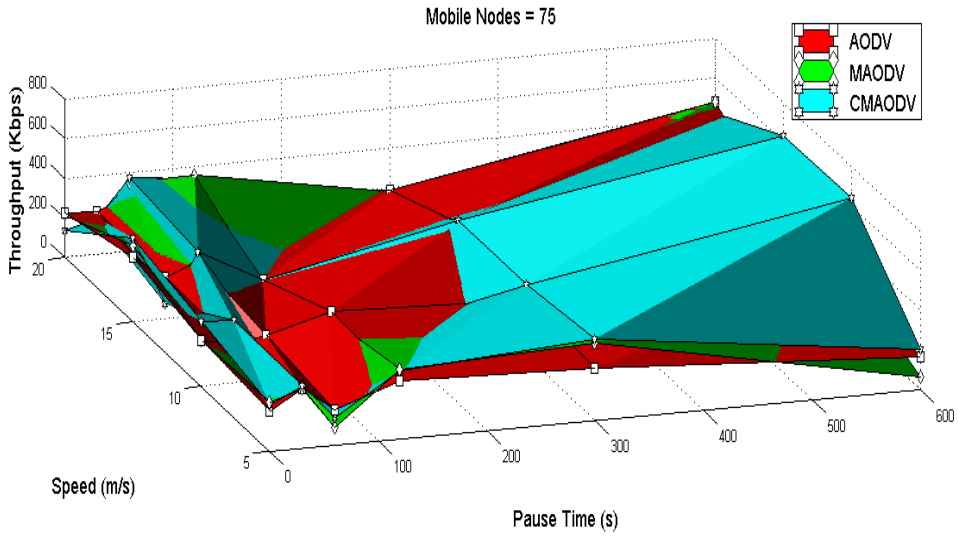

Figure 5.19: Throughput for 75 nodes

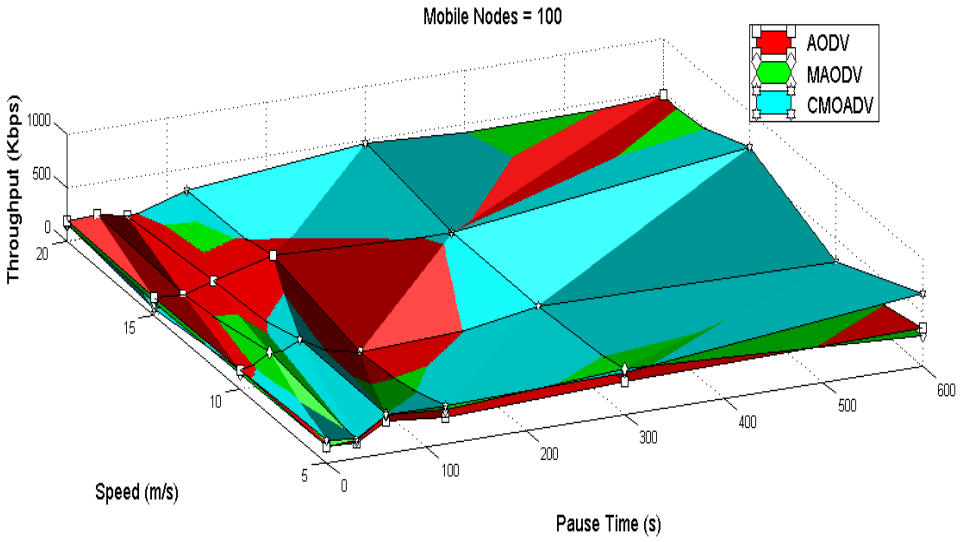

Figure 5.20: Throughput for 100 nodes

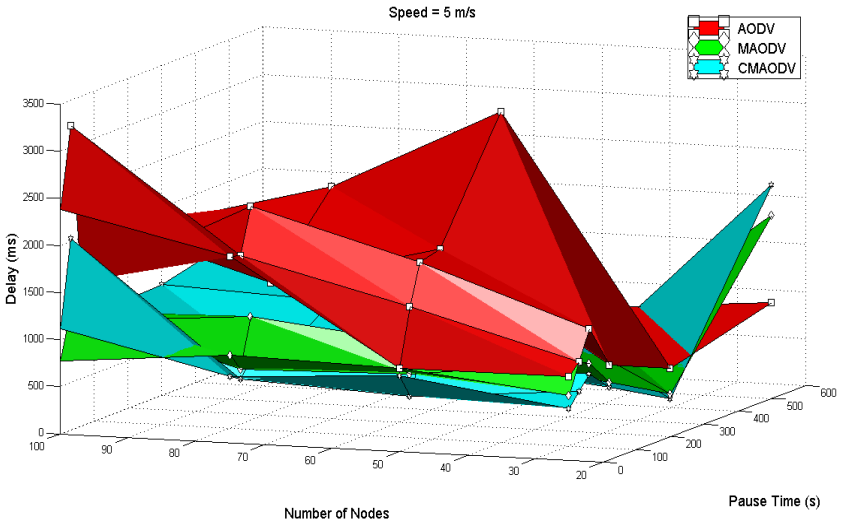

Figure 5.21: Delay for speed 5 m/s

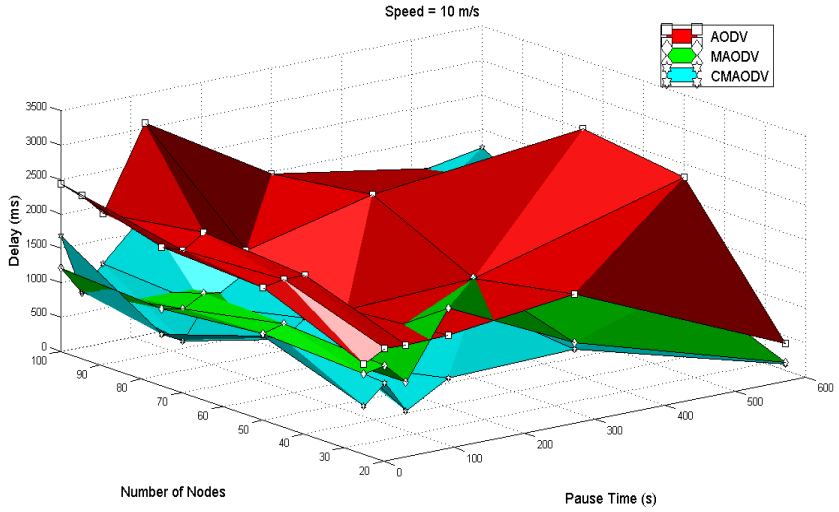

Figure 5.22: Delay for speed 10 m/s

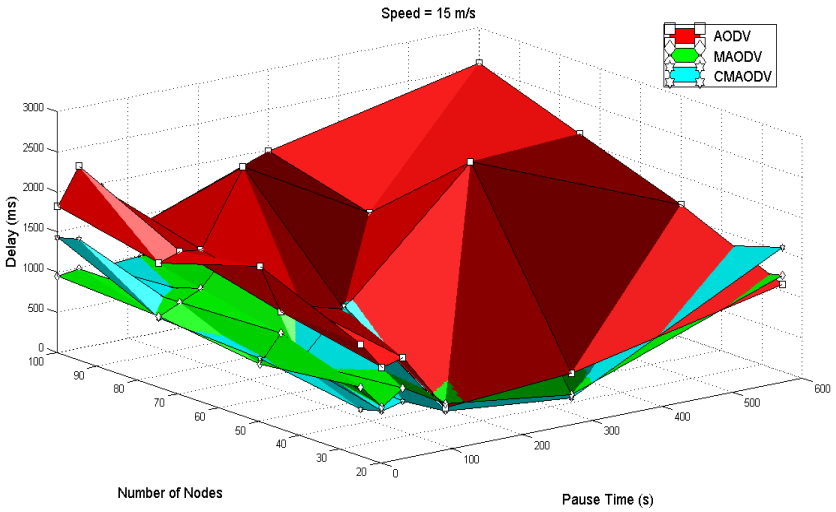

Figure 5.23: Delay for speed 15 m/s

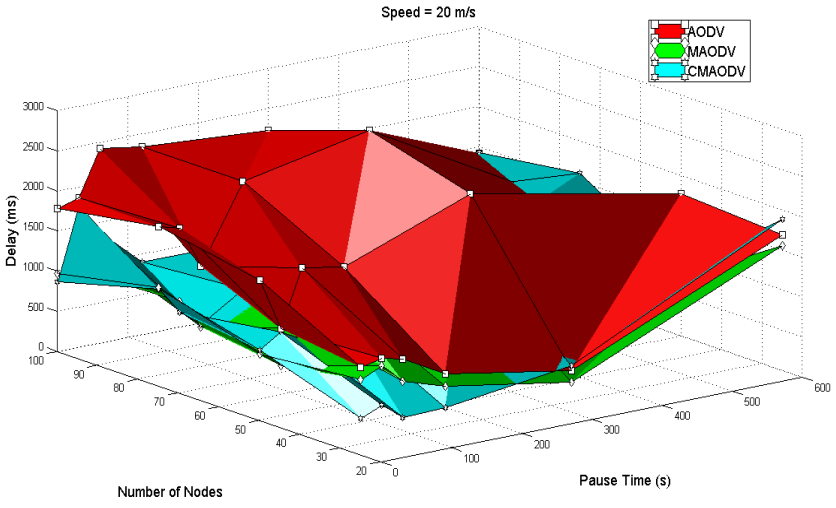

Figure 5.24: Delay for 20 nodes

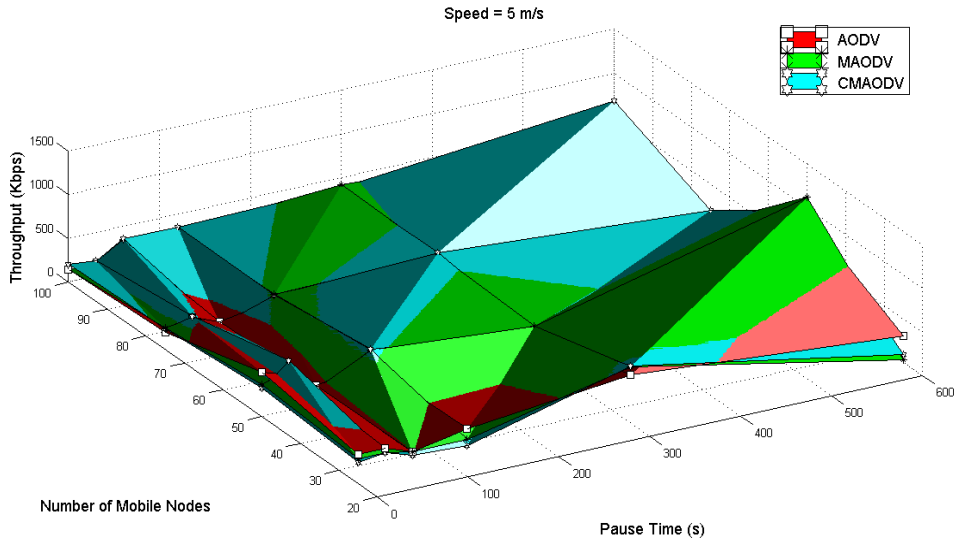

Figure 5.25: Throughput for speed 5 m/s

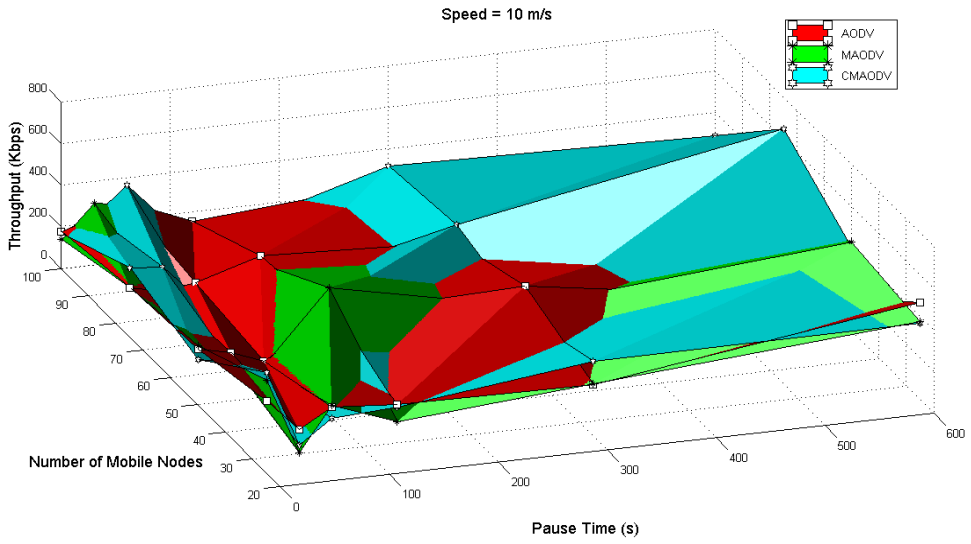

Figure 5.26: Throughput for speed 10 m/s

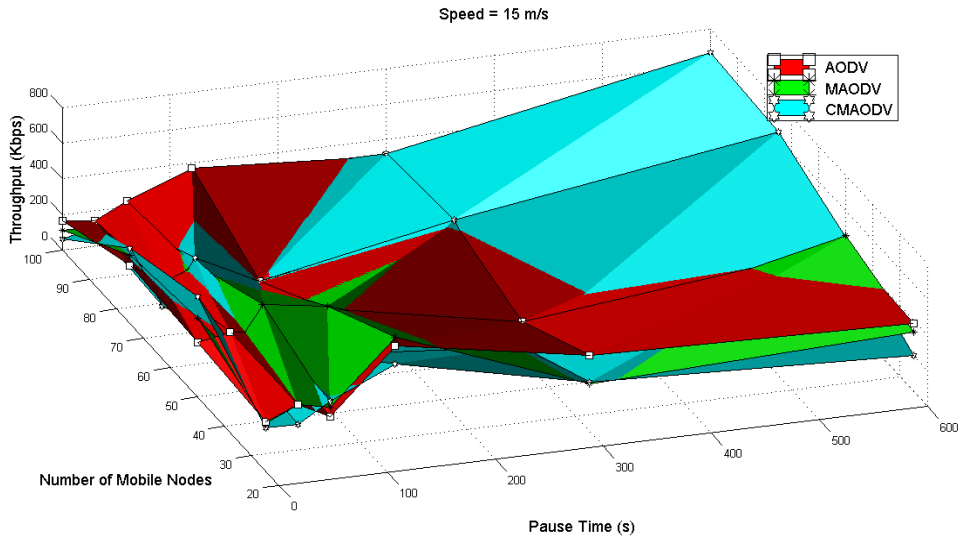

Figure 5.27: Throughput for speed 15 m/s

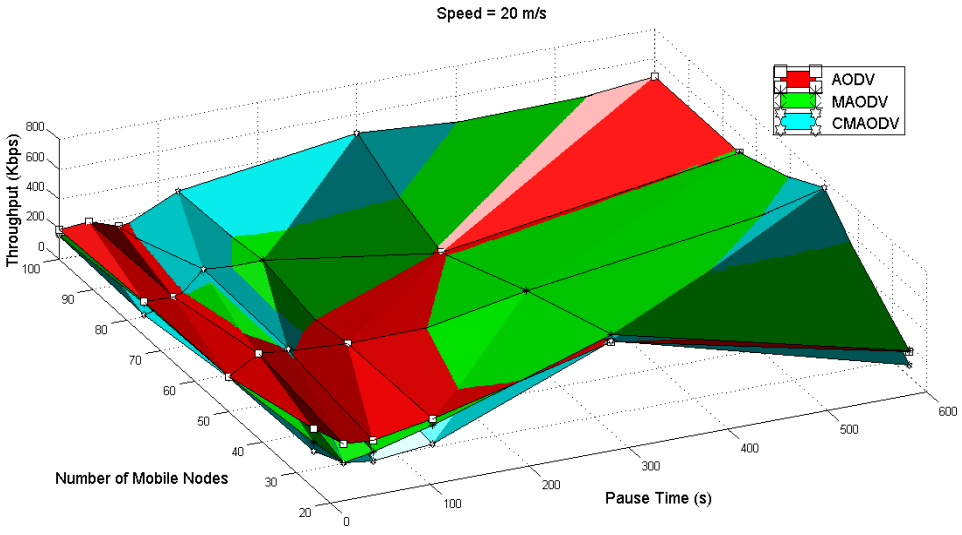

Figure 5.28: Throughput for speed 20 m/s

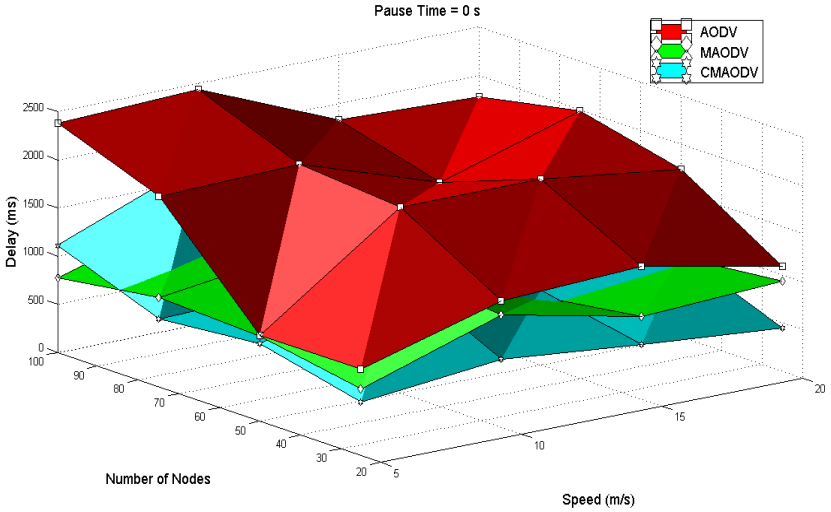

Figure 5.29: Delay for 0s pause

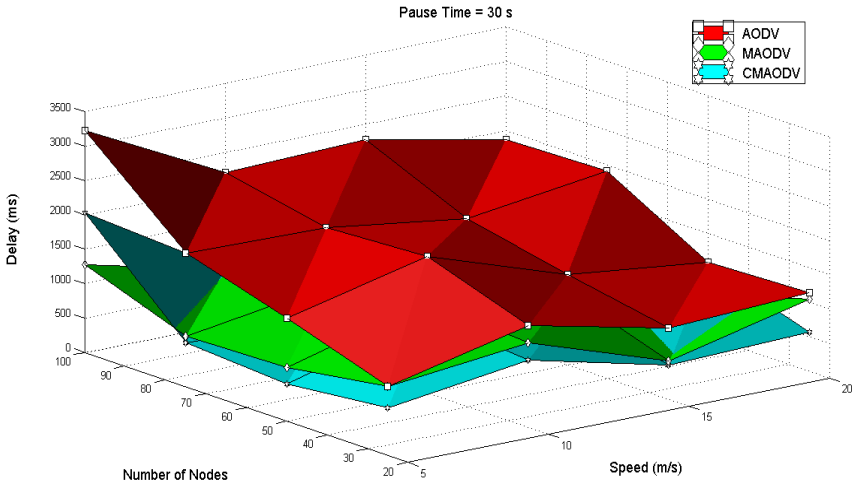

Figure 5.30: Delay for 30s pause

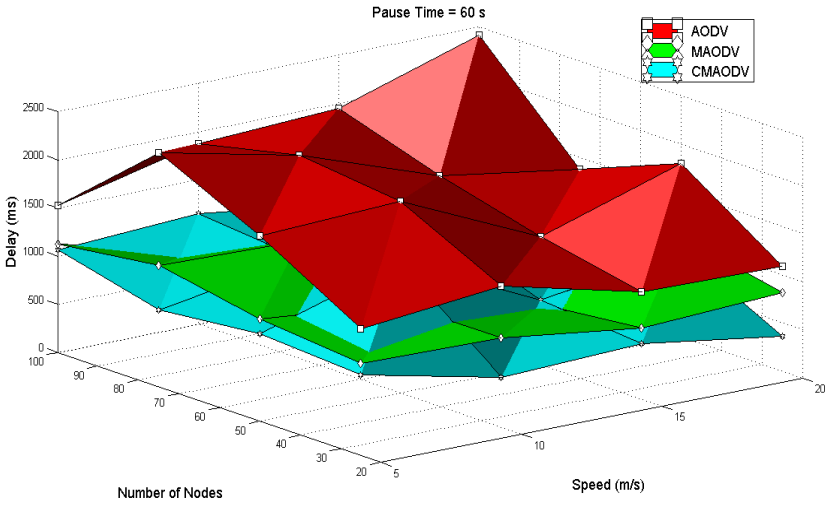

Figure 5.31: Delay for 60s pause

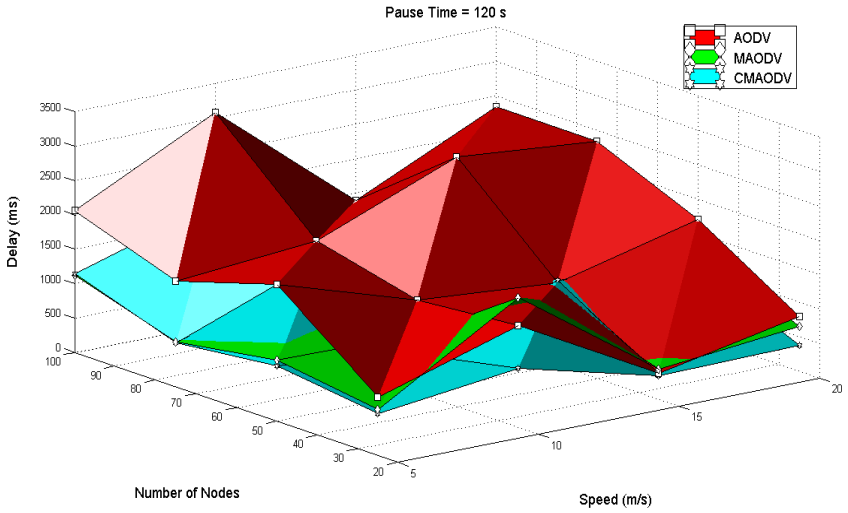

Figure 5.32: Delay for 120s pause

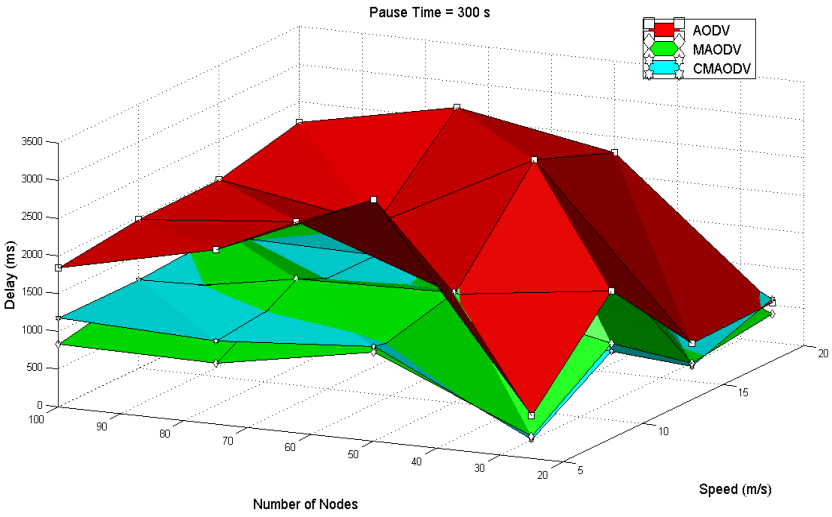

Figure 5.33: Delay for 300s pause

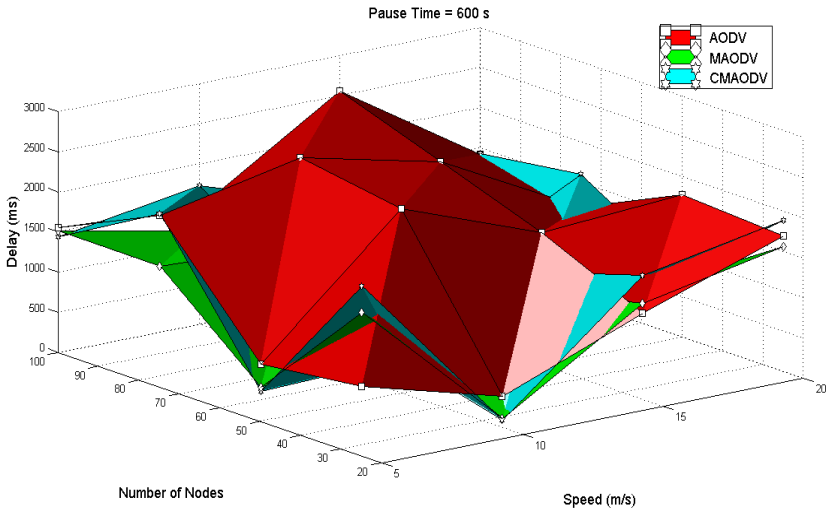

Figure 5.34: Delay for 600s pause

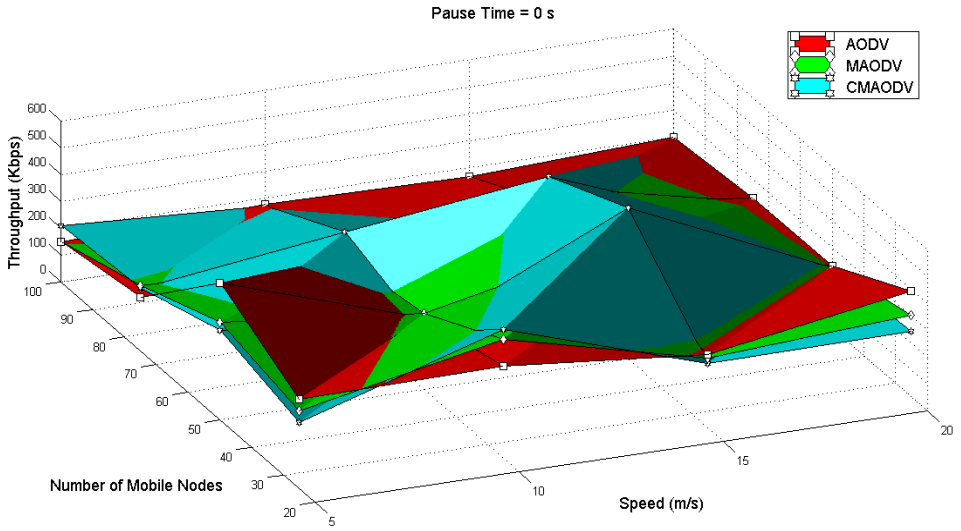

Figure 5.35: Throughput for 0s pause

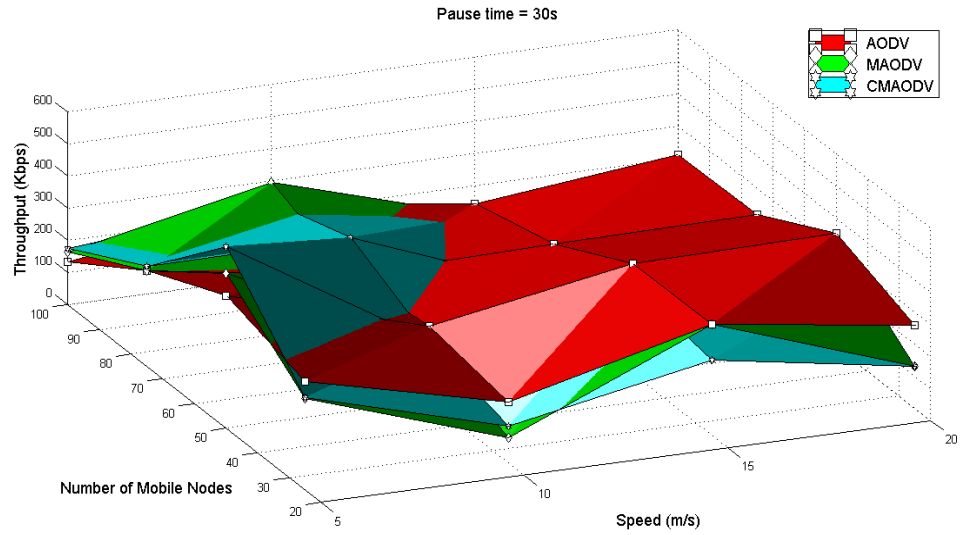

Figure 5.36: Throughput for 30s pause

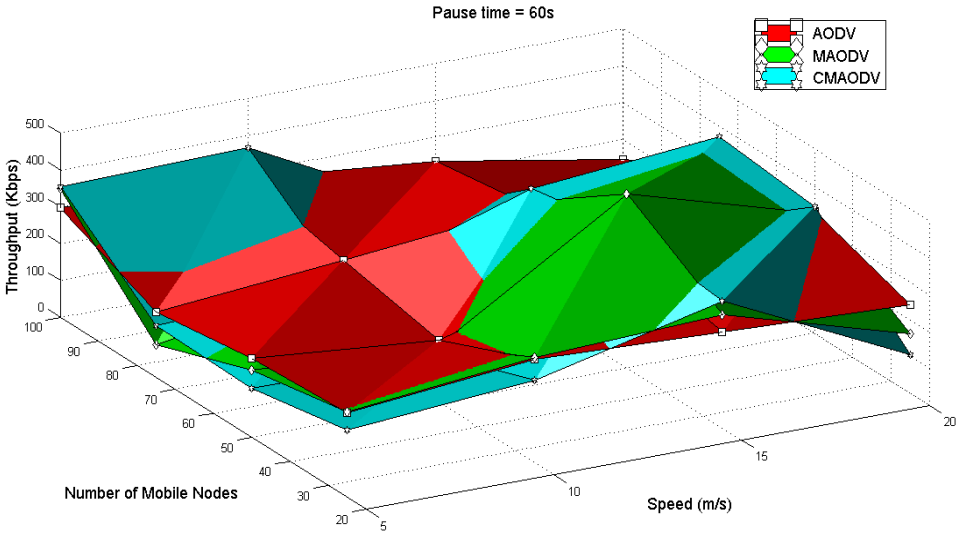

Figure 5.37: Throughput for 60s pause

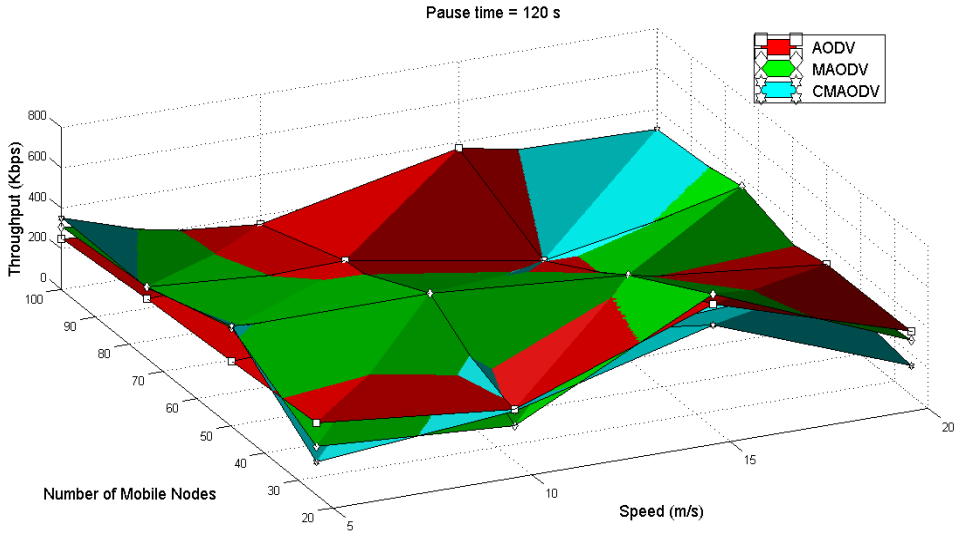

Figure 5.38: Throughput for 120s pause

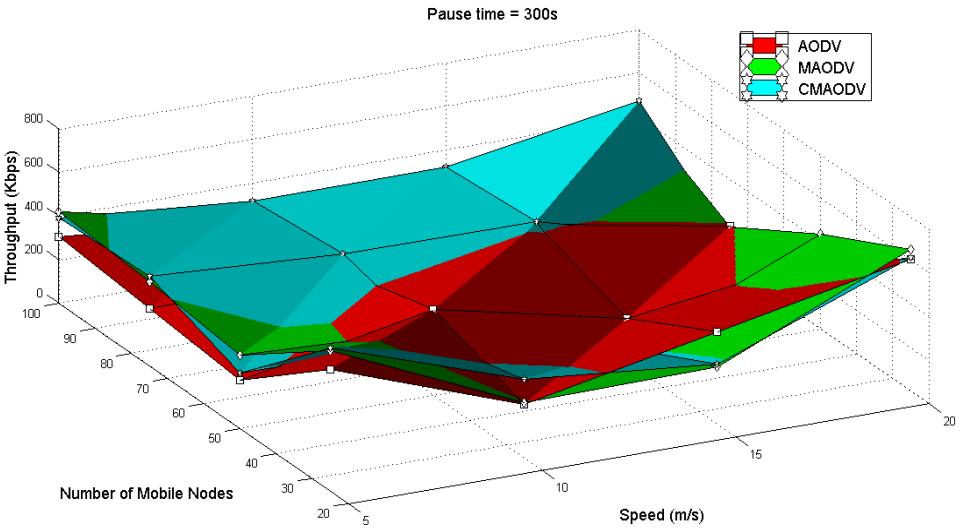

Figure 5.39: Throughput for 300s pause

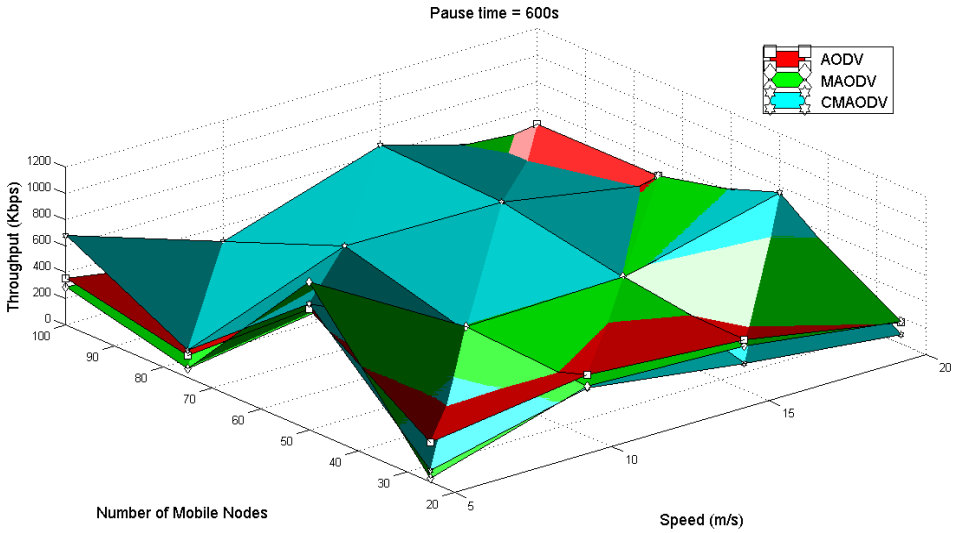

Figure 5.40: Throughput for 600s pause

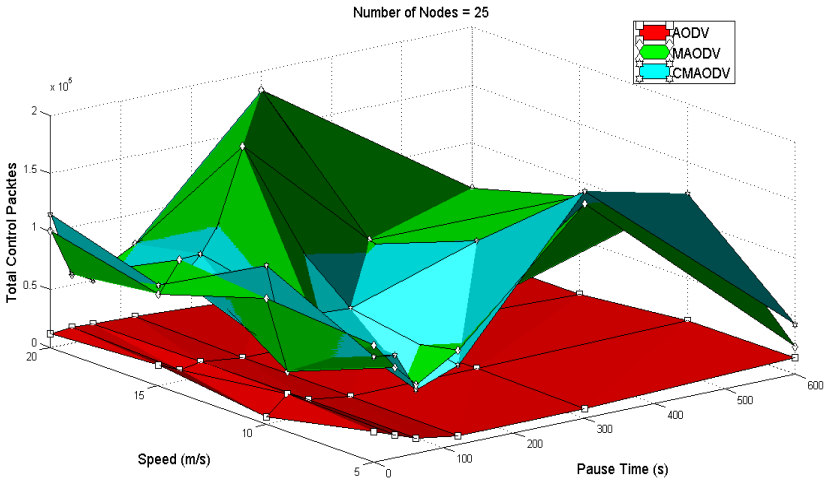

Figure 5.41: Total Control Packets for 25 nodes

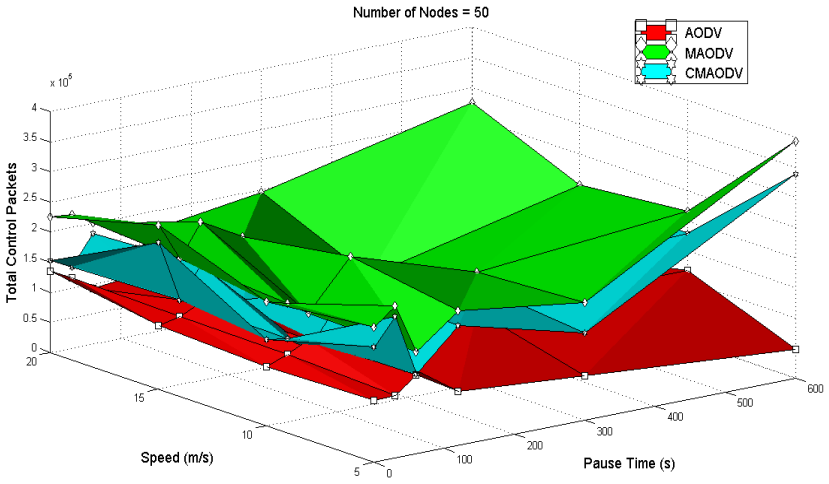

Figure 5.42: Total Control Packets for 50 nodes

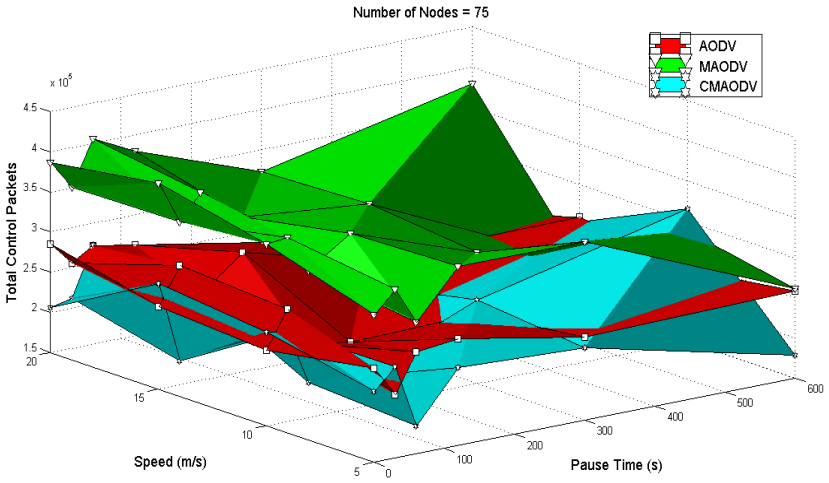

Figure 5.43: Total Control Packets for 75 nodes

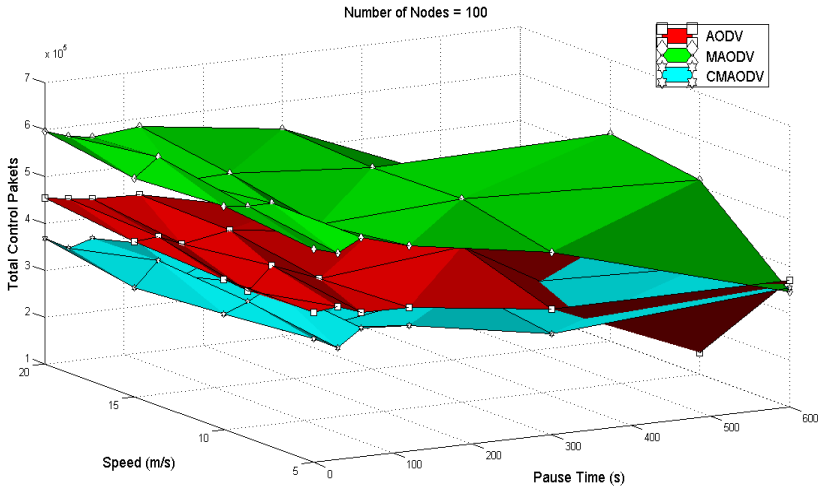

Figure 5.44: Total Control Packets for 100 nodes

5.5.5 Result Analysis

- From figure 5.17, 5.18, 5.19 and 5.20 it can be concluded that as the number of nodes increase, i.e as the density in the network increase, CMAODV gives better throughput as compared to AODV and MAODV.

- From figure 5.25, 5.26, 5.27 and 5.28 it can be observed that as the speed of nodes increase, the performance of CMAODV degrades with respect to throughput in comparison to MAODV and AODV.

- From figure 5.35, 5.36, 5.37, 5.38, 5.39 and 5.40 it is observed that as the pause time of the nodes is increased, that is, the mobility of the nodes is reduced; CMAODV gives better throughput than MAODV and AODV.

- From figure 5.13, 5.14, 5.15 and 5.16 it is observed that at lower number of nodes, CMAODV has lowest delay but also throughput is less, while as the number of nodes increase, the delay of CMAODV is more than that of MAODV but less than AODV.

- From figure 5.29, 5.30, 5.31, 5.32, 5.33 and 5.34 it can be concluded that with increasing pause time, the delay of CMAODV increases and is more than that of MAODV but is less than AODV in majority of cases. Only in the cases where there was minimum mobility (refer Figure 5.34), delay of CMAODV was more than AODV.

- From figure 5.21, 5.22, 5.23 and 5.24it has been observed that with increase in the speed, the delay of CMAODV becomes more than that of MAODV but is still less than AODV.

- From figure 5.41 it can be observed that the control packets that are generated by CMAODV is higher than that of both MAODV and AODV.

- While figure 5.42 shows that for 50 nodes, the contorol overhead for CMAODV is less than that of MAODV but more than AODV.

- Figure 5.43 suggests that for 75 nodes, at lower mobility, the control over head of CMAODV is higher than that of both MAODV and AODV but at higher mobility, the control overhead is lower than both MAODV and AODV.

- For 100 nodes, figure 5.44 CMAODV has lowest control over head for all the scenarios.
- Thus from figures 5.41, 5.42, 5.43 and 5.44 it can be concluded that for high density of nodes, control overhead of CMAODV is much less than that of MAODV and AODV.

Chapter 6

Conclusions and Future Scope of Work

6.1 Conclusion

The aim of the proposed protocol CMAODV was to increase the throughput and reduce the control overhead. From the results of the simulation it is observed that in CMAODV, as the density of nodes increase, throughput increases. From the results, it is seen that when the number of nodes in a 1000m x 1000m area are above 50 nodes, protocol provides higher throughput with considerable lower control overhead in comparison to both MAODV and AODV. Using the proposed protocol, delay is less than unicast routing protocol but is slightly higher than that of MAODV. Thus, CMAODV provides better results when density of nodes is high, making it highly scalable.

6.2 Future Scope of Work

From the results of the simulation it is observed that yet there is scope of improvement with respect to certain aspects of current protocol. The interval at which hello packets are sent for the maintenance of the clusters plays an important part in the performance of protocol. Ideally the rate of hello packets should change as per the mobility in the topology, but in current protocol it is kept static. Thus in future work, the rate of sending hello packets can be made dynamic.

The protocol, currently uses best effort to transmit the data. But, from the literature survey in chapter 3 it can be observed that the transfer of multimedia traffic will benefit if the routes are established based on certain flow requirements. For this the quality of service(QoS) should be incorporated which provides the routes as per the parameters such as bandwidth. This will reserve the resources and ensure certain quality of service.

If the parameter is bandwidth, then it is possible that there are multiple paths from source to receiver, but, all of them provide only a part of bandwidth that is requested. In this case, if an attempt is made to find a single path which will satisfy the requirement, then none will be found. But in reality the bandwidth in network is present, its just that it is divided among different paths. Solution to this is using multipath scheme. Through this, bandwidth of several paths can be aggregated and the requirement can be satisfied.

Appendix A

List of Websites

http://www.winlab.rutgers.edu/žhibinwu/html/ns2_wireless_scene.htm

http://www.winlab.rutgers.edu/žhibinwu/html/network_simulator_2.html

http://www.isi.edu/nsnam/ns/tutorial

http://ti.tuwien.ac.at/ecs/people/albeseder/simcomp/simcomp

http://www.oldnabble.com

Appendix B

List of Papers Prepared

1. Prepared a paper entitled “A Survey of Multicast Routing Protocols for Mobile Ad Hoc Networks”.

2. Prepared a paper entitled “ Cluster based Multicast Ad Hoc On Demand Routing Protocol”.

References

[1] K. Bur, *Quality Of Service Aware Multicast Routing For Multimedia Applications In Mobile Ad Hoc Networks*. PhD thesis, Bogazici University, 2006.

[2] C. W. paper, "Cisco visual networking index: Global mobile data traffic forecast update, 20102015," 2011.

[3] O. S. Badarneh and M. Kadoch, "Multicast routing protocols in mobile ad hoc networks: A comparative survey and taxonomy," *EURASIP Journal on Wireless Communications and Networking*, 2009.

[4] D. S. Shashank Khanvilkar, Faisal Bashir and A. Khokhar, "Multimedia networks and communication," *The Electrical Engineering Handbook*, 2004.

[5] F. Halsall, *Multimedia Communications*. Pearson Education, 2005.

[6] K. Kant and L. K. Awasthi, "Unicast and multicast routing protocols for manets: A comparative survey," *International Journal of IT and Knowledge Management (IJITKM)*, 2010.

[7] M. A. Ali and A. E.-S. I. Z. Morsi, "A survey of multicast routing protocols for ad-hoc wireless networks," *Minufiya Journal of Electronic Engineering Research (MJEER)*, vol. 17, July 2007.

[8] R. S. Prasun Sinha and V. Bharghavan, "Mcedar-multicast core-extraction distributed ad hoc routing," *Wireless Communications and Networking Conference*, vol. 3, 1999.

[9] G. G. Toh, C.-K. and S. Bunchua, "Abam: On-demand associativity-based multicast routing for ad hoc mobile networks," *Vehicular Technology Conference*, 2000.

[10] C. Wu and Y. Tay, "Amris: a multicast protocol for ad hoc wireless networks," *Military Communications Conference Proceedings*, 1999.

[11] A. M. Jason Xie, Rajesh R. Talpade and M. Liu, "Amroute: ad hoc multicast routing protocol," *Mobile Networks and Applications*, vol. 7, 2002.

[12] M. Ching-Chuan Chiang, Gerla and L. Zhang, "Adaptive shared tree multicast in mobile wireless networks," *Global Telecommunications Conference*, vol. 3, 1998.

[13] J. B. K. Tomochika Ozaki and T. Suda, "Bandwidth-efficient multicast routing for multihop, ad-hoc wireless networks," *INFOCOM 2001, 20th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings*, 2001.

[14] B. S. M. Subir Kumar Das and C. S. R. Murthy, "A dynamic core based multicast routing protocol for ad hoc wireless networks," *MobiHoc '02 Proceedings of the 3rd ACM international symposium on Mobile ad hoc networking and computing*, 2002.

[15] L. Ji and M. S. Corson, "Differential destination multicast-a manet multicast routing protocol for small groups," *INFOCOM 2001. 20th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings*, 2001.

[16] M. G. Ching-Chuan Chiang and L. Zhang, "Forwarding group multicast protocol (fgmp) for multihop, mobile wireless networks," *Cluster Computing*, vol. 1, 1998.

[17] K. S. D. G. V. V.Mallaiah, Dr.A.VinayaBabu and A.Sreelatha, "Multicast zone routing protocol in wireless mobile ad hoc networks," *International Journal of Engineering Science and Technology*, vol. 2, 2010.

[18] S. Lee and C. Kim, "Neighbor supporting ad hoc multicast routing protocol," *MobiHoc '00 Proceedings of the 1st ACM international symposium on Mobile ad hoc networking and computing*, 2000.

[19] M. G. Sung-Ju Lee and C.-C. Chiang, "On-demand multicast routing protocol," *Wireless Communications and Networking Conference*, vol. 3, 1999.

[20] E. M. Royer and C. E. Perkins, "Multicast operation of the ad-hoc on-demand distance vector routing protocol," *ACM*, 1999.

[21] M. R. M. A. Al-Sakib Pathan, Muhammad Monowar and C. Hong, "Namp: Neighbor aware multicast routing protocol for mobile ad hoc networks," *The International Arab Journal of Information Technology*, vol. 5, 2008.

[22] G. H. Zeyad M. Alfawaer and N. Ahmed, "A novel multicast routing protocol for mobile ad hoc networks," *American Journal of Applied Sciences*, vol. 4, 2007.

[23] A.Sabari and K.Duraiswamy, "Ant based adaptive multicast routing (aamrp) for mobile ad hoc network," *International Journal of Computer Science and Information Security*, vol. 6, 2009.

[24] C.-C. Shen and C. Jaikaeo, "Ad hoc multicast routing algorithm with swarm intelligence," *Mobile Networks and Applications*, vol. 10, 2005.

[25] A. R. H. H. Ismail Ghazi Shayeb and A. B. Nasoura, "A survey of clustering schemes for mobile ad-hoc network (manet)," *American Journal of Scientific Research*, 2011.

[26] R. Agarwal and D. M. Motwani, "Survey of clustering algorithms for manet," *International Journal on Computer Science and Engineering*, 2009.

[27] S. Chinara and S. K. Rath, "A survey on one-hop clustering algorithms in mobile ad hoc networks," *Springer*, 2009.

[28] A. J. K. H. Tebbe and P. M. Ruiz, "Qos-aware mesh construction to enhance multicast routing in mobile ad hoc networks," *First International Conference on Integrated Internet Ad Hoc and Sensor Networks (INTERSENSE 2006)*, 2006.

[29] M. D. A. Darehshoorzadeh and M. R. Jahed, "Quality of service support for odmrp multicast routing in ad hoc networks," *Springer*, 2007.

[30] O. K. Aisha-Hassan A. Hashim, Mohammad M. Qabajeh and L. Qabajeh, "Review of multicast qos routing protocols for mobile ad hoc networks," *IJCSNS International Journal of Computer Science and Network Security*, 2008.

[31] C. E. Kaan Bur, "Ad hoc quality of service multicast routing," *Computer Communications*, 2005.

[32] Q. Xue and A. Ganz, "Ad hoc qos on-demand routing (aqor) in mobile ad hoc networks," *Journal of parallel and distributed computing*, 2003.

[33] S. Sajama and Z. J. Haas, "Independent-tree ad hoc multicast routing (itamar)," *Vehicular Technology Conference*, vol. 2, 2001.

[34] Y. W. K. Yuh Shan Chen and T. L. Lin, "A lantern tree based qos multicast protocol with reliable mechanism for wireless ad-hoc network," *Computer Communications and Networks*, 2002.

[35] C. H. F. Adel Ben Mnaouer, Lei Chen and J. W. Tantra, "Ophmr: An optimized polymorphic hybrid multicast routing protocol for manet," *IEEE Transactions On Mobile Computing*, vol. 5, 2007.

[36] L. Layuan and L. Chunlin, "A qos multicast routing protocol for clustering mobile ad hoc networks," *Computer Communications*, vol. 30, 2007.

[37] T.-H. L. Yuh-Shyan Chen and Y.-W. Lin, "A hexagonal-tree tdma-based qos multicasting protocol for wireless mobile ad hoc networks," *Telecommunication System*, September 2007.

[38] J. Pan and R. Jain, "A survey of network simulation tools: Current status and future developments," 2008.

[39] "http://ti.tuwien.ac.at/ecs/people/albeseder/simcomp/simcomp."