ADAMA SCIENCE & TECHNOLOGY UNIVERSITY

SCHOOL OF ENGINEERING

DEPARTMENT OF COMPUTING

CASE BASED AND FUZZY METHODS ON-DEMAND MULTICAST ROUTING PROTOCOL FOR MOBILE AD-HOC NETWORKS

By:

Fikreab Lopiso

SEPTEMBER, 2016

A MASTER'S THESIS

ON

CASE BASED AND FUZZY METHODS ON-DEMAND MULTICAST ROUTING PROTOCOL FOR MOBILE AD-HOC NETWORKS

Submitted

in partial fulfillment of the requirement for the award of the degree of

MASTER OF SCIENCE IN SOFTWARE ENGINEERING

By:

Fikreab Lopiso

Advisor(s):

Mr. Dileep Kumar G.

SEPTEMBER, 2016

A MASTER'S THESIS

ON

CASE BASED AND FUZZY METHODS ON-DEMAND MULTICAST ROUTING PROTOCOL FOR MOBILE ADHOC NETWORKS

Submitted

in partial fulfillment of the requirement for the award of the degree of

MASTER OF SCIENCE IN SOFTWARE ENGINEERING

By:

Fikreab Lopiso

APPROVAL BY BOARD OF EXAMINERS

Chairman Department of Graduate Committee	Signature
Mr. Dileep Kumar G.	
Advisor	Signature
Examiner	Signature
External Examiner	Signature

DECLARATION

I, the undersigned, declare that this thesis is my original work and has not been presented as a partial degree requirement for a degree in any other university and that all sources of materials used for the thesis have been duly acknowledged.

Fikreab Lopiso Lachebo

Signature: _____

Date: _____

CERTIFICATION

I, the undersigned, certify that i read and hereby recommend for the acceptance by the Adama Science and Technology University a thesis entitled: "**case based and fuzzy methods on-demand multicast routing protocol for mobile ad-hoc networks**" in partial fulfillment of a degree of masters of Science in software engineering.

Mr. Dileep Kumar G.

(Advisor)

Signature _____

Date _____

Abstract

Mobile Ad-hoc Network is a self-organizing and self-configuring network without the need of any centralized base station. Each node participating in the networks acts as host as well as router in order to transmit, receive and forward packets.

Multicast routing in MANETs is mainly used for group oriented computing, which is an efficient method to lead data packets from one source group to several nodes as destination group. It can reduce communication costs, processing overhead and delivery delay. Although multicast routing algorithms in MANETs could be efficient in many situations, but their forwarding structure and network resource consumption makes them significantly less efficient than unicast routing algorithms.

In this thesis work fuzzy logic method is used to create small, strong forwarding group and case based method is used to restrict the domain of control packet flooding so as to reduce the overhead. The combination of above two methods is applied to On-demand multicast routing protocol (ODMRP). An ns-2 simulation study performed and our results revealed that the resultant increases packet delivery rate reduces average end-to-end delay.

Keywords: Mobile Ad-hoc Network (MANET), Multicast Routing Protocols, ODMRP, CFODMRP.

Acknowledgments

First of all I would like to thank almighty God for his blessings. My deepest gratitude should go to my advisor Mr. Dileep Kumar G. for his guidance and advice throughout working on this thesis. He gave me all his knowledge, time, patience and advices.

My deepest gratitude should go to Dr. Henock Mulugeta, this thesis work would not have been possible from the beginning without the enthusiasm and motivation of Dr. Henock Mulugeta in developing the initial concepts of the thesis proposal.

I am greatly honored to thank, and to express my gratitude to my Family and friends from my heart for their unlimited Support during my research work.

Acronyms and Abbreviations

ARP Address Resolution Protocol

CBR Constant Bit Rate

CMU Carnegie Mellon University's Monarch Research Group

E2E End-to-end

IETF Internet Engineering Task Force

LL Link Layer

MANET Mobile Ad-Hoc Network

NAM Network Animator

NS-2 Network Simulator 2

ODMRP On-Demand Multicast Routing Protocol

OTCL Object Oriented Tool Command Language

RREP Route Reply

RREQ Route Request

TCP Transmission Control Protocol

TTL Time to Live

UDP User Datagram Protocol

List of Figures

List of Tables

Table of Contents

CHAPTER ONE: INTRODUCTION

1. 1. General Background

A MANET is a collection of wireless mobile nodes which have the ability to communicate with each other without having fixed network infrastructure or any central base station [1]. Since mobile nodes are not controlled by any other controlling entity, they have unrestricted mobility and connectivity to others. Routing and network management are done cooperatively by each other nodes. Due to limited transmission power, multi hop architecture is needed for one node to communicate with another through network. In this multi hop architecture, each node works as a host and as well as a router that forwards packets for other nodes that may not be within a direct communication range. Each node participates in an ad-hoc route discovery protocol which finds out multi hop routes through the mobile network between any two nodes [2]. These infrastructure-less mobile nodes in ad-hoc networks dynamically create routes among themselves to form own wireless network on the fly. Thus, mobile ad-hoc networks provide an extremely flexible communication method for any place where geographical or terrestrial constraints are present and need network system without any fixed architecture, such as battlefields, and some disaster management situations [3].

Generally, the followings are some the challenges and issues, which are specific to MANET [1]:

1. Autonomous: There is no centralized administrative entity to manage the operation of different mobile nodes.

2. Device discovery: Identifying newly added nodes require dynamic route update mechanism.

3. Bandwidth optimization: Wireless links have significantly lower capacity than wired links.

4. Scalability: Most of the MANET routing protocols are not scalable for large number of nodes.

5. Limited Physical Security: Mobility implies higher security risks such as eavesdropping, spoofing and Denial-of-Service (DoS) attacks.

6. Infrastructure-less and Self-operated: Self-healing feature demands MANET to realign itself and comprehend any node moving out of its range.

7. Poor Transmission Quality: This is an inherent problem of wireless communication caused by several sources that result in degradation of the received signal.

8. Ad hoc Addressing: Addressing, routing and location management are difficult to implement due to limited resources.

9. High Error Rate: The obstacle in the route causes problems such as shadowing, reflection, scattering, fading, refraction and diffraction of signals during communication between source and destination. The packets are garbled and received with high error rate because of the above mentioned problems.

10. Low Data Rate: Nodes in MANET perform well in short range communication. A higher frequency bandwidth can transmit more data, but is vulnerable to interference.

11. Dynamic Topology and Scalability: Ad hoc networks do not allow the same kind of aggregation techniques that are available to standard Internet routing protocols. Hence they are vulnerable to scalability issues.

12. Security: As MANETs are distributed and dynamic without much infrastructure and centralized monitoring, they are vulnerable to various security threats.

13. Energy limitation: MANET allows mobile nodes to operate with battery power. Nodes in MANET should be energy efficient and operate with limited processing and memory resources.

The objective of a multicast routing protocol for MANET [4] [5] [6] [7] is to support the dissemination of information from a sender to all the receivers of a multicast group while trying to use the available bandwidth efficiently in the presence of frequent topology changes. Multicasting is an interesting and important communication paradigm as it models several application areas via subscription services (news groups, TV, radio), collaboration or conferencing services (eg. virtual conferencing) etc. In an ad-hoc environment, hosts generally co-operate as a group to achieve a given task, thus the MANET model is a suitable environment for the multicast paradigm. Also the multicast model improves network utilization through mass data distribution, which is ideal for bandwidth constrained networks like MANETs. Therefore multicast communication is very important in ad-hoc networks.

1.2. Statement of the problem

According t0 [5] [6] [7] [8] [9] [10] Multicasting has many benefits such as reduce communication costs, reduce processing overhead and delivery delay.

The inherent characteristics to MANET presented in previous above section 1.1. Possess even series challenges to provide efficient multicast routing protocol for MANETs.

Multicast routing algorithms [8] [9] [10] [11] [12] [13] have become increasingly important in the field of wireless ad-hoc networks, because they enable the distribution of data to a potential large set of nodes. Nodes form a multicast delivery structure which in normal cases performs better than using multiple unicast routing paths. This is crucial in ad-hoc environments, where bandwidth and power resources are at a premium. The major impediment, however, is that nodes can move randomly, possibly causing frequent and unpredictable topology change. Multicast network algorithms, however, must deliver packets to several hosts simultaneously, which then must discern if their role is to receive or forward the packets towards the multicast sinks. Although multicast routing algorithms are desirable in many situations, it is crucial to optimize their forwarding mechanism and network resource consumption.

Therefore, this thesis should explore and answer the following research questions.

RQ1: What factors affect the performance multicast routing protocols for MANETS?

RQ2: What techniques are used to improve the performance of multicast routing protocols for MANETS?

RQ3: Is the proposed protocol performs better than existing on-demand multicast routing protocol (ODMRP)?

1.3. Objectives

1.3.1. General Objective

The general objective of this thesis work is to propose an integrated case based-fuzzy logic method on-demand multicast routing protocol for mobile ad-hoc networks.

1.3.2. Specific Objectives

In order to achieve the general objective, the following specific objectives are identified.

- To identify what characteristics reduces the performance of on-demand multicast routing protocols for mobile ad-hoc networks.
- To propose a solution how to enhance the performance of on-demand multicast routing protocols for mobile ad-hoc networks.
- To simulate the proposed routing protocol using NS-2, a network simulator and analyze the performance.
- To discuss and report experimental results found and recommend further research works in this area.

1.4. Methodology

The strategy adopted for the thesis is described as follows.

Literature survey in the area

Related literatures such as Books, Journals articles, conference papers and additional useful materials from internet should be reviewed. This should help to investigate the underlying principles and theories, methodologies, techniques and tools of the various methods that can be adopted, manipulated and used for synthesizing and gaining a new perspective in to what has been done in previous researches in the area and what can be done in this research.

Analysis and modeling

The performance of existing on-demand multicast routing protocol (ODMRP) and proposed protocol case based fuzzy logic method on-demand multicast routing protocol (CFODMRP) can be analyzed.

Mechanism of deriving conclusion

Based on performance measure parameters analyzed to conclude that the proposed protocol has increases its performance or not.

Tools available to compare the metrics

There are different tools available to compare performance evaluation of different multicast routing protocols for mobile ad-hoc networks. In this thesis, NS-2 is used.

1.5. Contribution of the Work

The major contribution of the thesis is to propose case based fuzzy logic method on-demand multicast routing protocol for mobile ad-hoc networks with better performance by changing the way how the multicast forwarding group is created, and limiting the domain of control packet flooding, by using previous route information. So that this work should have an immense effect on future researches on multicast routing protocol for mobile ad-hoc networks in improving their performance by understanding the context of multicast forwarding group and network wide flooding of control packets. At the end of this work, the result should help future researchers that aim to work on improving performance of multicast routing protocol for mobile ad-hoc networks and performance evaluation of multicast routing protocol for mobile ad-hoc networks.

1.6. Organization of the Thesis

In Chapter two briefly describes Literature Review on what is MANET, and some characteristics of MANETs and some application areas. Moreover it describes routing protocols categories in general and multicast routing as specific and lastly it describes what are the factors affecting performance of multicast routing and techniques to overcome such problems.

Changes that lead to the design of the case based fuzzy logic method (CFODMRP) protocol are discussed in Chapter three. In Chapter four we present an implementation and evaluation of our approach using NS-2 simulator, where we compare CFODMRP with standard ODMRP. In chapter five discusses the results. Finally Chapter six discusses conclusion and future Work. Here the main points in the research should be discussed. In addition recommendations along with future research ideas are discussed.

CHAPTER TWO: REVIEW OF LITERATURE

2.1. Mobile Ad-hoc Networks (MANETs)

A computer network allows sharing of resources and information among different devices connected to the network [14]. Computer networks can be classified according to the hardware and software technology that is used to interconnect the individual devices in the network, such as Optical-Fiber, Ethernet, Wireless Local Area Network (WLAN), MANET etc. Ethernet uses physical wiring to connect devices. Frequently deployed devices include hubs, switches, bridges and/or routers. Wireless LAN technology is designed to connect devices without wiring. These devices use radio waves or infrared signals as a transmission medium. MANETs also use radio waves or infrared signals as a transmission medium but unlike Wireless LANs, MANETs have no access points. Access point is a device that allows wireless communication devices to connect to a network using Wi-Fi, Bluetooth or related standards. Lack of access points forces mobile nodes in ad-hoc network to function as both host and routing device.

Mobile ad-hoc network (MANET) is a self-organizing collection of wireless mobile nodes that form a temporary and dynamic wireless network established by a group of mobile nodes on a shared wireless channel without the aid of a fixed networking infrastructure or centralized administration. A communication session is achieved either through single-hop transmission if the recipient is within the transmission range of the source node, or by relaying through intermediate nodes otherwise. For this reason, MANETs are also called multi-hop packet radio networks. However, the transmission range of each low-power node is limited to each other's proximity, and out-of-range nodes are routed through intermediate nodes [4] [7] [15].

A routing protocol in ad-hoc networks has two phases (Request phase and Reply Phase) to establish connection between source and receiver pair and once the connection is established a data packet is forwarded on that path.

An efficient routing protocol is needed whenever a packet needs to be transmitted to a destination via number of nodes and numerous routing protocols have been proposed for such kind of ad-hoc networks. These protocols find a route for packet delivery and deliver the packet to the correct destination. The studies on various aspects of routing protocols have been an active area of research for many years.

Applications of MANETs:

Because of its speedy and convenient deployment, robustness, and low cost, a MANET can find its applications in the following areas:

1) Military applications
2) Emergency operations
3) Wireless mesh networks
4) Wireless sensor networks

2.2. Multicast Routing in Mobile Ad-hoc Networks

Multicasting is the transmission of one data packets to zero or more receivers in the group, means one copy of same data can be transmit to multiple receivers at a same time (see Figure.2.1[7]). In the multicast, any node in the network can join or leave a multicast group at any time, and any node in the network can send data to any multicast group. The multicast sources do not need to know who the receivers are but need only know the multicast group address [4] [7] [16].

Multicast is usually a separate service from unicast for efficiency reasons. For example, the path from a source to two or more receivers may share one or more links (Figure 2.1) and as a result, sending a separate unicast packet to each receiver would mean that the same packet would traverse each shared link multiple times. Due to this communication cost is reduced and efficiency of wireless channels is also improved [3] [16].

The goal of an efficient multicast routing protocol is to deliver a copy of each packet to each multicast receiver by duplicating the packet in the network as few times as possible (e.g., only at node 2 in Figure 2.1).

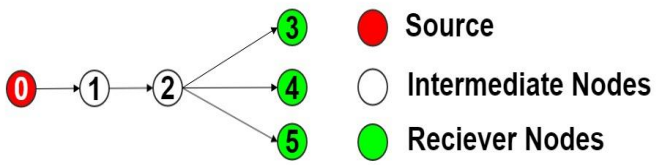

Figure 2.1: Multicasting

There are many benefits of multicasting in MANET's. Some of them are [4] [7][17]:

- Minimize processing power and bandwidth consumption.
- Reduce transmission cost.
- Minimize delivery delay.

2.2.1. Challenges in MANET Multicast Routing

Multicast routing protocols used in ad-hoc networks do not perform well due several reasons, the dynamic communication environment, and characteristics of mobile nodes in the networks described in section 1.1 are the dominant ones [3] [4] [16] [17].

I. Dynamic communication environment: In mobile ad-hoc networks, nodes can move randomly. Due to the dynamic communication environment the network topology, which is typically multi-hop, can change frequently and unpredictably, resulting in: -
   1- Route changes
   2- Repeated reconstruction of route path and
   3- Possibly packet losses.

II. Characteristics of mobile nodes in the network: In mobile ad-hoc networks, nodes are equipped with limited resources such as nodes operate by limited capacity batteries and mobility of modes. Due to these nodes processing power is limited, resulting in: -
   1- Route failure: In MANETs most of the time route should be failed. The main causes of route failures are node mobility and power constraints. The route reestablishment duration after route failure in Ad-hoc networks depends on the underlying routing protocol, mobility pattern of mobile nodes, and traffic characteristics.
   2- Route Reconstruction: In MANETs most of operation is route establishing due to the dynamic nature of mobile nodes. This route reconstruction or reestablishing consumes a lot of these limited mobile nodes resources which in turn create an overhead in a given routing protocol.

2.2.2. Classification of multicast routing protocols in Mobile Ad-hoc Networks

To compare and analyze mobile ad-hoc network multicast routing protocols, appropriate classification methods are important. Classification methods help researchers and designers to understand distinct characteristics of a routing protocol and find its relationship with others. These characteristics are mainly related to the information which is exploited for routing, when this information is acquired, and the roles which nodes may take in the routing process.

Classifications of multicast routing protocols are shown in figure 2.2 [18].

Based on Topology:

Based on how delivery structures among group members are constructed there are three categories [10] [16] [17] [18]-Tree Based Mesh Based and Hybrid Based see figure 2.3[18].

I. **Tree Based**- In tree-based multicast routing protocols there is only one path between source and receiver. Tree based routing protocols are not robust to operate in highly mobile environment. A tree based multicast routing protocol establishes and maintains a shared multicast routing tree to deliver data from a source to receivers of a multicast group. Tree Based multicast routing protocol is further divided into two types-Source Tree Based and Shared Tree Based.
    1. **Source Tree Based**-In Source Tree Based each source node maintains a separate tree and all the receivers lies under this node.
    2. **Shared Tree Based**- In this a single tree is shared by all the sources in multicast group

II. **Mesh Based** - In mesh based multicast routing protocols there is more than one path between source and receiver pair. If any link failure occurs then these redundant paths are very useful, and provide higher packet delivery ratio.

III. **Hybrid Based** - In Hybrid based multicast routing protocols the combination of the two methods described above are used when needed.

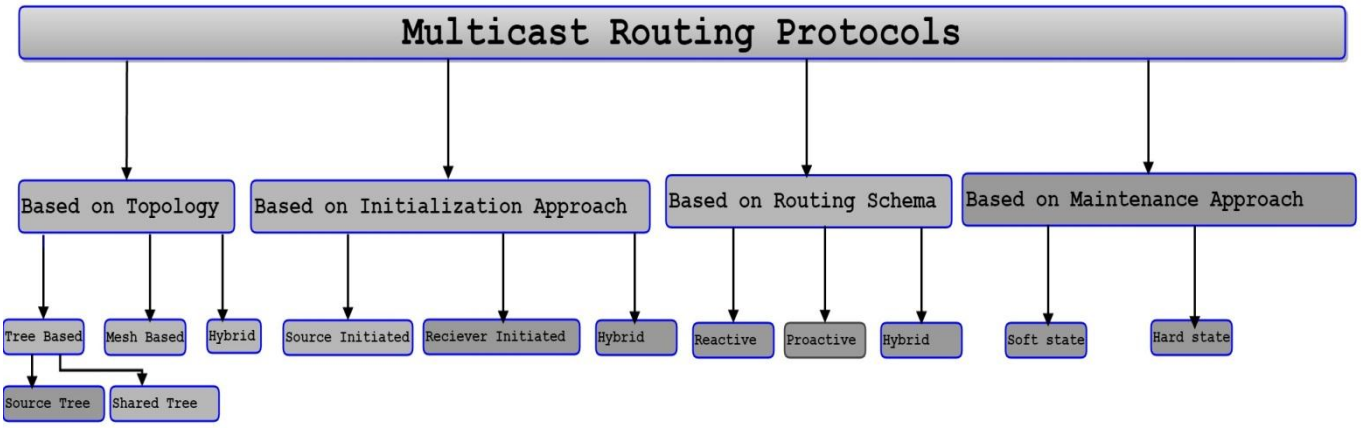

Figure 2.2: Classification of Multicast Routing Protocols

Based on Initialization approach

Based on how multicast connectivity is established and maintained there are three approaches, namely Source Initiated, Receiver Initiated and Agent/Hybrid approach [4] [16] [17] [18].

I. **Source Initiated**- In Source Initiated the formation of multicast group is started by source node.

II. **Receiver Initiated**- In Receiver Initiated the formation of multicast group is started by receivers of multicast group.

III. **Hybrid approach**- In this construction and maintenance of multicast group done by either source node or receiver node. This approach basically combination of these two approaches i.e. Source initiated and receiver initiated.

Based on Routing scheme

Based on how routing information is acquired and maintained by mobile nodes there are three approaches, namely -Reactive, Proactive and Hybrid approach [4] [16] [17] [18].

I. **Reactive approach**- Reactive approach also known as the on - demand approach. In this route is established on the basis of demand. When a node wants to communicate with other node and there is no route, then these routing protocols should try to establish the route.

II. **Proactive approach**- Proactive approach also known as the table driven approach, in which each node maintain the network topology information in the form of routing tables. And these tables are periodically exchange to get the up to date view of the network.

III. **Hybrid approach**- Hybrid approach is the combination of both these approaches. It is designed to overcome the limitation of these approaches (Proactive and Reactive approaches).

Based on Maintenance approach

There are two main approaches which are used for maintenance of multicast topology. This can be done by either soft state approach or either hard state approach [4] [16] [17] [18].

I. **Soft- State(SS)** approach is also known as "connectionless approach" .In this the control packets are flooded periodically to refresh the route, which cause a high packet delivery ratio at the cost of more control overhead.

II. **Hard- State (HS)** approach is also known as "connection oriented approach". In this the control packets are only transmitted when a link breaks, due to this low control overhead but at the cost of low packet delivery ratio.

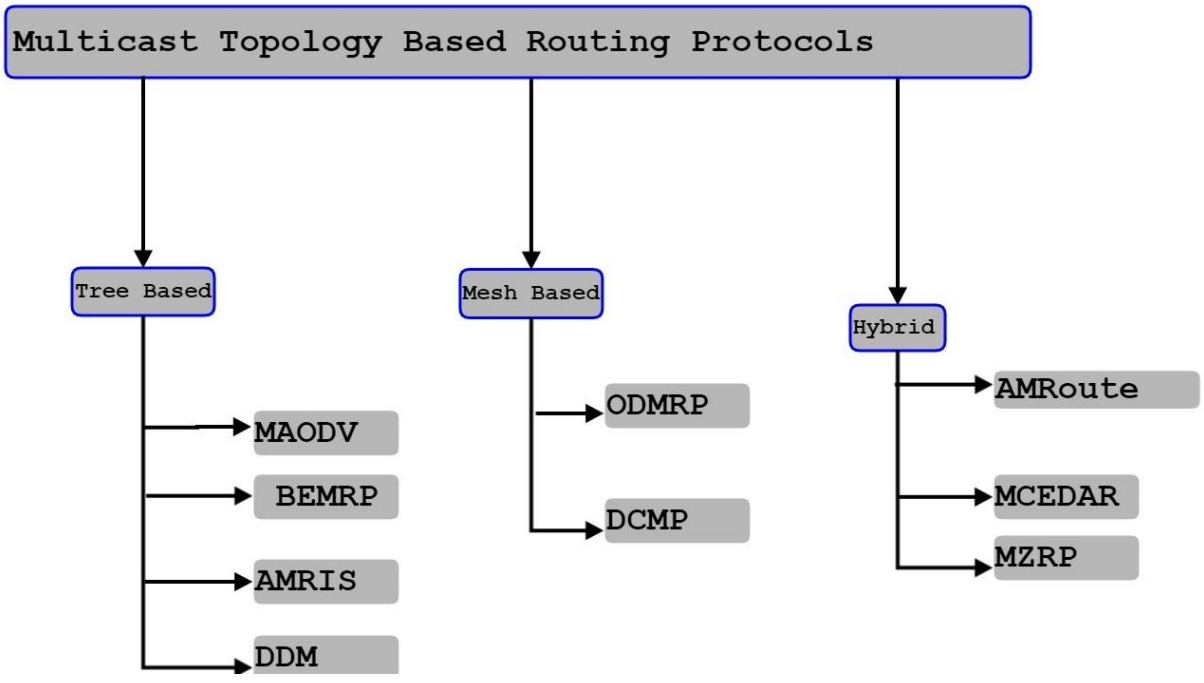

Figure 2.3: Classification of Topology based Routing Protocols

2.3. On-Demand Multicast Routing Protocol (ODMRP)

On-Demand Multicast Routing Protocol (ODMRP) [19] [20] [21] is one of the more popular multicast routing protocols for mobile ad-hoc networks.

2.3.1. Background

ODMRP [19] [20] [21] was developed by the Wireless Adaptive Laboratory of University of California, Los Angeles. The unique features of ODMRP are,

I. Forwarding tree concept where the protocol forms a subset of collaborating nodes into a packet forwarding tree.
II. Mesh based forwarding tree to avoid drawbacks of traditional tree based multicast protocols (i.e., frequent tree re-configuration, traffic concentration, non-shortest path, intermittent connectivity). ODMRP forms multiple potential forwarding paths, which enable multicast receivers to discover better routes.
III. Soft state based forwarding concept to avoid explicit messaging for setting up, tearing down and maintaining of multicast groups.

2.3.2. Definitions

I. **Join Query**

   A control packet sent by the multicast sender to initiate a multicast communication. In addition, the multicast source sends Join Query packets periodically to refresh the forwarding path.

II. **Join Table**

   A control packet sent by the multicast receivers. Join Table packet are forwarded upstream towards the multicast source to reinforce the forwarding path for multicast data.

III. **Forwarding nodes**

   These are intermediate nodes, which forward multicast data on behalf of the multicast source. Intermediate nodes indicate their willingness to participate in multicast data transmission by forwarding Join Query packets, which enable downstream nodes to select such nodes as forwarding nodes.

   By forwarding subsequent Join Table packets intermediate nodes elect themselves as forwarding nodes and thus form the forwarding path.

Forwarding state is maintained by periodic refreshing of routing paths by Join Query packets, which enable multicast receivers to discover better forwarding paths and recover from faulty forwarding paths.

The ODMRP consists of two phases, namely request and reply phase. To construct a multicast forwarding tree, an ODMRP source periodically floods join query packets to the entire network to advertise the availability of multicast. Upon receiving a non-duplicate join query packet, intermediate nodes record the sender as the upstream parent and a unique identifier of the packet (i.e., sequence number). Then intermediate nodes rebroadcast the packet. Duplicate join query packets are detected via unique identifier previously observed and suppressed for forwarding.

When the join query reaches a prospective receiver, the receiver selects the best path based on predefined criteria (i.e., least hop path, least delay path) and sends a join table packet back to the source. The join table packet is relayed by intermediate nodes and travels all the way back to the source on the reverse path. The join table packet reinforces the path established by the join query.

Subsequently, when the source sends data, the intermediate nodes become forwarding nodes in the data delivery tree. ODMRP maintains group membership as a soft-state in which parent-child relationships should be refreshed periodically. Hence, ODMRP does not require explicit join or leave mechanisms. However, periodic flooding of join query has a tradeoff. While frequent refreshing can improve the route recovery/discovery and thus result in an increased packet delivery ratio, it can overwhelm the nodes with excessive traffic overhead and waste network bandwidth and node resources. On the other hand, delayed path refreshing can actually delay route recovery/discovery process and can thus contribute to reduce the packet delivery ratio.

2.3.3. Protocol illustration

Following figures illustrate ODMRP forwarding path setup and multicast data forwarding. Figure 2.4 [21] illustrates join query packet forwarding, while Figure 2.5 [21] illustrates forwarding path setup by join table packets. Finally, Figure 2.6 [21] shows multicast data forwarding through previously established forwarding path.

In Figure 2.4 Node A broadcasts a Join Query packet as a multicast source, and all the surrounding nodes (B, C, D, E, and F) rebroadcast the received Join Query packet.

In addition, intermediate nodes retain a unique identifier of the last Join Query and the information of their immediate parent towards the multicast source. In the Figure 2.4, these packets open two possible forwarding paths from the source to multicast receiver L. L selects the path from J, and replies a Join Table packet back to the source reinforcing the selected path. While the Join Table packets travel toward the source, nodes K, J and C update their routing tables as forwarding nodes for multicast sender A (Figure 2.5). Once the Join Table packets reach the multicast sender A, it initiates sending of multicast data through the forwarding path (Figure 2.6).

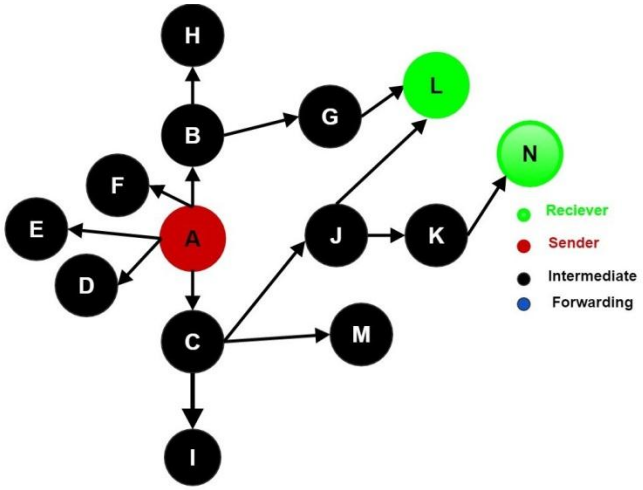

***Figure 2.4: ODMRP** Join Query*

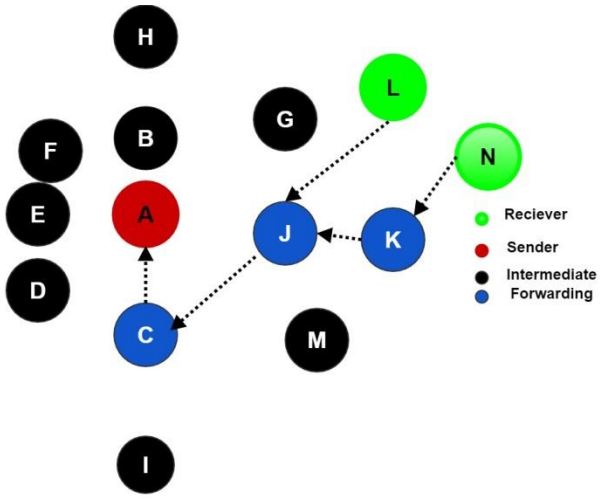

***Figure 2.5: ODMRP** Join Table*

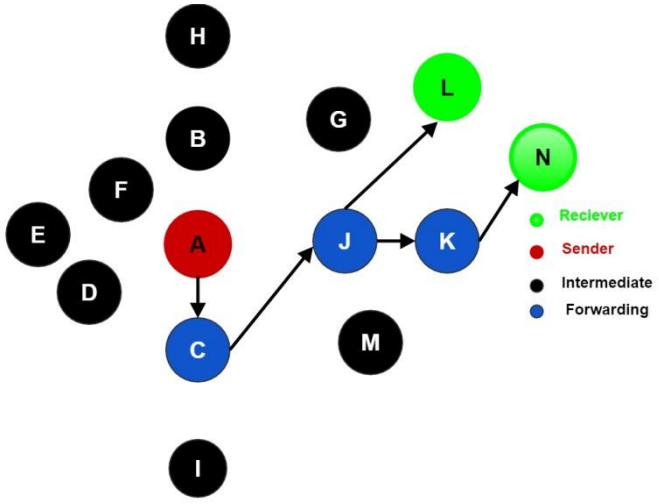

Figure 2.6: ODMRP data forwarding

Table 2.1: ODMRP Join Query packet format

Type	Reserved	Time to Live	Hop Count
Multicast Group IP address			
Sequence number			
Source IP address			
Previous Hop IP address			
Minimum Link Expiration Time			

Table 2.2: ODMRP Join Table packet format

Type	Reserved	Time to Live	Hop Count
Multicast Group IP address			
Sequence number			
Source IP address			
Next Hop IP address			

2.3.4. Protocol data structures

Following section documents different data structures used in ODMRP [21].

Protocol packet format

ODMRP uses following packet formats for join query and join table packets.

I. **join query packet** (table 2.1) consists of following fields,
   1. Type : ODMRP packet type (join query)
   2. Reserved : Not set
   3. Time to Live : Number of maximum hops to travel
   4. Hop Count : Number of hops traveled so far
   5. Multicast Group IP Address : IP address of the multicast group
   6. Sequence Number : Unique identifier assigned for the packet by the multicast source
   7. Source IP Address : IP address of the multicast source
   8. Previous Hop IP Address : IP address of the previous hop
   9. Minimum Link Expiration Time : Link expiration time assigned by multicast source

II. **join table packet** (table 2.2) consists of following fields,
   1. Type : ODMRP packet type (join table)
   2. Reserved : Not set
   3. Time to Live : Number of maximum hops to travel
   4. Hop Count : Hop count observed by the multicast sender
   5. Multicast Group IP Address : IP address of the multicast group
   6. Sequence Number : Unique identifier assigned by the multicast source for previous Join Query packet
   7. Source IP Address : IP address of the join table sender
   8. Next Hop IP Address : IP address of the next hop parent towards multicast source

Table 2.3: ODMRP routing table

Multicast Group IP address
Source IP address
Last Sequence number
Next Hop Parent
Hop count
Minimum Link Expiration Time
Forwarding flag

Routing table format

Each node maintains following routing table entries for each multicast source (table 2.3).

1. Multicast Group IP Address : IP address of the multicast group
2. Source IP Address : IP address of the multicast source
3. Last Sequence Number : Last observed unique identifier sent by the multicast source
4. Next Hop Parent : IP address of the next hop parent towards the multicast source
5. Hop Count : Number of hops towards parent through previous hop parent
6. Minimum Link Expiration Time : Link expiration time set by previous join query packet
7. Forwarding flag: Packet forwarding flag set for the multicast source based on join table packets.

2.3.5. Advantages of ODMRP

1. Simplicity

When compared with other wireless multicasting protocols ODMRP shows remarkable simplicity, which can be attributed to its simple control messaging format.

2. High delivery ratios

Despite its simplicity ODMRP shows high packet delivery ratios.

3. Low channel and storage overhead

In ODMRP, both forwarding and non-forwarding nodes maintain similar routing tables. In fact, the only difference between forwarding and non-forwarding nodes is the forwarding flag specifier in their routing table.

4. Robustness to host mobility

The multicast source can easily accommodate the mobility of the nodes by frequent route refreshing.

5. Periodic dynamic construction of the forwarding path

In addition, frequent route refreshing, enables the multicast subscribers to recover from faulty forwarding paths and to discover alternative better forwarding paths soon.

6. Maintenance and exploitation of multiple forwarding paths

Mesh based forwarding path setup enables the multicast subscribers to exploit the availability of multiple forwarding paths. In addition, makes it possible for the multicast receivers to recover paths and discover alternative forwarding paths.

7. Non dependence on a specific routing protocol

Any routing protocol with broadcasting capabilities can easily adopt ODMRP.

8. Unicast routing capability

Unlike most other multicast routing protocols, ODMRP does not require specific multicasting protocols for its implementation. Any networking protocol with broadcasting and unicasting can easily implement ODMRP.

9. Low control messaging overhead

Due to its mesh based forwarding node setup, nodes do not have to maintain forwarding trees. This has enabled the protocol to reduce its control messaging overhead. Route discovery and recovery are entirely done by Join Query packets. In addition, Join Query packets for route refreshing can be piggy backed with data packets.

10. Soft state based multicast subscription and un-subscription.

Forwarding path setup is entirely based on received Join Table packets by intermediate nodes. This collaborative setup works well to reduce the control packet overhead. To stop a multicast session, the sender just stops sending Join Query packets, similarly to unsubscribe from an ongoing multicast session, a receiver just stops sending Join Table upstream. In addition, forwarding nodes are demoted to non-forwarding nodes if they do not send Join Table packets upstream.

2.3.6. Drawbacks of ODMRP

1. Waste network bandwidth

ODMRP Join Query packets are flooded throughout the network. Each node receiving a Join Query packet is expected to rebroadcast it. The rationale behind this design was that it would improve the forwarding path setup. But on the other hand, redundant Join Query packets not only waste network bandwidth and channel capacity but also increase packet collisions and drain resources of both senders and receivers. In addition, this redundant messaging overhead significantly affects the efficiency of the protocol in real world environments with a high density of nodes.

2. Dependence on a single forwarding path for data forwarding

Once a forwarding path in established through a Join Query/Join Table cycle, the multicast receivers rely on the established forwarding path for the entire route refresh duration. Although, ODMRP uses mesh based networking for forwarding path setup, it does not allow the multicast receivers to utilize additional forwarding paths. This design flaw essentially converts the mesh based forwarding structure to a tree based one for the entire duration of the refresh cycle.

3. Adversaries can impact forwarding behavior

An adversary or failed node on established forwarding path can easily deny all nodes in its downstream path from receiving data packets.

4. Multicast sender has to periodically refresh forwarding path

In order to maintain the forwarding tree, the multicast sender has to frequently refresh forwarding path by sending Join Query packets. Although this enables multicast receivers to discover better forwarding paths and recover from faulty forwarding paths it is not desirable for less faulty networks. In fact, this increases the protocol overhead and directly impacts the efficiency of the protocol. In addition, frequent Join Query broadcasts drain resources of the nodes and would discourage other nodes from participating in packet forwarding.

5. Does not consider mobility of the nodes

The protocol does not consider the mobility of the forwarding/receiving nodes. The design of the protocol assumes that frequent refresh cycles can accommodate node mobility. But as discussed above, frequent forwarding path refresh cycles reduce the efficiency of the protocol.

2.4. Proposals to improve the performance of ODMRP for MANETS

Various proposals have been made in the literature to improve the performance of on-demand multicast routing protocols in Mobile ad-hoc networks. These proposals are broadly classified into two groups [8] [10] [11] based on the drawbacks described in section 2.3.6:

1. Delivery structure proposals
2. Flooding proposals

2.4.1. Delivery structure proposals

The delivery structure proposals mainly deal with the way how delivery structure be optimized for multicasting from sender to multicast group receivers so that a packet send from sender reaches multicast group receivers properly. This proposal is motivated by the fact that creating stable and efficient delivery structure should increases the performance of the routing protocol such that the protocol performance is affected by packet losses which are due to the quality of links and performance of nodes participating in mesh creation phase.

From delivery structure proposals efficient forwarding are presented.

Efficient forwarding: This technique [8] [10] [11] enhances performance of routing protocol by creating a small and strong delivery structure which consists of mobile nodes with high power, low mobility speed, and high available bandwidth assuming that nodes with the above characteristics has impact on performance of the routing protocol which should lead to decreased resources consumption and higher stability of the delivery structure.

Having larger forwarding group increases resilience against node and link failure which also increases overhead. A node is classified as strong when it has certain desirable properties which increase the probability of data delivery when this node should participate in the forwarding group. Examples of such properties are high bandwidth availability, low loss rate, low moving speed, or high power level. If a node is not strong it should be classified as a weak node and it might be not optimal to include the node in the forwarding process. For example, if a node has a low power level, the probability that the mode powers down in the near future should be high which leading to packet loss due to node failures and re-routing.

A strong forwarding group is made out of a set of strong nodes in the network. A weak forwarding group is made out of weak nodes in network. But in many cases, it is impossible to establish forwarding group in the network only by using strong nodes. For example, some nodes might have weak links due to mobility and link quality might change over time. Also, it might be possible that no strong group should exist. A strong forwarding group should be smaller than a weak forwarding group, because it removes weak nodes from forwarding group networks.

In this method, better routes (e.g. composed of nodes and links having high bandwidth, high power level and low mobility) have a high probability to be saved and forwarded but weak routes (having low bandwidth, low power level, high mobility) have a lower probability. As a result, we can establish a small and strong forwarding group instead of a larger and weaker one. A larger forwarding group suffers from high data and control overhead and consumes more power, bandwidth and wireless resources. Having weak nodes participating in the forwarding group increases the probability of packet loss and re-routing.

2.4.2. Flooding Proposals

Since flooding affects performance of routing protocols by sending of join query packet to the entire network as shown in section 2.3.6, so it needs a mechanism which restricts the domain of join query packet flooding to the entire network.

From flooding proposals use of previous route experience and restricting the domain of control packet flooding are presented.

Use of previous route experience: Use of previous route experience [22] [23] method minimizes the repetitive broadcasting of join query packets. Assuming that the source broadcasts join query packet to the entire network for more than one time for the same multicast receivers and if the source address and the sequence number of the control packet matches, the forwarding group member next to the source should rebroadcasts back to the source immediately by resolving from CBR whether it is forwarding group to the source without the intervention of other intermediate forwarding nodes. Since the routing table is built based on CBR method, it maps the solution for same control packets similar to the past event. This decreases the repetitive broadcasting of join query packet for the same multicast receiving nodes which in turn decreases the packet overhead as the number of nodes increases.

Restricting the domain of control packet flooding: These [8] [10] [11] methods restrict the domain of control packet flooding.

If the mobility of nodes is not too high, the topology should not have changed much between the last update of the mesh structure. So, it is possible to restrict the domain of the join query packet flooding to the entire network.

For each multicast group the node is participating in, the source ID and the time when the last join query is received from the source is recorded. If no join query is received from a source within the refresh period that entry is removed from the Member Table.

If mobility of nodes is low, intersection of new forwarding groups and previous forwarding group should be high. In such scenarios it is beneficial to limit the propagation domain of join query packets to an area restricted to a few hops away of the members of the previous forwarding group. However, the current forwarding group of a network with high mobility has lower intersection with the previous forwarding group.

In this method, the new forwarding group can be established from the current forwarding group. Join query packets which are going to be sent to a region far away from the previous forwarding group should be dropped. To implement this idea, we added a field to the join query packet (Number of previous forwarding group), which counts the number of the previous forwarding group nodes which was visited by this join query packet. If a join query packet has visited many nodes but it doesn't see any previous FG nodes, we can assume that this join query packet should not be useful so it should be discarded. Therefore, when a node receives a new join query packet, it extracts NOPFG (Number of previous forwarding group) and Hop count fields from the incoming packet. Hop count field is the number of hops to this node from the sender. When a node receives a join query packet with a hop count greater than the minimum value, it decides if the join query packet should be forwarded or discarded based on a random value. This random value is based on the forwarding probability. The minimum value of hop count allows a join query packet to traverse sufficient number of hops to prevent from discarding all copies of join query packets.

2.5.Comparison of the previously done proposals

In general three proposals have been presented. These proposals improve the performance of on-demand multicast routing protocol. The two flooding proposals and Delivery structure proposal efficient forwarding are based on the join request packets which are flooded to the entire network to establish or update the multicast delivery structure of nodes to forward multicast data from source nodes to destination. When we compare the flooding proposals, Use of previous route experience emerges as a good proposal, because it doesn't fail to find route from source to destination when the network density is high.

In general efficient forwarding together with restricting the domain of control packet flooding should work well. And it improves increases packet delivery ratio by up to 40%, reduces average end to end delay by about 35%. However, the assumption is that when the density of network increases these proposals fail to find route to the destination.

CHAPTER THREE: CASE-BASED FUZZY-LOGIC ODMRP (CFODMRP)

3.1. Overview

Case-based and fuzzy-logic based algorithm is an integrated method to improve the performance of multicast routing protocols in mobile ad-hoc networks. It mainly deals with the way how the mesh delivery structure was created and how routes between source and receivers pair of a given multicast group is updated and maintained. The multicast routing protocols designed to wired networks does not perform well on ad-hoc environment where the mobile nodes are free to move and the nodes are equipped with limited resources. As a result, the performance of such multicast routing protocols degrades. The case-based fuzzy-logic methods are the integration of fuzzy logic and case based methods which are aimed to provide some possible solutions by reducing the overhead the protocols creates due to mesh delivery structure and flooding of control packets to the entire network.

The case-based fuzzy-logic methods are aimed to creating a routing path between source and receivers pair having nodes which are more stable and efficient than other nodes and on creating routing path from source to destination every node stores routing information so that it should be re-used later.

The fuzzy logic methods are used to establish routing path between multicast source and receivers containing nodes which are with higher power levels, less moving speed, and links having high bandwidth.

The case based methods are used to restrict the domain of control packet flooding to the entire network to establish and refresh route path between multicast source and receivers. The case based methods are applied to forwarding group nodes which are created by the fuzzy logic methods earlier.

Once forwarding group is created between multicast source and receivers is created, the source refresh the route between multicast source and receivers by using previous route information stored in each forwarding nodes using case based methods.

3.2. The Proposed Algorithm

In this section we present CFODMRP routing protocol algorithm in detail, with implementation using NS-2 of version 2.35.

3.2.1. Fuzzy Logic Based Approach

In mobile ad-hoc networks the assumption made is that all nodes in the networks are not equally participated due to node and link characteristics such that some nodes are strong nodes than others and some links have higher bandwidth availability than others.

In order to establish a forwarding group that is stable and more efficient several fields (found at last column of table 3.1) are added in the join query packet described in section 2.3.4 to allow better route selection in the route request process as shown in table 3.1 [10]. Based on such information, the next nodes should be able to compute the probability of caching and forwarding the received join query packets to next node.

In MANETs, nodes are classified as strong and weak nodes. The strong node has properties like high power level, high bandwidth availability, and low moving speed. A strong forwarding group is formed by using strong nodes. The probability of data delivery is increased by using strong forwarding group in the path. These strong forwarding groups are formed by using fuzzy logic based approach which should lead to decreased resources consumption and higher stability of the delivery structure.

Table 3.1: modified join query packet

Type	Reserved	Time to Live	Hop Count
Multicast Group IP address			
Sequence number			
Source IP address			
Previous Hop IP address			
Minimum Link Expiration Time			
Bandwidth	*Loss*	*Speed*	*Power*

Once a node receives a join query packet, it needs to process the additional parameters added to the join query packets like bandwidth, speed, power, and loss rate of the previous node. To process the above parameters, node needs to use fuzzy logic to handle network dynamics, imprecise information and uncertainty. As a result, the node computes a probability of caching and forwarding of the join query. If a join query packet has arrived through a "strong" node, the join query packet is cached and forwarded with a high probability. A simple membership function is used to Fuzzify parameters.

The value of node parameters are shown in horizontal axis and membership probability is shown in vertical axis. By using the parameters value node have to classify them as low, medium and high probability nodes. Before forwarding node replaces its own parameters information in join query packet.

Figure 3.1 [10] show each node's decision process based on the fuzzy logic. Input to this process is the previous node's operating parameters (such as bandwidth, speed, power and loss rate) where the probability of catching and forwarding is the output of the process.

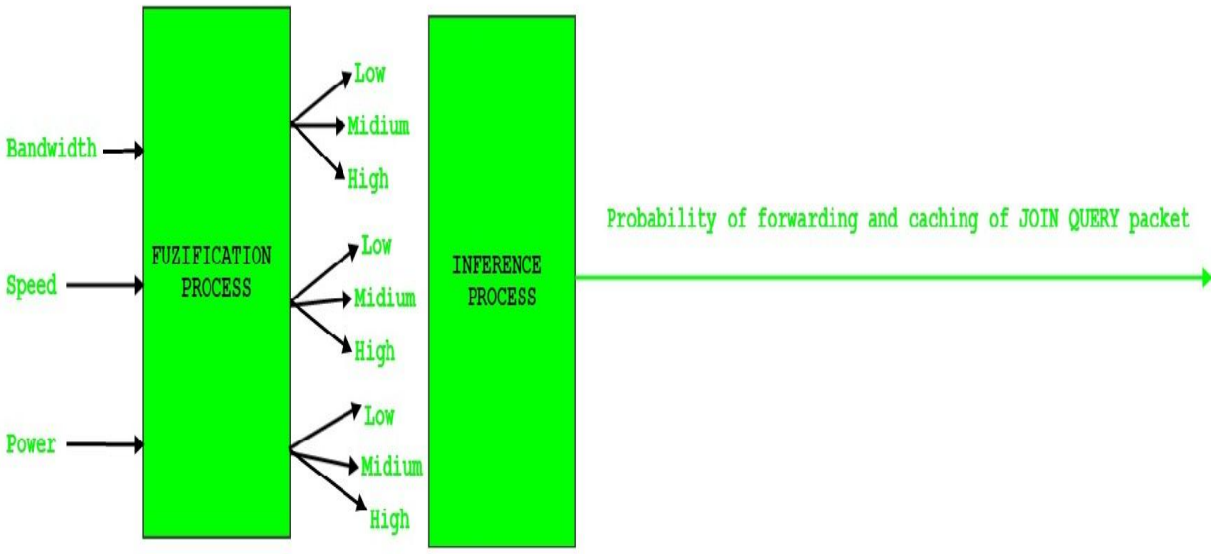

Figure 3.1: Fuzzification process

To decide about saving or discarding a received join query packet, we use only inside information of packets and some updated average values for loss, speed, bandwidth and power. These average values would be more accurate with receiving number join query packets

In the inference stage of the fuzzy process, inference laws shown in Algorithm 1 are used to compute the probability of caching and forwarding based on the simple rules.

If ((bandwidth is high) **and** (power is high) **and** (speed is low) **and** (loss rate is low))

then increase probability of forwarding join query packet

If ((bandwidth is low) **and** (power is low) **and** (speed is high) **and** (loss rate is high))

then decrease probability of forwarding join query packet

If ((bandwidth is low) **and** (power is high) **and** (speed is medium) **and** (loss rate is medium))

then do not change probability of forwarding join query packet

Algorithm 1: Inference Rules.

3.2.2. Case Based Approach

In On-demand multicast routing protocol for Mobile ad-hoc networks the assumption made is that whenever the source has data to send it broadcasts the join query packets to the entire network which creates an overhead.

Case based method are used to reduce such overhead so that the route refresh and update is based on previous route information stored in each node of multicast group members assuming that the source broadcasts join query packet to the entire network for more than one time for the same multicast receivers.

Based on this method a routing table is built on each node using case based reasoning (CBR) method and it should carry a multicast data and retains resulting experience for reuse as shown in Figure 3.2 below [22].

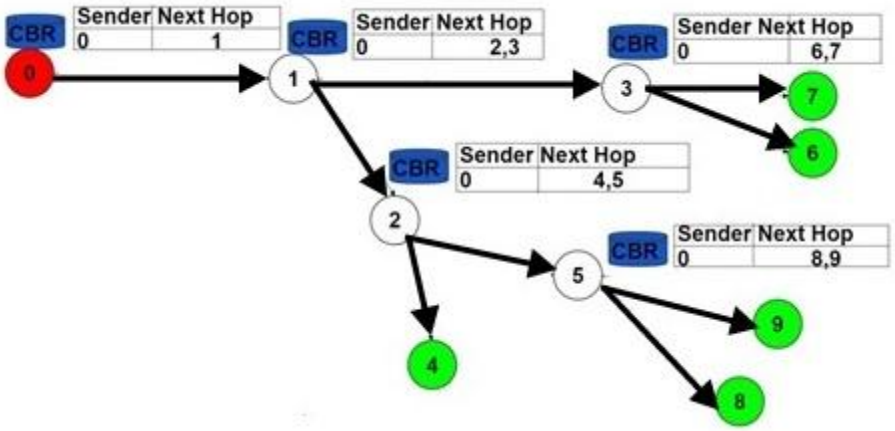

Figure 3.2: routing table

If the source address and the sequence number of the join query packet matches, the forwarding group member next to the source should rebroadcasts back to the source immediately as shown in Figure 3.3 [22]. This decreases the repetitive broadcasting of join query packet for the same multicast receivers which in turn decreases the packet overhead.

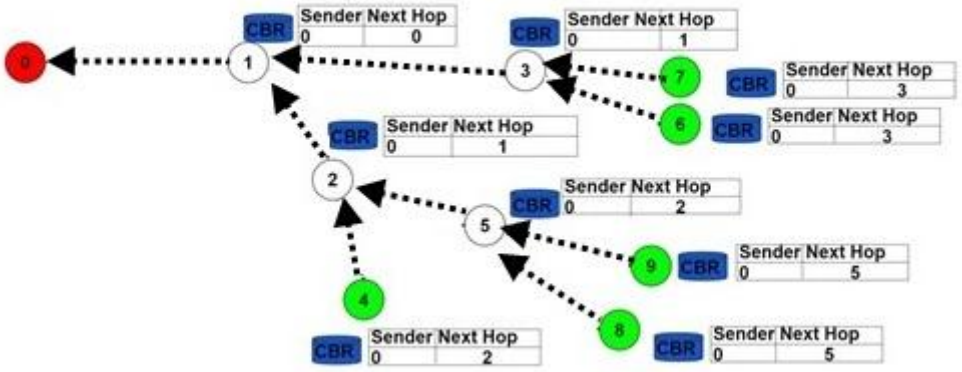

Figure 3.3: join table

When the number of an intermediate forwarding nodes increases due an increase in number of nodes the source join query packet should broadcasted to all members and the join table reply should be broadcasted back to multicast source in reverse direction during which CBR records and updates the previous join table propagation to multicast source and forwarding group membership information along with multicast table formed.

3.2.3. Case-Based Fuzzy-Logic ODMRP (CFODMRP)

The combination of the two methods described above is applied on ODMRP is to reduce the overhead and improves scalability.

The following algorithm 2 shows case based fuzzy logic method improved ODMRP.

```
CFODMRP_handle_function (ROUTE_REQ_packet rr_packet)
{
Get source_address and sequence_number from rr_ packet fields;
If jq_packet was matched then reply_to_source;
Get parameters (loss rate, bandwidth, speed, power)
from rr_packet fields;
Fuzzify parameters;
Compute probability of Route-REQ forwarding
based on fuzzification results;
Replace parameters of this node to rr_packet fields;
Forward rr_packet based on the computed probability;
If rr_packet was forwarded then cache rr_packet;//cache in message cache and CBR
}
```

Algorithm 2: Case-Based Fuzzy-Logic Improved ODMRP

In the above algorithm jq_packet is reply_to_source when it gets matching entries from CBR stored in each node.

CHAPTER FOUR: IMPLEMENTATION

Designing an efficient network plays an important role in this world and then it is even essential part to check the performance of the designed network, which will be a difficult task in a real time application. For this many network simulators have been designed so far among the most reputed one is NS2 (Network Simulator), NS2 is an open source simulating tool it is combination of C++ & Otcl with less document support specially used by developers [24].

4.1. Why NS-2

I. It is an open source simulator which can be downloaded freely
I. It is used for the simulation of network protocols with different network topologies.
III. It is capable of simulating wired as well as wireless networks.
IV. Very flexible and easy graphical interface to view the results.

4.2. Introduction to NS-2

Network Simulator (NS-2) [25] is a discrete event simulator for networking research. It is an ongoing open source project, which is being developed with the contributions of various network researchers. The open nature of the project enables researchers to easily access and modify the implementation of protocols. In addition, the open source development model encourages the researchers to develop on previous work done in their research area. Following object-oriented concepts, the low level simulation engine and protocols are developed in C++, while tcl/tk based high level scripting layer makes it possible to change simulation scenarios dynamically. Although NS-2 has in depth support for wired protocols such as TCP, it does not have built in support for common wireless protocols.

The Carnegie Mellon University's Monarch Research Group [26] has developed several wireless protocols, including ODMRP for NS-2 platform. Implementation of CFODMRP discussed below is based on this reference implementation of ODMRP.

As discussed above NS-2 consists of two layers, C++ lower layer implements the core protocols while OTcl upper layer provides an interpreter for Tcl bindings, which makes it possible to change simulation parameters without C++ code changes and recompilation of the simulator.

Although this two layer model makes the simulator highly efficient and flexible, it also makes protocol development much more challenging. The protocol developers not only have to implement the protocols in C++, but also have to implement OTcl interfaces for tcl scripting.
It is focused on modeling network protocols including wired, wireless and satellite networks with transport protocols such as TCP, and UDP with both unicasting and multicasting capabilities. It models Web, Telnet, and FTP applications. It also includes the implementation of ad-hoc routing and sensor networks. It provides mechanisms for gathering statistics, tracing, and error modeling for the simulations carried out. Apart from the core code of the NS-2, there have been numerous contributions from other researchers.

We have used NS-2 to perform the network simulations. The simulation results were used to evaluate the performance of routing protocol CFODMRP and are used to compare the result with ODMRP. We chose NS-2 for the implementation because it is a freely distributed code and supports many interesting protocols.

C++ and Tcl are the two languages used in NS-2. Two languages are needed to perform complex programming, coupled with the need for speed when we vary Parameters/configurations to explore a large number of scenarios while studying the various protocols. C++ is fast to execute, however it is slow to change, making it suitable for the complicated protocol implementation. However, it is very slow when varying parameters and rerunning simulations. Tcl, on the other hand, is much slower but very convenient for varying simulation parameters. Consequently, C++ is used for implementing properties of the protocol while Tcl is used to implement code that needs to be changed often in order to study the protocol behavior.

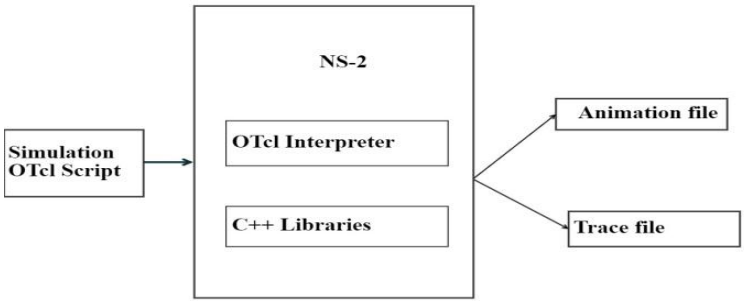

Figure 4.1: Structure of NS-2

As shown in Figure 4.1, an OTcl script written by a user is interpreted by NS. While OTcl script is being interpreted, NS creates two main analysis reports simultaneously [25]. One of them is NAM (Network Animator) object that shows the visual animation of the simulation. The other is the trace object that consists of the behavior of all objects in the simulation. Both of them are created as a file by NS. The former is ".nam" file used by NAM software that comes along with NS. The latter is a ".tr" file that includes all simulation traces in the text format.

4.3. Wireless networking in NS-2

4.3.1. Mobile Node

Wireless networking in NS-2 is defined by having a mobile node. A *Mobile Node* thus is the basic *Node* object with added functionalities of a wireless networks. Mobile nodes have the following functionalities:

I. Ability to move within a given topology.
II. Ability to receive and transmit signals to and from a wireless channel etc.

A major difference between a node of wired network and mobile node is that a Mobile Node is not connected by means of Links to other nodes or mobile nodes. Moreover, routing in mobile networks especially in Ad-hoc networks is distributed and there is no centralized entity (as router in wired network). Therefore a mobile node acts as a router and as a node at the same time.

4.3.2. Node movement

The mobile node is designed to move in a three dimensional topology. However the third dimension (Z) is not used. That is, the mobile node is assumed to move always on a flat terrain with Z always equal to 0. Thus, the mobile node has X, Y, Z (=0) co-ordinates that is continually adjusted as the node moves.

There are two mechanisms to induce the movement in mobile nodes.

In the first method, starting position of the node and its future destinations may be set explicitly. These directives are normally included in a separate movement scenario file. The starting position and future destinations for a mobile node may be set by using the following APIs:

$node set X_ \<x1\>

$node set Y_ \<y1\>

$node set Z_ \<z1\>

$ns at $time $node setdest \<x2\> \<y2\> \<speed\>

At $time sec, the node would start moving from its initial position of (x1,y1) towards a destination (x2,y2) at the defined speed.

In this method the node-movement-updates are triggered whenever the position of the node at a given time is required to be known. This may be triggered by a query from a neighboring node seeking to know the distance between them, or the setdest directive described above that changes the direction and speed of the node.

An example of a movement scenario file using the above APIs, can be found in tcl/mobility/scene/scen-500x500-50-50-20-0. Here, 500x500 defines the length and width of the topology with 50 nodes moving at a maximum speed of 20m/s with an average pause time of 50s. These node movement files may be generated using CMU's scenario generator which is found under indep-utils/cmu-scen-gen/setdest.

The second method employs random movement of the node. The primitive to be used is:

$mobilenode start

which starts the mobile node with a random position and have routined updates to change the direction and speed of the node. The destination and speed values are generated in a random fashion. The mobile node movement is implemented in C++.

Irrespective of the methods used to generate the node movement, the topography for mobile nodes needs to be defined. It should be defined before creating the mobile nodes. Normally, flat topology is created by specifying the length and width of the topography using the following primitive:

set topo [new Topography]

$topo load_flatgrid $opt(x) $opt(y)

where opt(x) and opt(y) are the boundaries used in simulation.

The movement of the node is determined by setting an area and mobility scenarios.

4.3.3. Network Components in a mobile node

The network stack for a mobile node consists of a link layer (LL), an ARP module connected to LL, an interface priority queue (IFq), a MAC layer (MAC), a network interface (netIF), all connected to the channel. These network components are created and plumbed together in OTcl.

Each component is discussed briefly as follows:

Link Layer

The LL object is responsible for simulating the data link protocols like in wired networks. The only difference being the link layer for mobile node has an ARP module connected to it which resolves all IP to hardware (Mac) address conversions. Normally for all outgoing (into the channel) packets, the packets are handed down to the LL by the Routing Agent. The LL hands down packets to the interface queue. For all incoming packets (out of the channel), the mac layer hands up packets to the LL which is then handed off at the node_entry_point [25].

ARP

The Address Resolution Protocol module receives queries from Link layer. If ARP has the hardware address for destination, it writes it into the Mac header of the packet. Otherwise it broadcasts an ARP query, and caches the packet temporarily. For each unknown destination hardware address, there is a buffer for a single packet. In case, additional packets to the same destination are sent to ARP, the earlier buffered packet is dropped. Once the hardware address of a packet's next hop is known, the packet is inserted into the interface queue [25].

Interface Queue

The PriQueue../ns-2/priqueue.h [25] is implemented as a priority queue which gives priority to routing protocol packets, inserting them at the head of the queue. It supports running a filter over all packets in the queue and removes those with a specified destination address.

Mac Layer

The IEEE 802.11 distributed coordination function (DCF) Mac protocol has been implemented by CMU. It uses a RTS/CTS/DATA/ACK pattern for all unicast packets and simply sends out DATA for all broadcast packets. The implementation uses both physical and virtual carrier sense [25].

Tap Agents

Agents that subclass themselves as Tap../ns-2/mac.h [25] defined in mac.h can register themselves with the mac object using method installTap(). If the particular Mac protocol permits it, the tap should promiscuously be given to all packets received by the MAC layer, before address filtering is done [25].

Network Interfaces

The Network Interface layer serves as a hardware interface which is used by mobile node to access the channel. The wireless shared media interface is implemented as Phy/WirelessPhy../ns-2/wireless-phy.h [25]. This interface subject to collisions and the radio propagation model receives packets transmitted by other node interfaces to the channel. The interface, stamps each transmitted packet with the meta-data related to the transmitting interface like the transmission power,wavelength etc. This meta-data in packet header is used by the propagation model in receiving network interface to determine if the packet has minimum power to be received and/or captured and/or detected (carrier sense) by the receiving node. The model approximates the DSSS radio interface (Lucent WaveLan direct-sequence spread-spectrum) [25].

Radio Propagation Model

It uses Friss-space attenuation () at near distances and an approximation to Two ray Ground () at far distances. The approximation assumes specular reflection off a flat ground plane. See tworayground.{cc,h}in [25].

Antenna

An omni-directional antenna having unity gain is used by mobile nodes. See antenna.{cc,h} for implementation details in[25].

4.4. NS-2 Trace Support

The NS-2 trace files are used for post processing the simulation that is carried on. The following are the supported formats of trace files:

- Trace files for wired networks that is wired
- Satellite
- Wireless (old and new trace)
- Wired-cum-wireless

In the simulation, this research has used the new trace format for wireless networks. And the following section describes the new trace format. [25].

s-t 0.267662078 -Hs 0 -Hd -1 -Ni 0 -Nx 5.00 -Ny 2.00 -Nz 0.00 –Ne

-1.000000 -Nl RTR -Nw --- -Ma 0 -Md 0 -Ms 0 -Mt 0 -Is 0.255 -Id -1.255 –It message -Il 32 -If 0 -Ii 0 -Iv 32

s-t 1.511681090 -Hs 1 -Hd -1 -Ni 1 -Nx 390.00 -Ny 385.00 -Nz 0.00

–Ne -1.000000 -Nl RTR -Nw --- -Ma 0 -Md 0 -Ms 0 -Mt0 -Is 1.255 -Id

-1.255 –It message -Il 32 -If 0 -Ii 1 -Iv 32

s-t 10.000000000 -Hs 0 -Hd -2 -Ni 0 -Nx 5.00 -Ny 2.00 -Nz 0.00 –Ne

-1.000000 -Nl AGT -Nw --- -Ma 0 -Md 0 -Ms 0 -Mt 0 -Is 0.0 -Id 1.0 -It tcp -Il 1000 –If 2 -Ii 2 -Iv 32 -Pn tcp -Ps 0 -Pa 0 -Pf 0 -Po 0

r-t 10.000000000 -Hs 0 -Hd -2 -Ni 0 -Nx 5.00 -Ny 2.00 -Nz 0.00 -Ne

The new trace format as seen above can be divided into the following fields:

Event type

In the traces above, the first field (as in the older trace format) describes the type of event taking place at the node and can be one of the four types:

S Send

R Receive

D Drop

F Forward

General tag

The second field starting with "-t" may stand for time or global setting

-t Time

-t Global setting

Node property tags

This field denotes the node properties like node-id, the level at which tracing is being done like agent, router or MAC. The tags start with a leading "-N" and are listed as below:

-Ni Node id

-Nx Node's x-coordinate

-Ny Node's y-coordinate

-Nz Node's z-coordinate

-Ne Node's energy level

-Nl Trace level, such as AGT, RTR, MAC

-Nw Reason for event

The different reasons for dropping a packet are given below:

END DROP_END_OF_SIMULATION

COL DROP_MAC_COLLISION

DUP DROP_MAC_DUPLICATE

ERR DROP_MAC_PACKET_ERROR

RET DROP_MAC_RTERY_COUNT_EXCEDDED

STA DROP_MAC_INVALID_STATE

BSY DROP_MAC_BUSY

NRTE DROP_RTR_NO_ROUTE i.e no route is available

LOOP DROP_RTR_ROUTE_LOOP i.e there is a routing loop

TTL DROP_RTR_TTL i.e TTL has reached zero

TOUT DROP_RTR_QTIMEOUT i.e packet has expired

CBK DROP_RTR_MAC_CALLBACK

IFQ DROP_IFQ_QFULL i.e no buffer space in IFQ

ARP DROP_IFQ_ARQ_FULL i.e dropped by ARP

OUT DROP_OUTSIDE_SUBNET i.e dropped by base stations on receiving routing updates from nodes outside its domain

Packet information at IP level

The tags for this field start with a leading "-I" and are listed along with their explanations as following:

-Is Source address.source port number

-Id Dest address.dest port number

-It Packet type

-Il Packet size

-If Flow id

-Ii Unique id

-Iv TTL value

Next hop info

This field provides next hop info and the tag starts with a leading "-H".

-Hs Id for this node

-Hd Id for next hop toward the destination

Packet info at MAC level

This field gives MAC layer information and starts with a leading "-M" as shown below:

-Ma Duration

-Md Destination Ethernet address

-Ms Source Ethernet address

-Mt Ethernet type

Packet info at "Application level"

The packet information at application level consists of the type of application like ARP, TCP, the type of ad-hoc routing protocol like DSDV, DSR, and AODV etc. are being traced. This field consists of a leading "-P" and list of tags for different applications are listed as below:

-Parp Address Resolution Protocol

-Po ARP Request/Reply

-Pm Source MAC address

-Ps Source address

-Pa Destination MAC address

-Pd Destination address

-Pdsr DSR routing protocol (an example DSR and the following details for DSR)

-Pn How many nodes traversed?

-Pq Routing request flag

-Pi Route request sequence number

-Pp Route reply flag

-Pl Reply length

-Pe Source of source routing to destination

-Pw Error report flag

-Pm Number of errors

-Pc Report to whom

-Pb Link error form linka->linkb

-Pcbr Constant bit rate information and the following are details of CBR

-Pi Sequence number

-Pf How many times his packet was forwarded

-Pq Routing request flag

-Pi Route request sequence number

-Pp Route reply flag

-Pl Reply length

-Pe Source of source routing to destination

-Pw Error report flag

-Pm Number of errors

-Pc Report to whom

-Pb Link error form linka->linkb

-Pcbr Constant bit rate information and the following are details of CBR

-Pi Sequence number

-Pf How many times his packet was forwarded

4.5. NS-2 Network Animator (NAM)

Network Animator (NAM) is an animation tool for viewing network simulation traces and real world packet traces [25]. It supports topology layout, packet level simulation and various data inspection tools.

Before starting to use NAM, a trace file has to be created. This trace file is usually generated by NS-2 as discussed above. It contains topology information for example node and links as well as packet losses. During simulation, the user can produce topology configuration, layout information and packets traces using tracing events in NS-2.

Once the trace file is generated, NAM can be used to animate it. Upon starting, NAM should read the trace file, create the topology, pop up a window, do layout if necessary and pause at time 0. Through its user interface, NAM provides control over many aspects of animation. The following figure shows a screenshot of NAM window simulating wireless networks.

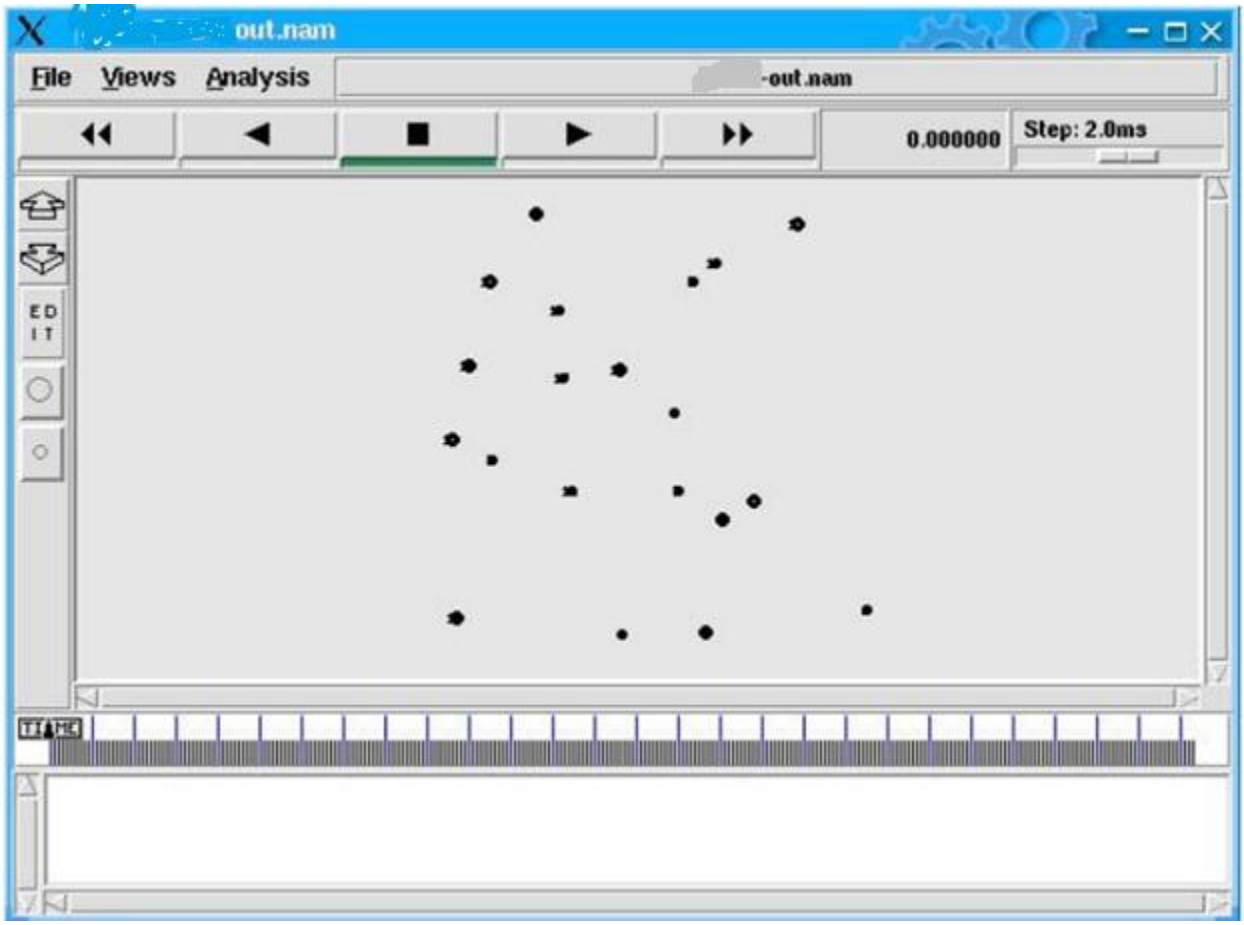

Figure 4.2: NS-2 Network Animator window

4.7. Implementation of ODMRP

Following files define the corresponding classes for the implementation of ODMRP.

Figure 4.3 illustrates a class diagram of the protocol implementation.

1. defs.h - Defines different protocol parameters
2. hdr_ odmrp.h/cc - Defines ODMRP packet header (hdr_odmrp class)
3. odmrpagent.h/cc - Defines ODMRP node (ODMRP Agent class)
4. msg_cache.h/cc - Defines packet cache (MSGCache class)
5. mem_table.h/cc - Defines ODMRP member table (McastMemberTable class)
6. packet_buffer.h/cc - Defines packet buffer (PacketBuffer class)
7. jq table.h/cc - Defines Join Query look up table (JQTable class)
8. jr table.h/cc - Defines Join Table look up table (JRTable class)
9. join query timer.h/cc - Defines Join query timer (JoinQueryTimer class)
10. join reply timer.h/cc - Defines Join Table timer (JoinReplyTimer class)

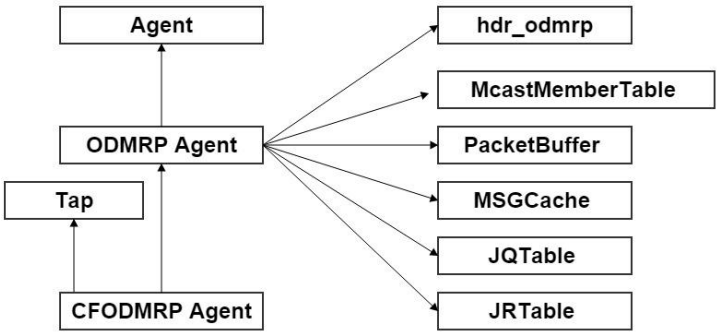

Figure 4.3: Class diagram

4.7.1. ODMRPAgent

In NS-2 simulator, endpoints where network packets are consumed or constructed are called Agents. Therefore ODMRPAgent class defines the network endpoints for ODMRP protocol. The ODMRPAgent inherits from generic Agent class and ODMRPAgent maintains its routing table as a set of hash tables (maps), which are defined in JQTable and JRTable classes. JoinQueryTimer and JoinReplyTimer classes define the route refresh timers and route expiration timers.

4.7.2. ODMRP packet header

ODMRP packet header is defined as a wrapper for both Join Query and Join Table packets. This design avoids addition of multiple packet types for the protocol implementation. In addition, it follows NS-2's programming convention, where each protocol is defined with a single packet header. ODMRP packet header is defined in **hdr_odmrp** class.

```
class hdr_odmrp {
    private:
        int valid_ack_;
        odmrpaddr_t mcastgroup_addr_;
        odmrpaddr_t prev_hop_addr_;
        struct odmrp_join_query join_query_;
        struct odmrp_join_reply join_reply_;
     public:
        int valid_;
        static int offset_;
}
```

4.7.3. Join Query packet

Join Query packet is implemented as an embedded packet, which could be embedded in ODMRP header. Join Query packet is defined as follows,

```
struct odmrp_join_query {
    int valid_;
    int seqno_;
    int hop_count_;
};
```

4.7.4. Join Table packet

Similarly Join Table packet is defined as a sub-message, which could be embedded in ODMRP header. Join Table is defined as,

```
struct odmrp_join_reply {
    int valid_;
    int seqno_;
    int count_;
    struct prev_hop_pairs pairs_[OD_MAX_NUM_PREV_HOP_PAIRS];
};
```

4.7.5. TCL hooks

In order to create instances of ODMRP, tcl hooks are created as follows. This enables creation of ODMRP nodes in tcl scripts.

```
static class ODMRPAgentClass : public TclClass {
    public:
        ODMRPAgentClass() : TclClass("Agent/ODMRPAgent") {}
        TclObject* create(int,  const char*const*) {
                return (new ODMRPAgent);
         }
} class_ODMRPAgent;
```

4.7.6. TCL commands

Agents implement command method that defines different tcl commands, which are supported by the particular agent. ODMRPAgents support following commands:

1. startodmrp - Initiates execution of ODMRP agent
2. reset - Resets execution of ODMRP agent
3. print-membership - Prints current agents membership table
4. mcast-src - Checks whether current node is a multicast source
5. mcast-sink - Checks whether current node is a multicast receiver
6. ip-addr - Returns agent's IP address
7. mac-addr - Returns agent's MAC address
8. join-group - Advices agent to subscribe to a multicast group
9. leave-group - Advices agent to unsubscribe from a multicast group
10. log-target - Creates a log trace

4.7.7. Timers

JoinQueryTimer and JoinReplyTimer classes implement the timers for route refreshing and routing table entry expiration.

4.7.8. Packet handling

ODMRPAgent defines recv, sendJoinQuery and sendJoinReply methods for packet handling. Following Pseudo-code illustrate the implementation of recv method for packet receipt.

Algorithm 3: Pseudo-code of recv() method for a received packet

1- First thing we should do is to check we are not receiving a packet we sent ourselves.

   I- If so If that is the case we should drop the packet and return
   II- If not

2- Retrieve the ttl value of a given packet
3- if (iph->ttl() <= 1) {

   I. drop(packet, DROP_RTR_TTL);
   II. packet = 0;
   III. return;

}

4- check the type of received packet while(iph->ttl() > 1)

I. if is new Join Query packet packet_cache.enterInCache() is called

II. if is type Data handleDataPacketSend()is called

III. if is type Join Table packet handlePacketReceipt()is called

4.7.9. Tracing ODMRPAgents

In order to trace simulation execution, NS-2 defines Trace objects, which log trace events (i.e., sending/receiving packets). CMUTrace extends this functionality to wireless protocol tracing. Following code snippet implements tracing functionality for ODMRP.

```
void CMUTrace::format_odmrp(Packet *p, int offset) {
    hdr_odmrp *mh = hdr_odmrp::access(p);
    sprintf(pt_->buffer() + offset, "-%d- %d [%d %d %d %d] [%d %d %d %d] (%d,
%d)",
   mh->mcastgroup_addr(),
   mh->prev_hop_addr(),
   mh->join_query(),
   mh->join_query_seqno(),
   mh->join_query_hopcount(),
   mh->join_query_pb(),
   mh->join_reply(),
   mh->join_reply_seqno(),
   mh->join_reply_count(),
   mh->join_reply_ack(),
   mh->join_reply_pairs_src_addr(0),
   mh->join_reply_pairs_prev_hop_addr(0));
}
```

4.8. Implementation of CFODMRP extensions

4.8.1. CFODMRPAgent

CFODMRPAgent class inherits from ODMRPAgent class. In addition, CFODMRPAgent inherits from the Tap class. This enables CFODMRPAgent to implement the case based reasoning and fuzzy logic methods of CFODMRP protocol. cfodmrpagent.h/cc files define the implementation details of CFODMRPAgent.

4.8.2. TCL hooks

As ODMRPAgent, CFODMRPAgent class also defines tcl hooks for instantiation of CFODMRPAgent instances.

```
static class CFODMRPAgentClass : public TclClass {
public:
CFODMRPAgentClass() : TclClass("Agent/CFODMRPAgent") {}
TclObject* create(int, const char*const*) {
return (new CFODMRPAgent);
}
} class_ CFODMRPAgent;
```

4.8.3. Data structures

ODMRPAgent maintains following data structures and above data structures with slight modification.

- **Join Query** packet is implemented as an embedded packet, which could be embedded in CFODMRP header. Join Query packet is defined as follows,

```
struct cfodmrp_join_query {
int valid_;
int seqno_;
int hop_count_;
double power_level;
double bandwidth;
double moving_speed;
};
```

• **fwd_prob** - A hash map, which maps the computed forwarding probability with the IP address of the multicast source. This map is used in subsequent redundant data forwarding.

4.8.4. Processing Additional parameters send by Join query packet

Any node which receives Join Query packet processes the additional parameters added to it to decide whether to select a given node as forwarding group member or no and whether to cache and forward a given join query packet to the next node.

To do these every node has to compute forwarding probability as,

```
If (cfodmrp_join_query.power_level>0.5 &&
      cfodmrp_join_query.bandwidth>4 &&
     cfodmrp_join_query.moving_speed<=6)
{
enterInTable(nsaddr_t grp_addr, nsaddr_t src_addr, nsaddr_t prev_hop, int seqno);
sendcfodmrp_join_query (odmrpaddr_t mcast_grp);
}
else if (cfodmrp_join_query.power_level<=0.3 &&
      cfodmrp_join_query.bandwidth<=2 &&
     cfodmrp_join_query.moving_speed>12)
{
double fw_prob -= 1;
}
else if (cfodmrp_join_query.power_level>0.5 &&
      cfodmrp_join_query.bandwidth<=2 &&
     cfodmrp_join_query.moving_speed==12)
{
// do nothing
}
```

CHAPTER FIVE: RESULTS

In this chapter, we present the performance evaluation of ODMRP and CFODMRP Algorithms using NS-2. First simulation scenarios should be discussed; the movement model and communication model of nodes should be discussed next. Then detail simulation results and analysis should be presented.

5.1. Simulation environment

The simulations were performed in ns2.35 [19] network simulator. Ns-2 is a leading discrete event simulator for network protocol development and analysis.

5.1.1. Scenario

The simulations consist of four test topologies with 100, 200, 300, 400, and 500 mobile nodes in an area of 1000 meter *1000meter. Each test scenario simulates a single multicast group for 80 seconds, where a randomly selected multicast source and 25% of nodes participate in a multicast session. Each simulation was repeated 5 times with 5 different random seeds for ODMRP and CFODMRP. Following results were obtained by normalizing simulation results. It is believed that these configurations are close to real wireless networks with high node mobility. Table 6.1 summarizes different simulation parameters.

5.1.2. Movement model

The random waypoint mobility scenario generator in ns2 was used to produce node movement models with maximum speeds of 10, 15, 20, 25, 30, 35 and 40 m/s, and an average pause time of 10 seconds.

5.1.3. Communication model

The simulation radio model was based on Lucent WaveLAN IEEE 802.11 with a 2Mbps transmission rate and a transmission range of 250 m. The IEEE 802.11 wireless LAN Distributed Coordination Function (DCF) was used as the link layer model. The Carnegie Mellon University's Monarch Research Group's [26] multicast communication scenario generator was used to create multicast communication models.

5.2. Performance Metrics

Following metrics were used to evaluate the performance efficiency of the proposed method.

3. **End to end delay**: The average end-to-end delay from a transmission of the packet to a successful reception at a receiver.
4. **Number of received packets**: Total number of packets which have been received during the simulation period. This allows calculating packet loss rate.
5. **Overall overhead**: Total traffic in network.
6. **Life time**: minimum time until the first node finishes its power.

Table 5.1: A summary of simulation parameters

Parameter	Value
Area	1000m*1000m
Radio Propagation range	250m
Channel capacity	6 Mbits/sec
Pause time	10 sec
Simulation time	80 sec
Packet size	512 bytes
Number of mobile nodes	100, 200, 300, 400, and 500
Traffic type	Constant bit rate
Packet size	512 bytes
Maximum speed	0,5,10, 15, 20, 25, and 30 m/s
Mobility model	Random waypoint

5.3. Simulation Results

5.3.1. Impact of number of nodes

If the number of nodes increases in the network then in ODMRP overhead increases because the flooding domain is increased. In a network with more number of nodes, forwarding group has more members. Also if network density is increased, loss rate is increased. But in the case of CFODMRP transmission of Join query, packets is less due to the use of previous route information stored in CBR, so overhead is less in the case of CFODMRP. This section analyzes performance of CFODMRP under increasing of number of nodes.

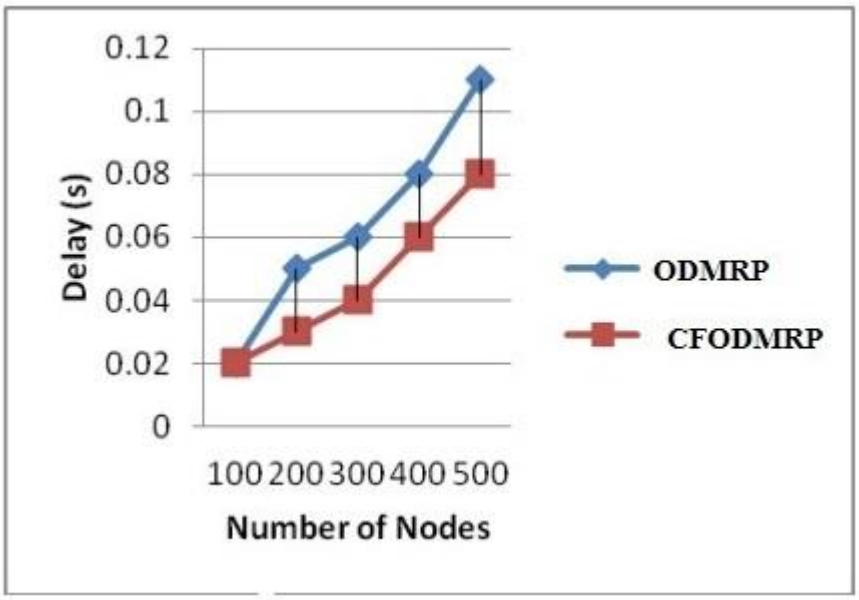

Figure 5.1: Delay vs Number of Nodes

Figure 5.1 compares end to end delay of CFODMRP and ODMRP under increase of number nodes. When number of nodes in network increases, overall overhead of ODMRP increases because the flooding domain is increased. In a network with more number of nodes, forwarding group has more members. Also if network density is increased, loss rate is increased. If number of nodes is less than 300, standard ODMRP shows good performance in terms of delivery ratio because it performs complete flooding in entire network (Fig. 7).Standard ODMRP has good performance while network has sufficient capacity to transfer overall overhead.

When density of network is higher, number of nodes which are included in join query packet forwarding and forwarding group is increasing. This problem is critical when number of nodes is very high. CFODMRP try to use good nodes and routes which lead to higher delivery probability, lower overhead and delay, especially at denser networks.

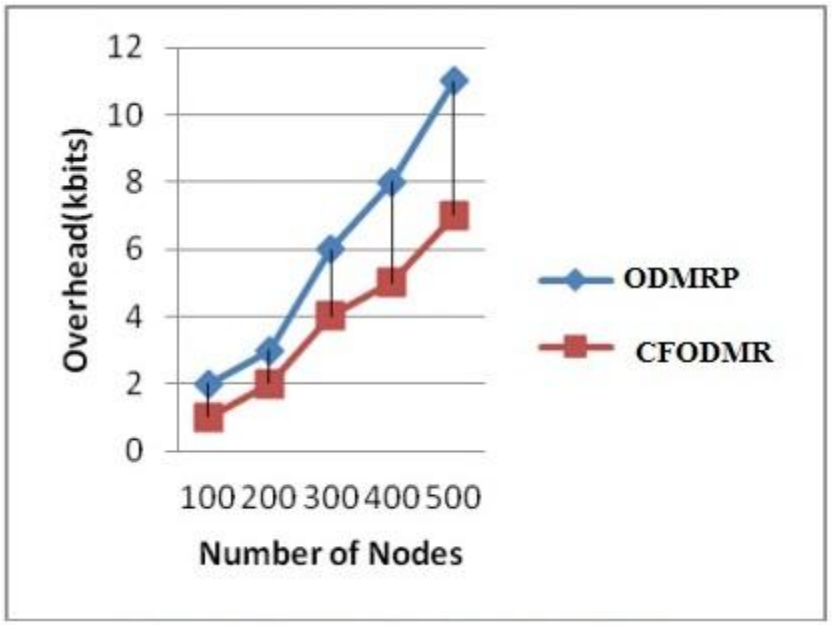

Figure 5.2: Overhead vs Number of Nodes

Figure 5.2 compares overhead of CFODMRP and ODMRP under increase of number of nodes. When the number of nodes increases in the network, overhead increases more in ODMRP compared to CFODMRP.

As a result, standard ODMRP delivers higher overhead (Figure 5.2) to network resulting in higher delay and higher probability for collisions. CFODMRP gives the best performance due to the limitation of the flooding domain by using previous route information stored in CBR.

A higher overhead leads to higher power consumption and lower network life time. Again, CFODMRP has lowest power consumption due to the lowest overhead. While the standard ODMRP is very sensitive to increasing network density the proposed methods are more tolerant.

5.3.2. Impact of speed

Multicast routing protocols faces many challenges with nodes mobility. Nodes move in network with different speed. This section analyzes the performance of CFODMRP under increases of node speed.

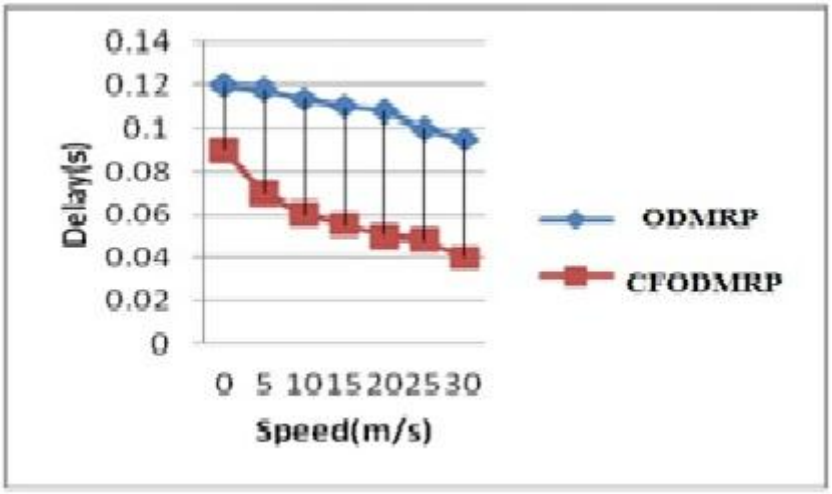

Figure 5.3: Delay vs Speed

Figure 5.3 shows the CFODMRP has less delay compared standard ODMRP. In CFODMRP, the forwarding group can be established from nodes having higher power level, less moving speed, and links having higher bandwidth which leads to decrease repetitive broadcasting of Join query packets. Even though speed of nodes increases because of less mobility of Join query packets delay decreases in case of CFODMRP. But, in standard ODMRP delay is higher than CFODMRP because of more mobility of Join query packets.

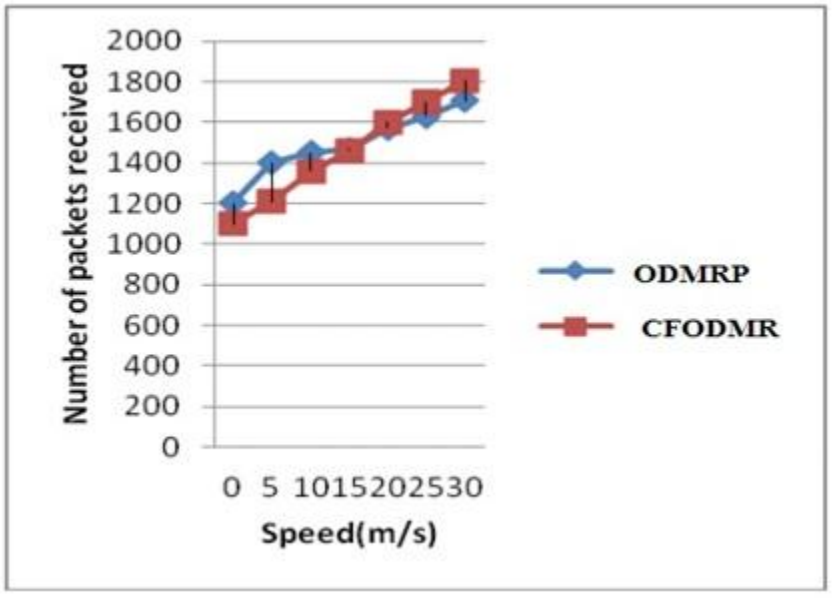

Figure 5.4: Number of packets received vs Speed

Figure 5.4 compares number of packets received in CFODMRP and ODMRP under increase of speed. CFODMRP uses strong forwarding group to transfer data from source to destinations, so breakage of links in CFODMRP is less when compared to ODMRP. So Number of packets received in CFODMRP is more when node speed is more.

In case of high mobility ODMRP suffers from high overhead. This high overhead leads to more power consumption. It leads to frequent node failures, leads to less number of receiving nodes when compared to CFODMRP.

5.3.3 Impact of traffic density

This section analyzes the performance of CFODMRP and standard ODMRP under increase of traffic density.

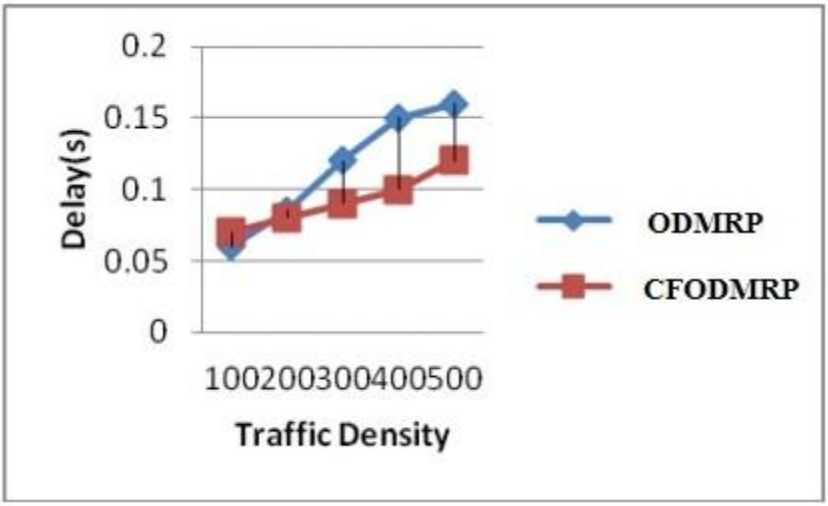

Figure 5.5: Delay vs Traffic density

Figure 5.5 shows end to end delay for all simulated methods. The standard ODMRP floods data packets in a weak forwarding group. The proposed CFODMRP try to form a small and good forwarding group. A small forwarding group has very low overhead in comparison to standard ODMRP. Therefore the proposed method has very low end to end delay in comparison to standard ODMRP.

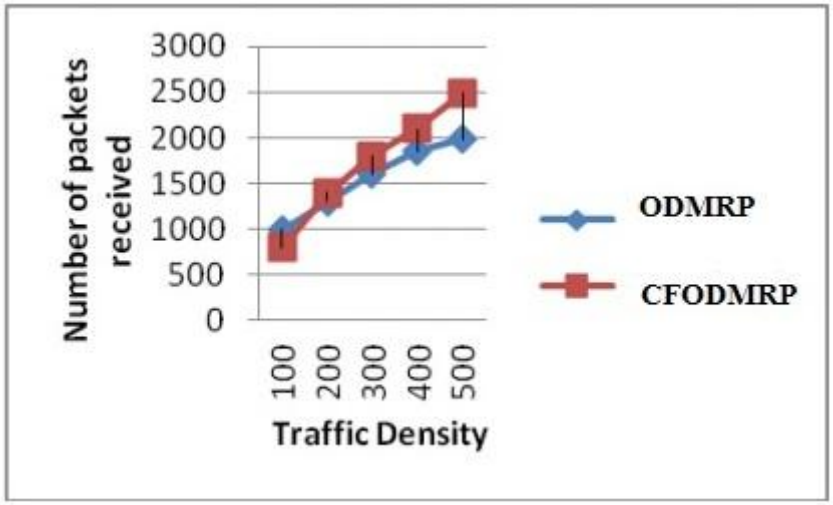

Figure 5.6: Number of packets received vs Traffic density

Figure 5.6present number of received data packet by two methods. The CFODMRP use good nodes and routes to forward data packet. Therefore they deliver very high number of data packets. Also it forms a small and strong forwarding group instead great and weak forwarding group.

CHAPTER SIX: CONCLUSION AND FUTURE WORK

In this research work, Case based and Fuzzy method approaches are used to improve performance of multicast routing algorithms in MANETs specially the ODMRP. Firstly fuzzy logic tries to construct a small and efficient forwarding group instead of great and inefficient one. Secondly case based method tries to reduce the domain of control packet flooding, it achieve more improvement in the reduction of control overhead. These two methods collectively applied on ODMRP and developed Case Based Fuzzy improved ODMRP. Regard to simulation results, Case Based Fuzzy improved ODMRP model has many advantages to the standard ODMRP such as lower control and data overhead, low end to end delay, high packet delivery ratio and very low power consumption. Thus it can be used in many applications in ad- hoc networks.

In the future work, these methods are applied in other MANET routing protocols.

REFERENCES

[1] Chlamtac a, Marco Conti b, Jennifer J-N.Liu c. "*Mobile ad hoc networking: imperatives and challenges*" Istituto IIT,Consiglio Nazionale delle Ricerche, Pisa, Italy 2003.

[2] J.Hoebeke, I. Moerman, B. Dhoedt, P. Demeester, "*An overview of mobile ad hoc networks: Applications and challenges*", Journal-Communications Network 3.3 (2004): 60-66.

[3] Shikha, Vijender Kaushik, "*Reliable Multicasting Protocols in Ad-hoc Networks-Simulated Analysis of Impact of Mobility Speed and Number of Senders*", International Journal of Advanced Research in Computer and Communication Engineering, Vol. 2, Issue 11, November 2013.

[4] S. Jain and K. Agrawal, "*A Survey on Multicast Routing Protocols for Mobile Ad Hoc Networks*" International Journal of Computer Applications, Vol 96, Issue14, June 2014.

[5] PAVAN PICHKA, H.SANTHI, DR.N.JAISANKAR, DEVI PRIYA.S,"*A Comprehensive Study of Existing Multicast Routing Protocols Used In Mobile Ad Hoc Networks*", International Journal of Engineering Science and Technology (IJEST), Vol. 4, No.05, May 2012

[6] N. Sah, N. R. Prakash, and D. Bagai," Analysis of Multicast Routing Protocols Using Quality of Service", International Journal of Signal Processing Systems, Vol. 3, No. 2, December 2015

[7] D.Vodnala, Dr. S. P. Kumar, and S.Aluvala, "An analysis study of various multicast routing protocols for mobile ad hoc networks", International Journal of Emerging Technology and Advanced Engineeing, 2014.

[8] B.Ravi Prasad , A.Damodaram , G.Venkateswara Rao "Fuzzy Improved Adaptive Delay Multicast Routing Protocol", International Journal of Computer Applications , Volume 90, No 4, March 2014

[9] A.Hinds, M.Ngulube, S.Zhu, and H.Al-Aqrabi, "A Review of Routing Protocols for Mobile Ad-Hoc Networks (MANET)", International Journal of Information and Education Technology, February 2013.

[10] A. S. Shafigh, K. Abdollahi and M. Kouchaki, "Improving Performance of On-Demand Multicast Routing Protocol by Fuzzy Logic", World Applied Sciences Journal, 2011.

[11] A.S. Shafigh, K.Abdollahi, and M. Kouchaki, "*Developing a Fuzzy Logic Based on Demand Multicast Routing Protocol",* Hindawi Publishing Corporation Journal of Electrical and Computer Engineering, 2012.

[12] Dr.S.S.Dhenakaran , A.Parvathavarthini “An Overview of Routing Protocols in Mobile Ad-Hoc Network”, International Journal of Advanced Research in Computer Science and Software Engineering, Volume 3, Issue 2, February 2013

[13] M. Nagaratna, V.K. Prasad , and C.R. Rao, “Performance Evaluation of Tree - Based Multicast Routing Protocols in MANETs”. International Journal of Computer Science and technology (IJCST), 2011, 2(3), p.p. 558-562.

[14] A. S. Tanenbaum, (2003), “Computer Networks”, Fourth Edition, 2003.

[15] M. Iiyas, (2003), “The Hand Book of Ad Hoc Networks”, CRC Press, 2003.

[16] B.R.Prasad, Dr. A. Damodaram, and Dr. G. Venkasteswara Rao, “Performance comparison of different multicast routing protocols in mobile ad-hoc networks”, International Journal of Engineering Research and Technology, 2012.

[17] Dharmendra Sutariya,” performance evaluation of multicast routing protocols in mobile ad hoc networks”, international journal of computer networking, wireless and mobile communications (ijcnwmc), vol 6, issue 2, April 2016.

[18] Mohammad M. Qabajeh, Aisha H. Abdalla, Othman O. Khalifa, and Liana K. Qabajeh,” A Survey on Scalable Multicasting in Mobile Ad HocNetworks”, Springer Science+Business Media, New York 2014.

[19] S. Ho Bae, S. Lee, W. Su, and M. Gerla. ,”The Design, Implementation, and PerformanceEvaluation of the On-Demand Multicast Routing Protocol in Multihop Wireless Networks”,In IEEE Network Magazine, volume 14, pages 70–77, January 2000.

[20] S. J. Lee, W. Su, and M. Gerla, ” On-Demand Multicast Routing Protocol (ODMRP) for AdHoc Network IETF Internet Draft”, July 2000.

[21] S. J. Lee, W. Su, and M. Gerla, “On-demand Multicast Routing Protocol in Multi-hop Wireless Mobile Networks”, Mobile Networks and Applications, Vol. 7, No. 6, pp. 441-453, Dec.2002.

[22] S.G. Fentie and V. Sreenivasarao, “A New Approach of Multicast Routing Protocols in MANETS based on Case based Reasoning Method”, Information Systems, Development Informatics & Business Management, Vol.4 Issue2, September 2013.

[23] P. Vashist and K.Hema, “New Multicast Routing Protocol In Ad-Hoc Network”, International Journal of Innovations in Engineering and Technology (IJIET), Vol 2, Issue2, April 2013.

[24] Atta ur Rehman Khan, Sardar M. Bilal, Mazliza Othman “A Performance Comparison of Network Simulators for Wireless Networks”

[25] The VINT project. The network simulator (ns-2). Retrieved July 12, 2016, from http://www.isi.edu/nsnam/ns.

[26] Monarch Research Group. Rice Monarch Project Software Distributions. Retrieved March 01, 2016, from http://www.monarch.cs.cmu.edu/software.html.

APPENDIX

Appendix A

```
# odmrp.tcl
# simulation of wireless ODMRP MANET Routing Protocol
# with 22 nodes,
# Written by Fikreab Lopiso
#===================================
#     Simulation parameters setup
#===================================
set val(chan)   Channel/WirelessChannel    ;# channel type
set val(prop)   Propagation/TwoRayGround   ;# radio-propagation model
set val(netif)  Phy/WirelessPhy            ;# network interface type
set val(mac)    Mac/802_11                 ;# MAC type
set val(ifq)    Queue/DropTail/PriQueue    ;# interface queue type
set val(ll)     LL                         ;# link layer type
set val(ant)    Antenna/OmniAntenna        ;# antenna model
set val(ifqlen) 50                         ;# max packet in ifq
set val(nn)     22                         ;# number of mobilenodes
set val(rp)     ODMRP                       ;# routing protocol
set val(x)      1178                      ;# X dimension of topography
set val(y)      500                      ;# Y dimension of topography
set val(stop)   10.0                        ;# time of simulation end

#===================================
#        Initialization
#===================================
#Create a ns simulator
set ns [new Simulator]

#Setup topography object
set topo       [new Topography]
$topo load_flatgrid $val(x) $val(y)
create-god $val(nn)

#Open the NS trace file
set tracefile [open odmrp.tr w]
$ns trace-all $tracefile

#Open the NAM trace file
set namfile [open odmrp.nam w]
$ns namtrace-all $namfile
$ns namtrace-all-wireless $namfile $val(x) $val(y)
set chan [new $val(chan)];#Create wireless channel

#===================================
#     Mobile node parameter setup
#===================================
$ns node-config -adhocRouting  $val(rp) \
```

```
                -llType        $val(ll) \
                -macType       $val(mac) \
                -ifqType       $val(ifq) \
                -ifqLen        $val(ifqlen) \
                -antType       $val(ant) \
                -propType      $val(prop) \
                -phyType       $val(netif) \
                -channel       $chan \
                -topoInstance  $topo \
                -agentTrace    ON \
                -routerTrace   ON \
                -macTrace      ON \
                -movementTrace ON

#=================================
#        Nodes Definition
#=================================
#Create 22 nodes
set n0 [$ns node]
$n0 set X_ 102
$n0 set Y_ 509
$n0 set Z_ 0.0
$ns initial_node_pos $n0 20
set n1 [$ns node]
$n1 set X_ 270
$n1 set Y_ 495
$n1 set Z_ 0.0
$ns initial_node_pos $n1 20
set n2 [$ns node]
$n2 set X_ 474
$n2 set Y_ 543
$n2 set Z_ 0.0
$ns initial_node_pos $n2 20
set n3 [$ns node]
$n3 set X_ 652
$n3 set Y_ 498
$n3 set Z_ 0.0
$ns initial_node_pos $n3 20
set n4 [$ns node]
$n4 set X_ 169
$n4 set Y_ 310
$n4 set Z_ 0.0
$ns initial_node_pos $n4 20
set n5 [$ns node]
$n5 set X_ 433
$n5 set Y_ 333
$n5 set Z_ 0.0
$ns initial_node_pos $n5 20
set n6 [$ns node]
$n6 set X_ 628
$n6 set Y_ 305
$n6 set Z_ 0.0
```

```
$ns initial_node_pos $n6 20
set n7 [$ns node]
$n7 set X_ 329
$n7 set Y_ 179
$n7 set Z_ 0.0
$ns initial_node_pos $n7 20
set n8 [$ns node]
$n8 set X_ 604
$n8 set Y_ 180
$n8 set Z_ 0.0
$ns initial_node_pos $n8 20
set n9 [$ns node]
$n9 set X_ 810
$n9 set Y_ 200
$n9 set Z_ 0.0
$ns initial_node_pos $n9 20
set n10 [$ns node]
$n10 set X_ 912
$n10 set Y_ 416
$n10 set Z_ 0.0
$ns initial_node_pos $n10 20
set n11 [$ns node]
$n11 set X_ 907
$n11 set Y_ 537
$n11 set Z_ 0.0
$ns initial_node_pos $n11 20
set n12 [$ns node]
$n12 set X_ 797
$n12 set Y_ 552
$n12 set Z_ 0.0
$ns initial_node_pos $n12 20
set n13 [$ns node]
$n13 set X_ 763
$n13 set Y_ 402
$n13 set Z_ 0.0
$ns initial_node_pos $n13 20
set n14 [$ns node]
$n14 set X_ 536
$n14 set Y_ 451
$n14 set Z_ 0.0
$ns initial_node_pos $n14 20
set n15 [$ns node]
$n15 set X_ 470
$n15 set Y_ 187
$n15 set Z_ 0.0
$ns initial_node_pos $n15 20
set n16 [$ns node]
$n16 set X_ 261
$n16 set Y_ 378
$n16 set Z_ 0.0
$ns initial_node_pos $n16 20
set n17 [$ns node]
```

```
$n17 set X_ 952
$n17 set Y_ 321
$n17 set Z_ 0.0
$ns initial_node_pos $n17 20
set n18 [$ns node]
$n18 set X_ 971
$n18 set Y_ 209
$n18 set Z_ 0.0
$ns initial_node_pos $n18 20
set n19 [$ns node]
$n19 set X_ 933
$n19 set Y_ 107
$n19 set Z_ 0.0
$ns initial_node_pos $n19 20
set n20 [$ns node]
$n20 set X_ 732
$n20 set Y_ 66
$n20 set Z_ 0.0
$ns initial_node_pos $n20 20
set n21 [$ns node]
$n21 set X_ 525
$n21 set Y_ 68
$n21 set Z_ 0.0
$ns initial_node_pos $n21 20

#=================================
#        Generate movement
#=================================
$ns at 1.0 " $n0 setdest 70 400 25 "
$ns at 1 " $n20 setdest 900 120 25 "

#=================================
#        Agents Definition
#=================================
#Setup a TCP connection
set tcp0 [new Agent/TCP]
$ns attach-agent $n0 $tcp0
set sink1 [new Agent/TCPSink]
$ns attach-agent $n20 $sink1
$ns connect $tcp0 $sink1
$tcp0 set packetSize_ 1500


#=================================
#        Applications Definition
#=================================
#Setup a FTP Application over TCP connection
set ftp0 [new Application/FTP]
$ftp0 attach-agent $tcp0
$ns at 1.0 "$ftp0 start"
$ns at 2.0 "$ftp0 stop"
```

```
#=================================
#        Termination
#=================================
#Define a 'finish' procedure
proc finish {} {
    global ns tracefile namfile
    $ns flush-trace
    close $tracefile
    close $namfile
    exec nam odmrp.nam &
    exit 0
}
for {set i 0} {$i < $val(nn) } { incr i } {
    $ns at $val(stop) "\$n$i reset"
}
$ns at $val(stop) "$ns nam-end-wireless $val(stop)"
$ns at $val(stop) "finish"
$ns at $val(stop) "puts \"done\" ; $ns halt"
$ns run
```

Appendix B

```
#
# ======================================================================
#  A sample connection pattern file with 20 nodes
# Written by Fikreab Lopiso
# ======================================================================
#
# nodes: 20, max conn: 6, send rate: 0.25, seed: 1.0
#
#
# 1 connecting to 2 at time 2.5568388786897245
#
set udp_(0) [new Agent/UDP]
$ns_ attach-agent $node_(1) $udp_(0)
set null_(0) [new Agent/Null]
$ns_ attach-agent $node_(2) $null_(0)
set cbr_(0) [new Application/Traffic/CBR]
$cbr_(0) set packetSize_ 512
$cbr_(0) set interval_ 0.25
$cbr_(0) set random_ 1
$cbr_(0) set maxpkts_ 10000
$cbr_(0) attach-agent $udp_(0)
$ns_ connect $udp_(0) $null_(0)
$ns_ at 2.5568388786897245 "$cbr_(0) start"
#
# 4 connecting to 5 at time 56.333118917575632
```

```
#
set udp_(1) [new Agent/UDP]
$ns_ attach-agent $node_(4) $udp_(1)
set null_(1) [new Agent/Null]
$ns_ attach-agent $node_(5) $null_(1)
set cbr_(1) [new Application/Traffic/CBR]
$cbr_(1) set packetSize_ 512
$cbr_(1) set interval_ 0.25
$cbr_(1) set random_ 1
$cbr_(1) set maxpkts_ 10000
$cbr_(1) attach-agent $udp_(1)
$ns_ connect $udp_(1) $null_(1)
$ns_ at 56.333118917575632 "$cbr_(1) start"
#
# 4 connecting to 6 at time 146.96568928983328
#
set udp_(2) [new Agent/UDP]
$ns_ attach-agent $node_(4) $udp_(2)
set null_(2) [new Agent/Null]
$ns_ attach-agent $node_(6) $null_(2)
set cbr_(2) [new Application/Traffic/CBR]
$cbr_(2) set packetSize_ 512
$cbr_(2) set interval_ 0.25
$cbr_(2) set random_ 1
$cbr_(2) set maxpkts_ 10000
$cbr_(2) attach-agent $udp_(2)
```

```
$ns_ connect $udp_(2) $null_(2)
$ns_ at 146.96568928983328 "$cbr_(2) start"
#
# 6 connecting to 7 at time 55.634230382570173
#
set udp_(3) [new Agent/UDP]
$ns_ attach-agent $node_(6) $udp_(3)
set null_(3) [new Agent/Null]
$ns_ attach-agent $node_(7) $null_(3)
set cbr_(3) [new Application/Traffic/CBR]
$cbr_(3) set packetSize_ 512
$cbr_(3) set interval_ 0.25
$cbr_(3) set random_ 1
$cbr_(3) set maxpkts_ 10000
$cbr_(3) attach-agent $udp_(3)
$ns_ connect $udp_(3) $null_(3)
$ns_ at 55.634230382570173 "$cbr_(3) start"
#
# 7 connecting to 8 at time 29.546173154165118
#
set udp_(4) [new Agent/UDP]
$ns_ attach-agent $node_(7) $udp_(4)
set null_(4) [new Agent/Null]
$ns_ attach-agent $node_(8) $null_(4)
set cbr_(4) [new Application/Traffic/CBR]
$cbr_(4) set packetSize_ 512
```

```
$cbr_(4) set interval_ 0.25
$cbr_(4) set random_ 1
$cbr_(4) set maxpkts_ 10000
$cbr_(4) attach-agent $udp_(4)
$ns_ connect $udp_(4) $null_(4)
$ns_ at 29.546173154165118 "$cbr_(4) start"
#
# 7 connecting to 9 at time 7.7030203154790309
#
set udp_(5) [new Agent/UDP]
$ns_ attach-agent $node_(7) $udp_(5)
set null_(5) [new Agent/Null]
$ns_ attach-agent $node_(9) $null_(5)
set cbr_(5) [new Application/Traffic/CBR]
$cbr_(5) set packetSize_ 512
$cbr_(5) set interval_ 0.25
$cbr_(5) set random_ 1
$cbr_(5) set maxpkts_ 10000
$cbr_(5) attach-agent $udp_(5)
$ns_ connect $udp_(5) $null_(5)
$ns_ at 7.7030203154790309 "$cbr_(5) start"
#
#Total sources/connections: 4/6
#
```

Appendix C

```
#
==========================================================================
# AWK script to calculate the number of sent packets, received packets,
forwarded packets, dropped packets and Packet delivery Ratio of the
network
#
==========================================================================
BEGIN {
sendLine = 0;
recvLine = 0;
fowardLine = 0;
}
$0 ~/^s.* AGT/ {
sendLine ++ ;
}a
$0 ~/^r.* AGT/ {
recvLine ++ ;
}
$0 ~/^f.* RTR/ {
fowardLine ++ ;
}
END {
printf "cbr send:%d received:%d, PDF:%.4f, forwarded:%d,
dropped:%d \n", sendLine, recvLine, (recvLine/sendLine),fowardLine,
(sendLine-recvLine);}
```