

# Topology Management Using Clustering In Adhoc Network

By

**Nagaria Anjana R.**

**09MCE023**



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING  
NIRMA UNIVERSITY,  
AHMEDABAD-382481**

**May, 2011**

# Topology Management Using Clustering In Adhoc Network

## Major Project

Submitted in partial fulfillment of the requirements

For the degree of

**Master of Technology in Computer Science and Engineering**

By

**Nagaria Anjana R.**

**09MCE023**



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING  
NIRMA UNIVERSITY,  
AHMEDABAD-382481**

**May, 2011**

## Declaration

This is to certify that

- i) The thesis comprises my original work towards the degree of Master of Technology in Computer Science and Engineering at Nirma University and has not been submitted elsewhere for a degree.
- ii) Due acknowledgement has been made in the text to all other material used.

**Nagaria Anjana R.**

## Certificate

This is to certify that the Major Project entitled ” *Topology Management Using Clustering In Adhoc Network*” submitted by *Ms. Nagaria Anjana R. (09MCE023)* , towards the partial fulfillment of the requirements for the degree of Master of Technology in Computer Science and Engineering of Nirma University, Ahmedabad is the record of work carried out by her under my supervision and guidance. In my opinion, the submitted work has reached a level required for being accepted for examination. The results embodied in this major project, to the best of my knowledge, haven’t been submitted to any other university or institution for award of any degree or diploma.

Dr. Sanjay Garg  
Guide, Section Head,  
Department of Computer Engineering,  
Institute of Technology,  
Nirma University, Ahmedabad.

Prof. Sharada Valiveti  
Guide, Associate Professor,  
Department of Computer Engineering,  
Institute of Technology,  
Nirma University, Ahmedabad.

Dr.S.N.Pradhan  
Professor and PG-Coordinator,  
Department of Computer Engineering,  
Institute of Technology,  
Nirma University, Ahmedabad.

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

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

## Abstract

Efficiency of any adhoc network also depends on the management of its moving nodes. The basic idea for topology management is to elect the leader and all the moving nodes' information collected by it. This is basically referred as clustering in adhoc network. Also self organized and distributed approach is applicable for this. A highly dynamic topology is a distinguishing feature and challenge of a mobile ad hoc network. Links between nodes are created and broken, as the nodes move within the network. This node mobility affects not only the source and/or destination, as in a conventional wireless network, but also intermediate nodes, due to the networks' multihop nature. The resulting routes can be extremely volatile, making successful ad hoc routing dependent on efficiently reacting to these topology changes. In order to better understand this environment, a number of characteristics have been studied concerning the links and routes that make up an ad hoc network. Several network parameters are examined, including number of nodes, network dimensions, and radio transmission range, as well as mobility parameters for maximum speed and wait times. Here different clustering techniques are reviewed and summarized to manage the topology of the network which I have proposed as an enhancement to the highest degree clustering scheme which provides load balancing in the network by managing the history about how many times each node has served as cluster head in the network. To manage and update the topology, Cluster structure is updated after fixed interval of time. Simulations are carried out using network simulator GLOMOSIM.

## Acknowledgements

As a human being, it is quite apparent that we have limited knowledge, but we always try to make the most of what we know, I would therefore, like to begin by thanking almighty, who is the ocean of knowledge for helping me.

It gives me immense pleasure in expressing my profound gratitude to my guide **Prof. Sharada Valiveti** and **Dr. Sanjay Garg**, Department of Computer Science and Engineering, Institute of Technology, Nirma University, Ahmedabad for their valuable guidance and continual encouragement throughout my dissertation work. They have devoted significant amount of their valuable time to plan and discuss the thesis work. They have taken the pain to go through the project and make necessary amendments as and when needed. Without their experience and insights, it would have been very difficult to do quality work.

My deep sense of gratitude to **Dr. S. N. Pradhan**, Professor and PG-Coordinator of Department of Computer Engineering, Institute of Technology, Nirma University, Ahmedabad for an exceptional support and continual encouragement throughout my project work.

I would like to thank **Dr. Ketan Kotecha**, Hon'ble Director, Institute of Technology, Nirma University, Ahmedabad for his unmentionable support, providing basic infrastructure and healthy research environment.

I would also thank my Institution, all my faculty members and my colleagues without whom this project would have been a distant reality. Last, no words are enough to acknowledge constant support and sacrifices of my family members and my teachers because of whom I am able to complete my dissertation work.

- **Nagaria Anjana R.**

# Contents

<b>Declaration</b>	<b>iii</b>
<b>Certificate</b>	<b>iv</b>
<b>Abstract</b>	<b>v</b>
<b>Acknowledgements</b>	<b>vi</b>
<b>List of Tables</b>	<b>x</b>
<b>List of Figures</b>	<b>xi</b>
<b>1 Introduction</b>	<b>1</b>
1.1 General . . . . .	1
1.2 Motivation of the work . . . . .	2
1.3 Objective of the work . . . . .	3
1.4 Thesis Organization . . . . .	3
<b>2 Literature Survey</b>	<b>4</b>
2.1 Introduction . . . . .	4
2.1.1 What is topology management? . . . . .	4
2.1.2 How is network managed in LAN? . . . . .	4
2.1.3 Why is network management different in adhoc network? . . . .	6
2.2 Clustering in Ad hoc Network . . . . .	7
2.2.1 What is Clustering? . . . . .	7
2.2.2 Node Role . . . . .	8
2.3 Survey of Clustering Techniques . . . . .	9
2.3.1 Lowest - ID Technique . . . . .	9
2.3.2 Max Degree Heuristic and K - CONID . . . . .	10
2.3.3 Max - Min D Hop clustering . . . . .	10
2.3.4 DCA and DMAC . . . . .	11
2.3.5 WCA and iWCA . . . . .	12
2.3.6 Dominating Set based clustering . . . . .	13
2.4 Observation . . . . .	14

<b>3</b>	<b>Simulator Study</b>	<b>16</b>
3.1	Network Simulators . . . . .	16
3.1.1	NS-2 . . . . .	16
3.1.2	Omnet++ . . . . .	17
3.1.3	Opnet . . . . .	17
3.1.4	QualNet . . . . .	18
3.2	Global Mobile Information System Simulator (GloMoSim) . . . . .	19
3.2.1	What is Glomosim? . . . . .	19
3.2.2	Glomosim Model . . . . .	20
3.2.3	Structure of Glomosim . . . . .	20
3.2.4	Configuration of network . . . . .	21
3.2.5	The Application configuration file . . . . .	22
3.2.6	Configuring mobility . . . . .	22
3.2.7	The Visualization tool . . . . .	23
3.2.8	Statistics . . . . .	23
3.2.9	PARSEC (PARallel Simulation Environment for Complex systems) . . . . .	24
3.3	Implementation . . . . .	26
3.3.1	Hello World Program in Parsec C . . . . .	26
3.3.2	Example of Parsec . . . . .	27
3.3.3	Example of Glomosim . . . . .	28
<b>4</b>	<b>Implementation of LowestID and MaxDegree</b>	<b>31</b>
4.1	Lowest ID and Max Degree Implementation . . . . .	31
4.2	Data Structures Used . . . . .	32
4.2.1	Control Packets generated . . . . .	32
4.2.2	Data structure to store One Hop Neighbor information . . . . .	34
4.3	Implementation of algorithm . . . . .	35
4.3.1	Initializing the data structure used . . . . .	35
4.3.2	Creating one hop neighbor table . . . . .	36
4.3.3	Defining type of the node . . . . .	38
4.3.4	Updating cluster structure . . . . .	39
<b>5</b>	<b>Analysis of Results And Proposed Approach</b>	<b>40</b>
5.1	Results of LowestID & MaxDegree Clustering Algorithm . . . . .	40
5.2	Load balancing in LowestID and MaxDegree . . . . .	42
5.3	Extension to MaxDegree . . . . .	43
<b>6</b>	<b>Conclusion and Future work</b>	<b>45</b>
	<b>Web References</b>	<b>46</b>
	<b>References</b>	<b>48</b>



*CONTENTS*

ix

**Index**

48

# List of Tables

2.1	Observation done for different clustering techniques . . . . .	15
3.1	Models in Glomosim Library . . . . .	20
4.1	Structure of nbrRequest Packet . . . . .	33
4.2	Structure of Status packet . . . . .	33
4.3	Structure of Temp Packet . . . . .	33
4.4	Structure of Degree Packet . . . . .	34
4.5	Structure of Neighbor information stored at each node . . . . .	34
5.1	Simulation Parameter . . . . .	40

# List of Figures

2.1	Cluster formation and node role . . . . .	8
3.1	Configuring network environment[2] . . . . .	21
3.2	Output of hello world . . . . .	26
3.3	Output of prime no generator . . . . .	27
3.4	Initial position of nodes . . . . .	28
3.5	After mobility . . . . .	28
3.6	Changing stat information . . . . .	29
3.7	Statistic generated for new network layer protocol . . . . .	30
4.1	Effect of no. of nbrRequest on neighbor detection . . . . .	36
4.2	Effect of no. of nbrRequest on collision . . . . .	37
5.1	Average number of cluster head in different size of network . . . . .	41
5.2	Comparison of serving factor in LowestID and MaxDegree . . . . .	42
5.3	Comparison of serving factor . . . . .	43
5.4	Average number of cluster head in different size of network . . . . .	44

# Chapter 1

## Introduction

### 1.1 General

The topology of an ad hoc network using clustering plays a key role in the performance of control algorithms in the network. In many cases, not all network links are needed for communication purposes. Weeding out redundant and unnecessary topology information, usually referred to as topology management, can significantly improve the performance of ad hoc networks and sustain network operations over extended period of time. Topology management has been effectively applied in ad hoc networks to supplement routing control protocols and to schedule efficient channel access to propagate broadcast data.

Ad hoc formation of groups of nodes is necessary in the areas of military battlefield networks, sensor networks. These clusters must be formed rapidly and must scale in size while minimizing non-mission-critical messaging overhead. Each cluster has a node that behaves as a cluster head and is the primary point of communication with nodes in other clusters. Cluster heads therefore have the responsibility of routing messages with other cluster heads. The configuration must be stable but it must be able to rapidly reconfigure when the topology changes significantly. For example,

when groups of nodes join or leave, or when the network is partitioned. There should be proper balance between the number of clusters and the size of each cluster. If there are too many clusters, then the routing overhead will be high. But if there are very large clusters, then the messaging overhead for cluster maintenance is likely to be high.

This report contains a general overview of wireless Mobile adhoc network technology, survey of different clustering techniques for ad hoc network, advantages and disadvantages of these techniques, proposed algorithm and description of the simulator used to gather the results.

## 1.2 Motivation of the work

Ad hoc Network has it's own challenging properties like limited battery power, dynamic topology, fast topology changes and all. There are many proactive routing protocols which dynamically finds the routes which may help to survive in dynamic environment. If the structure of the network is maintained in synchronization with the changes in it's topology, then it can be useful for the routing, broadcasting, throughput and delay faced in routing and load balancing in the Ad hoc network. Clustering is useful to maintain the topology of the network where the connections of the network are reduced by selecting proper nodes from the network which connects remaining nodes with it and thus provides the complete connectivity of the network and does the topology management.

### 1.3 Objective of the work

To develop an efficient cluster based and hierarchical Topology Management algorithm considering address of the nodes and the connectivity of each node which balances the load into the network by providing equal priority to each node to serve as cluster head in the network. This may help in saving the energy and using the energy of node equally in the management of the network.

### 1.4 Thesis Organization

The rest of the thesis is organized as follows:

**Chapter 2**, *Literature Survey*, The chapter describes existing clustering techniques, its advantage and limitations, the parameters selected to form the algorithm and performance of algorithm.

**Chapter 3**, *Simulator Study*, focuses on basics of network simulators and choosing the simulator for the implementation.

**chapter 4**, *Implementation of Lowest ID and MaxDegree*, The chapter gives description of the implementation and procedures generated of Lowest ID and Max Degree algorithm and also describes the data structures used for it.

**chapter 5**, *Analysis of Results and Proposed Approach*, The chapter analyzes the results of the Lowest ID and MaxDegree algorithm and gives the description of the newly proposed approach and results of new approach is also described.

**chapter 6**, *Conclusions and Future Work*, Concluding remarks and future work is presented.

# Chapter 2

## Literature Survey

Ad hoc networks are multi hop wireless networks where nodes may be mobile. These networks are used in situations where temporary network connectivity is needed such as in disaster relief or battle site networks. Also it can be useful in civilian environment like conference venues, meeting rooms, boats and small aircrafts etc.

### 2.1 Introduction

#### 2.1.1 What is topology management?

Topology of an ad hoc network plays a key role in the performance of the control algorithms used for routing, transmission, broadcasting etc. Several links are needed for establishing efficient sharing of data packets and weeding out unnecessary and redundant topology information is called Topology Management.

#### 2.1.2 How is network managed in LAN?

Simple Network Management Protocol [11] is the most widely deployed management protocol standard on the Internet. It has a simple API (Application Programming Interface) and it runs over the Transmission Control Protocol (TCP)/IP protocol stack.

The bulk of network management functions involve gathering information from network elements. In SNMP, this information is represented in a structured manner in the Management Information Base (MIB). Every node in the network maintains an MIB that can have information about its current configuration, operation statistics, and parameters to control its functioning. Objects in the MIB are divided into several groups. Each group identifies a specific class of objects or variables that can be accessed by the management station. For example, the interface group contains statistics and configuration information about the physical interface of the entity. Objects in the MIB are defined using the Basic Encoding Rules (BER) associated with ASN.1.

In a typical setup, a node is assigned the responsibility to manage a subnetwork or a set of agents. This node, called the manager node, polls its agents by sending SNMP Protocol Data Units (PDU). The management station can issue three types of PDU: GetRequest, GetNextRequest, and SetRequest. The agent responds to these messages with GetResponse PDU only if it is successful in executing the request. For the GetRequests, the agent sends back the requested value in the response PDU. For SetRequest, the agent sends back the new value of the target object in the set PDU. Agents can also send an unsolicited trap PDU to the management station that informs about the occurrence of an exceptional condition. Since SNMP uses User Datagram Protocol (UDP) as the underlying communication protocol, each PDU sent is treated as an independent request.

In a decentralized management architecture, SNMP provides some facilities for communication between managers. It defines another PDU called InformRequest for manager- to- manager (M2M) communication. The receiving entity responds with a GetResponse PDU. SNMP has also defined an M2M MIB to configure SNMP entities acting in the manager's role.



### 2.1.3 Why is network management different in adhoc network?

Adhoc networks are different than conventional networks due to the following properties.

- Autonomous:

No centralized administration entity is available to manage the operation of the different mobile nodes.

- Dynamic topology:

Nodes are mobile and can be connected dynamically in an arbitrary manner. Links of the network vary timely and are based on the proximity of one node to another node.

- Device discovery:

Bandwidth optimization- Wireless links have significantly lower capacity than the wired links. Identifying relevant newly moved in nodes and informing about their existence need dynamic update to facilitate automatic optimal route selection.

- Limited resources:

Mobile nodes rely on battery power, which is a scarce resource. Also storage capacity and power are severely limited.

- Scalability:

Scalability can be broadly defined as whether the network is able to provide an acceptable level of service even in the presence of a large number of nodes.

- Infrastructure-less and self operated:

Self healing feature demands MANET should realign itself to blanket any node moving out of its range.

- Poor Transmission Quality:

This is an inherent problem of wireless communication caused by several error sources that result in degradation of the received signal.

- Topology maintenance:

Updating information of dynamic links among nodes in MANETs is a major challenge.

## 2.2 Clustering in Ad hoc Network

### 2.2.1 What is Clustering?

Clustering is the method which divides the network into separate or overlapping zones. Clustering selects a set of nodes from the whole network such that from these nodes any of the node of the network is reachable and it does not require to maintain all the links between all the nodes in the network.

The selected subset of the nodes lead to all the other nodes in the network. These leading nodes are called Cluster head. The cluster heads are either directly connected or connected via any other node. These intermediate nodes are called Gateways. Clustering is useful and provides following advantages.

- There is a back bone created considering only special nodes like cluster heads and gateways. So it requires less no of connections to be maintained.
- If cluster based routing is implemented then only cluster heads have to maintain route information.
- Mobility of node affects only when the movement of node is inter cluster.

### 2.2.2 Node Role

There are different node roles assigned to each node while running the clustering algorithm and according to the node role node may contribute in the management of the network.

- **Cluster Head** They are the nodes selected by different clustering techniques to lead the network to create a back bone. This node serves as the head to the subset of ordinary nodes.
- **Gate Ways** These are some of the ordinary nodes which are connected to more than one clusters. Thus they connect two clusters and also contribute in creation of the back bone of network.
- **Ordinary node** The nodes which are connected directly or by k-hop to any of the cluster head in the network are called ordinary nodes.

Figure 2.1 illustrates node role and cluster formation in the network.

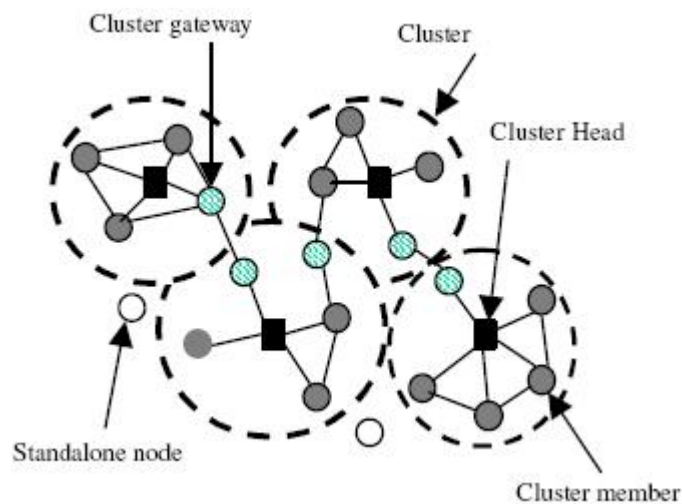


Figure 2.1: Cluster formation and node role

## 2.3 Survey of Clustering Techniques

There are different clustering algorithms already implemented which considers different parameters. This section describes working of some basic clustering algorithm and the performance evaluation of it.

### 2.3.1 Lowest - ID Technique

This is the simplest heuristic technique depended on the ID given to the node of the network. In this algorithm each node has the list of its direct neighbor. Then every node compares its own ID with ID of its neighbors. The node having the Lowest ID becomes the cluster head in the network. The method applied to find out the special nodes is simple but it has some drawbacks. If a node has lowest ID but the highest mobility in the network then it will not provide efficient balancing in the network.

Also it should be considered that the node having Lowest ID moves anywhere in the network It would always be selected as cluster head. So as the cluster head plays the special role in the network it may have more packet reception and delivery work to do. So in the Lowest ID, load balancing is not done due to bias ness of the node having lowest ID.

In LIDAR [1] , authors have proposed an improved lowest ID clustering where they are basically considering the energy and mobility of the node as the parameter. According to these parameters weight is assigned to each node. Now, the ID of the nodes are reassigned such that the node having the highest weight value is assigned lowest ID. Reassigning is limited to each cluster. Thus this method secures the biasness of lowest ID node and also provides load balancing.

### 2.3.2 Max Degree Heuristic and K - CONID

The heuristic is the same as lowest ID but here it is considered that the higher connectivity provides efficient network connectivity. So while electing the cluster heads the parameter considered is the connectivity of each node i.e. degree of each node. Node knows the degree of itself and also the degree of its neighbors. Then the comparison is done between the node connectivity values gathered. The node having the highest connectivity becomes the cluster head and other work as ordinary nodes. Max Degree Provides less clusters compared to lowest ID but there may be ties between nodes who have the same degree, to become cluster heads.

K-CONID [2] has merged two concepts of lowest ID and maximum connectivity of node. In this a pair of ID and degree  $did=(d,id)$  is assigned to each node. First the degree value ( $d$ ) of the pair is compared if any tie occurs then the node having lowest ID is selected as cluster head. This method is extended to  $k$  hop neighbor. The efficiency is tested by calculating the ratios of cluster head nodes to border nodes.

### 2.3.3 Max - Min D Hop clustering

The algorithm, Max-Min D -Cluster Formation in Wireless Ad Hoc Networks[3] elects the cluster head by considering the nodes which are at most  $d$  hop away from itself. Thus it may help to reduce the number of clusters in the network. Algorithm runs for  $2d$  rounds and data structures used are simple i.e. two array WINNER(winning node ID of particular round) and SENDER(node which has sent the winning ID for particular round) at each node of size  $2d$ .

Algorithm runs in four phases. First, the larger node ID is propagated through network in flood Max phase. Second, Lower Id is propagated in flood min. Third, Cluster head selection is done. Fourth back bone is created by linking the clusters.

Cluster head(CH) is decided if node gets its own ID back after 2d rounds of flooding. If this does not happen then node checks for the Node Pair(A node ID which at least occurs as WINNER in both 1st and 2nd round of flooding.). If any node pair does not exists then the node having maximum ID in the 1st d round of flooding is selected as cluster head. For the selection of gateways node checks that itself and its neighbors' cluster head are same or different? If they are same then node is not a gateway but if they are different then node becomes gateway.

Thus the algorithm provides less clusters and good cluster formation but it requires much flooding of messages in the network which may lead to congestion. The special nodes are selected as the local parameter node ID. if the mobility is also considered then the algorithm may give more efficient results.

In a Novel k-Hop Compound Metric Based Clustering Scheme for Ad Hoc Wireless Networks [4], authors have proposed Max- Min based compound metric algorithm for clustering. They have considered the highest connectivity, Node mobility and node ID to select the cluster heads. Thus this heuristic provides small overhead , fast convergence speed and good scalability for large scale networks.

### 2.3.4 DCA and DMAC

DCA and DMAC[8]are two distributed clustering algorithms based on weight assigned to each node and the mobility parameter considered. DCA is basically suitable for the network whose nodes move "slowly" or do not moves. But DMAC algorithm adapts to the changes in the network due to node mobility.

In DCA it is assumed that each node knows it's own ID , neighbor's ID and its weight which is any real number  $\geq 0$ .The node having highest weight in its one hop neighbors declares its node role as cluster head.

DMAC[8] adapts to the changes in the topology of the network due to mobility of the nodes. The cluster head selection process is same as DCA but DMAC is event driven which means it send two messages : i) send JOIN message when a node joins to any cluster and ii) Sends CH message when a node selects it's role as cluster head. The algorithm runs five procedures : Init , Link failure procedure, New\_Link procedure, The procedure done when receiving CH message and the procedure done when receiving JOIN message. One major advantage of this algorithm is that it allows mobility of nodes during cluster setup phase also. This is done by allowing nodes to react to the reception of packets from the other nodes and also to the link failure of a link with the other node.

### 2.3.5 WCA and iWCA

A Weight based clustering algorithm [5] is a clustering scheme which takes ideal degree , transmission power , mobility and battery power of a node into consideration for making clusters. It may also be called as combined metric based clustering as it considers more than one parameters into account. The algorithm takes degree difference with predefined ideal degree which provides load balancing. The total distance taken as a parameter provides energy aspect to the algorithm. Mobility serves to make algorithm synchronized with the mobility of nodes of network and also they consider the time a node had served as cluster head into network. Thus all the challenging parameters of mobile Ad hoc network are taken into consideration to form the clusters and maintain them So the algorithm performs better than all other algorithms described.

As there is a disadvantage with the WCA that the high mobility leads to high reaffiliation So, In iWCA [6], authors have proposed an improved WCA clustering scheme which maintains the clusters by considering two aspects. They reaffiliate the

clusters when either an ordinary node moves out of the current cluster heads or the cluster head runs out of battery power. This improves the stability of the network and also decreases the reaffiliation in the network.

### 2.3.6 Dominating Set based clustering

Dominating Set Theory of graphs are mainly used for the formation of clusters as it provides distributed aspect to find out leading nodes and maintaining complete connectivity in the network and also reduces the required links to manage.

In this method the network is treated as a graph  $G=(V,E)$  where  $V$  is the set of all nodes in the network and  $E$  is the set of edges of network.

- Dominating set: The subset( $S$ ) of nodes ( $G$ ) such that each node of network (or graph  $G$ ) is reachable from any of the node of subset ( $S$ ). There may be more than one dominating set of any graph  $G$ . Among them the dominating set having less number of members is called Minimal Dominating set. If the nodes of the dominating set are connected directly then the set is called Connected Dominating Set. If no two vertices of the dominating set are adjacent to each other then the set is called Independent Dominating Set[7].

In a Dominating Set Based Clustering Algorithm for Mobile Ad hoc Networks [7], authors have proposed a connected dominating set based clustering algorithm in which they do coloring of nodes as part of formation of clusters. Algorithm colors the node as white if the node is not in the dominating set, Black if the node is in the dominating set and Gray shows that still node does not have decided to be either in the dominating set or not.



## 2.4 Observation

As the survey of different algorithms is done, all clustering algorithm mainly try to divide the network in such a way that maximum connectivity can be achieved in the presence of mobility of the node by handling the battery and energy constraints of the nodes of the network. The algorithms mainly have two major parts.

- Cluster formation: Generally using graph theoretic approach of dominating sets[7] and spanning trees[9] to finding out the leading nodes in the network.
- Cluster Maintenance: This process is used to either periodically or on demand reform and select the cluster heads in the network.

The other main aspect to observe performance of any algorithm is the parameters used to form the clusters. Resulting clusters may be overlapping or non overlapping as per the methods used to form the cluster. Table 2.1 describes the summary of the techniques and parameter wise Classification of the clustering schemes.

Dominating set	<ul style="list-style-type: none"> <li>• Independent DS</li> <li>• Connected DS</li> <li>• Weakly Connected DS</li> </ul>
Hop count	<ul style="list-style-type: none"> <li>• 1-hop .(Can save energy but creates more clusters)</li> <li>• Multi hop.(Energy aware)</li> </ul>
Node role	<ul style="list-style-type: none"> <li>• Cluster head based .</li> <li>• Fully distributed.(but requires generally more rounds for clustering.)</li> </ul>
Energy aware	<ul style="list-style-type: none"> <li>• Mostly uses DS.</li> <li>• Load balancing.</li> <li>• Also have hop restrictions.</li> <li>• Can be implemented with some counter which builds restrictions for working time of a CH.</li> </ul>
Mobility Control	Have some Equations to find the mobility of nodes & based on that Clusters are formed. Generally node with lowest speed of mobility is selected as CH in the nbrs.
Passive clustering	Does not need any extra control messages for the cluster formation & maintenance.
Combined Metric based clustering	More than one factors like energy & mobility & all that are considered for the cluster formation. Which are going to be formed into an equation & which helps to do clustering.
No. of round for clustering	Related to the fact that how many constant or dynamic round are required for the cluster formation . Which is related to the important assumption of stability of node while cluster forming If rounds are more then the time required for the stability increases which does not provides good result.

Table 2.1: Observation done for different clustering techniques

# Chapter 3

## Simulator Study

Because of the complex nature of the MANETs, their simulation is a very challenging issue[13][16]. Simulators rely on various techniques for improving their accuracy, speed, scalability, usability, etc. This section gives an overview of some well known network simulators.

### 3.1 Network Simulators

#### 3.1.1 NS-2

NS simulator is based on two languages: an object oriented simulator, written in C++, and a OTcl(an object oriented extension of Tcl) interpreter, used to execute users command scripts. NS has a rich library of network and protocol objects. There are 2 class hierarchies: the compiled c++ hierarchy and the interpreted OTcl one, with one to one correspondence between them.

The OTcl script is provided by the user and we can define a particular network topology, the specific protocols and applications that we wish to simulate and the form of the output that we wish to obtain from the simulator. The OTcl can make use of the objects compiled in C++ through an OTcl linkage that creates a matching

of OTcl object for each of the C++.

NS makes use of flat earth model in which it assumes that the environment is flat without any elevations or depressions. However the real world does have geographical features like valleys and mountains. To run simulations involves the laborious process of learning a scripting language, followed by having to understand in detail the inner working of NS[13][16].

### 3.1.2 Omnet++

Omnet++ is an object-oriented modular discrete event network simulation framework. It has a generic architecture, so it can be (and has been) used in various problem domains like modeling of wired and wireless communication networks and protocol modeling. OMNeT++ itself is not a simulator of anything concrete, but it rather provides infrastructure and tools for writing simulations. One of the fundamental ingredients of this infrastructure is component architecture for simulation models. Models are assembled from reusable components termed as modules. Well-written modules are truly reusable OMNeT++ runs on Linux, Mac OS, and other Unix-like systems and on Windows (XP, Win2K, Vista, 7)[13][16].

### 3.1.3 Opnet

Opnet is the simulator specialized for network research and development. It allows us to design and study communication networks, devices, protocols, and applications with great flexibility. It provides a graphical editor interface to build models for various network entities from physical layer modulator to application processes. All the components are modeled in an object-oriented approach which gives intuitive easy mapping to real systems. It gives a flexible platform to test new ideas and solutions with low cost[13][16].

The disadvantage of using OPNET is that the simulation requires a lot of processing power and can be very time consuming particularly for network with a large number of transmitter and receivers.

### 3.1.4 QualNet

QualNet is the first commercial simulator in this comparison. It is based on GloMoSim developed at the University of California, Los Angeles (UCLA). GloMoSim uses the Parallel Simulation Environment for Complex Systems (PARSEC) for basic operations. QualNet also has a graphical user interface for creating the model and its specification. So it is by far easier to specify small to medium networks by using the GUI compared to specifying all connections in a special model file manually[13][16].

QualNet is a network simulator targeting at wireless solutions, however it also has support for wired networks. Its environment and library is very sophisticated, which makes it very easy to simulate a real network with QualNet. This however makes the simulation of logical networks a little bit more difficult, but it is possible as well.

Since it uses primarily Java for the GUI it is available for Linux as well as for Windows. The simulator itself is for the specified target system optimized C program.

## 3.2 Global Mobile Information System Simulator (GloMoSim)

### 3.2.1 What is Glomosim?

Glomosim is a library-based sequential and parallel simulator for wireless networks. This has been developed using PARSEC, a C-based parallel simulation language. Glomosim can be modified to add new protocols and applications to the library[17]. GloMoSim is the Global Mobile Information Systems Simulation Library from UCLA. GloMoSim currently supports protocols for a wired, purely wireless network and hybrid network with both wired and wireless capabilities. It uses a parallel discrete event simulation capability provided by parsec. It simulates networks with up to thousand nodes linked by a heterogeneous communications capability.

The node aggregation technique is used in Glomosim to give significant benefits to the simulation performance[17]. Initializing each node as a separate entity inherently limits the scalability because the memory requirements increase dramatically for a model with large number of nodes. With node aggregation, a single entity can simulate several network nodes in the system. This technique implies that the number of nodes in the system can be increased while maintaining the same number of entities in the simulation.

Glomosim uses PARSEC which is a Parallel Simulation Environment for Complex systems. It is a C-based discrete-event simulation language. It adopts the process interaction approach to discrete-event simulation. PARSEC has the ability to execute a discrete-event simulation model using several different asynchronous parallel simulation protocols on a variety of parallel architectures. PARSEC is designed to cleanly separate the description of a simulation model from the underlying simulation protocol, sequential or parallel, used to execute it. It provides powerful message

receiving constructs that result in shorter and more natural simulation programs[17].

There are some advantages of glomosim which makes it to be used rather than any other network simulators[17][18].

- Scalable simulation environment
- Supports Wired & Wireless network
- Layered approach
- Standard APIs
- Parallel discrete-event simulation

### 3.2.2 Glomosim Model

The following table lists the Glomosim models currently available at each of the major layers:

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

Table 3.1: Models in Glomosim Library

### 3.2.3 Structure of Glomosim

How the structure of glomosim is organized is described below[12].

- */doc* Contains the documentation.
- */scenarios* Contains directories of various sample configuration topologies.

- **/main** Contains the basic framework for GloMosim.
- **/bin** Contains the executables and the input/output files.
- **/include** Contains common include files.
- **/application** contains code for the application layer i.e. files for traffic generation.
- **/transport** contains the code for the transport layer.
- **/network** contains the code for the network layer.
- **/mac** contains the code for the MAC layer, including 802.11b.
- **/radio** contains the code for the physical layer.

### 3.2.4 Configuration of network

The GloMoSim configuration file `config.in` can be located under the path `glomosim-2.03/glomosim/bin`. It provides the configuration parameters for the simulation[12].

[config.in]	
NUMBER-OF-NODES	3
NODE-PLACEMENT	FILE
NODE-PLACEMENT-FILE	./nodes.input
#NODE-PLACEMENT	UNIFORM
MAC-PROTOCOL	802.11
NETWORK-PROTOCOL	IP
ROUTING-PROTOCOL	BELLMANFORD
APP-CONFIG-FILE	./app.conf
RADIO-TYPE	RADIO-ACCNOISE

[nodes.input]	[app.conf]
0 0 (20.2, 0.9, 0.11)	#CBR <src_node> <dest_node>
1 0 (80.4, 90.8, 0.17)	# <items> <item_size> <interval_time>
2 0 (60.7, 30.4, 0.10)	# <start_time> <end_time>
	CBR 0 1 10 512 1S 0S 0S

Figure 3.1: Configuring network environment[2]



### 3.2.5 The Application configuration file

Applications such as FTP and Telnet are configured in this file. The traffic generators currently available are FTP, FTP/GENERIC, TELNET, CBR, and HTTP. In most simulations, we use CBR data traffic and send data packets at regular time intervals. The app.conf file located in the bin folder provides a good explanation of how it can specify data traffic[12].

### 3.2.6 Configuring mobility

For static networks, mobility should be set to NONE as follows[12]:

**MOBILITY NONE**

GloMoSim provides the Random Waypoint mobility model, in which the nodes move towards a destination with a randomly generated speed (within the specified limits), and then pause there for some time. To use the Random Waypoint model, the parameters for mobility should be set as follows:

**MOBILITY RANDOM-WAYPOINT**

**MOBILITY-WP-PAUSE 30S**

**MOBILITY-WP-MIN-SPEED 0**

**MOBILITY-WP-MAX-SPEED 10**

By using a pause time of 0S (0 seconds), continuous random motion can be emulated. The maximum speed is usually around 20 m/s, which is equivalent to 72 km/h (approximately the average speed of a vehicle).

Another mobility model (easily obtained by using the Bonn Motion Software), specify the input trace files as follows:

## MOBILITY TRACE

### MOBILITY-TRACE-FILE `./mobility.in`

In this file parameters can be given from the current position with how much speed and how much displacement of the nodes will be done.

### 3.2.7 The Visualization tool

GloMoSim has a Visualization Tool that is platform independent because it is coded in Java[12]. To initialize the Visualization Tool, we must execute: `java GlomoMain` from the `java gui` directory. This tool allows to debug and verify models and scenarios; stop, resume and step execution; show packet transmissions, show mobility groups in different colors and show statistics.

The radio layer is displayed in the Visualization Tool as follows: When a node transmits a packet, a yellow link is drawn from this node to all nodes within its power range. As each node receives the packet, the link is erased and a green line is drawn for successful reception and a red line is drawn for unsuccessful reception. No distinction is made between different packet types (ie: control packets vs. regular packets, etc)

### 3.2.8 Statistics

This section allows us to specify the types of statistics which we are interested in at the end of the simulation. The statistics will be consolidated in the `glomo.stat` file, which can be found in the `bin` folder at the end of each simulation. The types of statistics available include:

- TCP
- UDP

- Routing
- Network
- MAC
- Radio
- Channel
- Mobility

### 3.2.9 PARSEC (PARallel Simulation Environment for Complex systems)

PARSEC (for PARallel Simulation Environment for Complex systems) is a C-based discrete-event simulation language. It adopts the process interaction approach to discrete-event simulation. An object (also referred to as a physical process) or set of objects in the physical system is represented by a logical process[15].

Interactions among physical processes (events) are modeled by timestamped message exchanges among the corresponding logical processes. One of the important distinguishing features of PARSEC is its ability to execute a discrete-event simulation model using several different asynchronous parallel simulation protocols on a variety of parallel architectures. PARSEC is designed to cleanly separate the description of a simulation model from the underlying simulation protocol, sequential or parallel, used to execute it.

Thus, with few modifications, a PARSEC program may be executed using the traditional sequential (Global Event List) simulation protocol or one of many parallel optimistic or conservative protocols. In addition, PARSEC provides powerful message receiving constructs that result in shorter and more natural simulation programs.

Useful debugging facilities are available. A front-end for visual specification of simulation models, and a runtime output display and visualization environment are currently being designed. The PARSEC language is derived from Maisie, but with several improvements, both in the syntax of the language and in its execution environment. Appendix C contains information about converting existing Maisie programs into PARSEC.

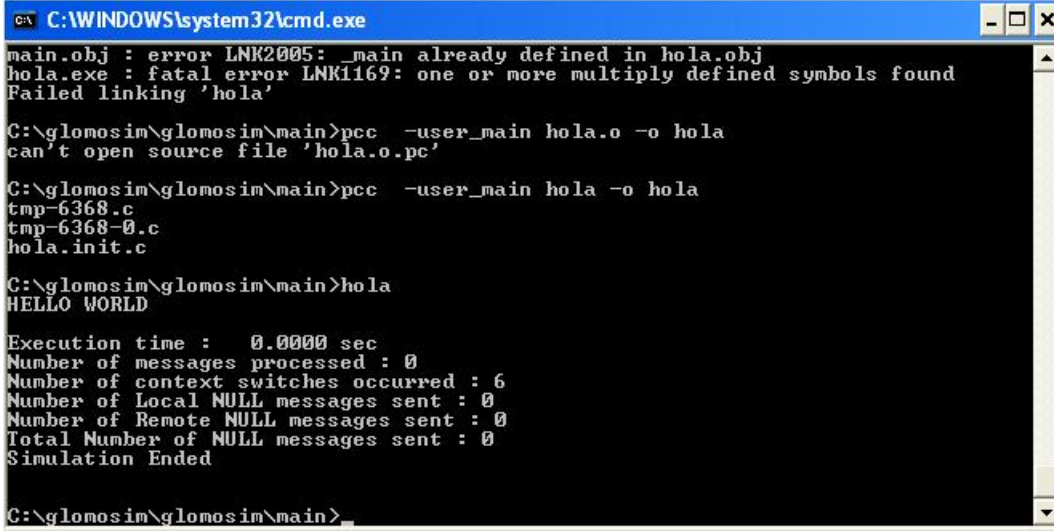
Every Parsec program must include an entity called driver which serves a purpose similar to the main function of a C program. This file reads the configuration file, initiates the simulation and writes the final statistics in the result file `glomostat` from some temporary file (`.STAT.x`). The sequence of events at run time is as follows:

- The main function in `driver.pc` is run. This is the C main function, where GloMoSim starts
- The main function calls `parsec main()` to start the Parsec simulation engine, initializes the simulation runtime variables and creates the driver entity method. The `parsec main` function is used when the user wants to write his own main and is found at `PCC_DIRECTORY/include/pc/api.h` (since the function is part of the Parsec runtime system, it is not possible to access the source for it).
- When the simulation ends, `parsec main()` returns, and the rest of the main function is executed.

## 3.3 Implementation

### 3.3.1 Hello World Program in Parsec C

This program simply printed hello world. This is the practice done to understand the structure of parsec programs. The output of the program is shown in figure 3.2.



```
C:\WINDOWS\system32\cmd.exe
main.obj : error LNK2005: _main already defined in hola.obj
hola.exe : fatal error LNK1169: one or more multiply defined symbols found
Failed linking 'hola'

C:\glomosim\glomosim\main>pcc -user_main hola.o -o hola
can't open source file 'hola.o.pc'

C:\glomosim\glomosim\main>pcc -user_main hola -o hola
tmp-6368.c
tmp-6368-0.c
hola.init.c

C:\glomosim\glomosim\main>hola
HELLO WORLD

Execution time : 0.0000 sec
Number of messages processed : 0
Number of context switches occurred : 6
Number of Local NULL messages sent : 0
Number of Remote NULL messages sent : 0
Total Number of NULL messages sent : 0
Simulation Ended

C:\glomosim\glomosim\main>
```

Figure 3.2: Output of hello world

### 3.3.2 Example of Parsec

Figure 3.3 shows the output of the program to generate all prime numbers less than 1000. The first instance of the Sieve entity is created by the driver and subsequent instances are created recursively such that the first number received by a new instance is a prime number.

```

C:\WINDOWS\system32\cmd.exe
Sieve number 117 is for prime number 641
Sieve number 118 is for prime number 643
Sieve number 119 is for prime number 647
Sieve number 120 is for prime number 653
Sieve number 121 is for prime number 659
Sieve number 122 is for prime number 661
Sieve number 123 is for prime number 673
Sieve number 124 is for prime number 677
Sieve number 125 is for prime number 683
Sieve number 126 is for prime number 691
Sieve number 127 is for prime number 701
Sieve number 128 is for prime number 709
Sieve number 129 is for prime number 719
Sieve number 130 is for prime number 727
Sieve number 131 is for prime number 733
Sieve number 132 is for prime number 739
Sieve number 133 is for prime number 743
Sieve number 134 is for prime number 751
Sieve number 135 is for prime number 757
Sieve number 136 is for prime number 761
Sieve number 137 is for prime number 769
Sieve number 138 is for prime number 773
Sieve number 139 is for prime number 787
Sieve number 140 is for prime number 797
Sieve number 141 is for prime number 809
Sieve number 142 is for prime number 811
Sieve number 143 is for prime number 821
Sieve number 144 is for prime number 823
Sieve number 145 is for prime number 827
Sieve number 146 is for prime number 829
Sieve number 147 is for prime number 839
Sieve number 148 is for prime number 853
Sieve number 149 is for prime number 857
Sieve number 150 is for prime number 859
Sieve number 151 is for prime number 863
Sieve number 152 is for prime number 877
Sieve number 153 is for prime number 881
Sieve number 154 is for prime number 883
Sieve number 155 is for prime number 887
Sieve number 156 is for prime number 907
Sieve number 157 is for prime number 911
Sieve number 158 is for prime number 919
Sieve number 159 is for prime number 929
Sieve number 160 is for prime number 937
Sieve number 161 is for prime number 941
Sieve number 162 is for prime number 947
Sieve number 163 is for prime number 953
Sieve number 164 is for prime number 967
Sieve number 165 is for prime number 971
Sieve number 166 is for prime number 977
Sieve number 167 is for prime number 983
Sieve number 168 is for prime number 991
Sieve number 169 is for prime number 997
Execution time : 0.1250 sec
Number of messages processed : 20482
Number of context switches occurred : 849
Number of Local NULL messages sent : 0
Number of Remote NULL messages sent : 0
Total Number of NULL messages sent : 0
NULL messages / Regular messages : 0.000

```

Figure 3.3: Output of prime no generator

### 3.3.3 Example of Glomosim

Figure 3.4 and 3.5 show the scenario of how nodes can be placed in Glomosim. The figure also shows the manual node placement in the terrain area. The yellow line shows the link between the nodes.

- Initially

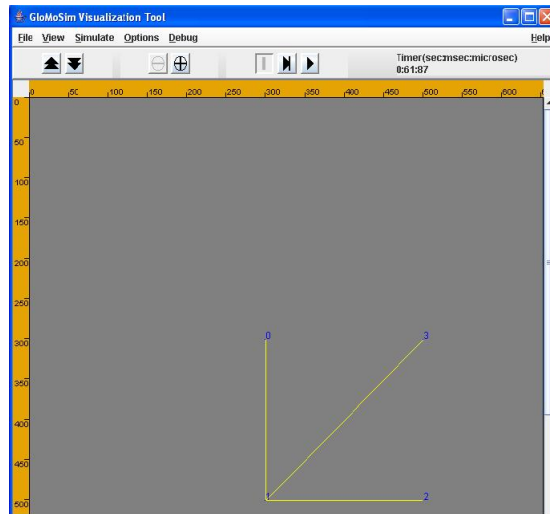


Figure 3.4: Initial position of nodes

- After Mobility

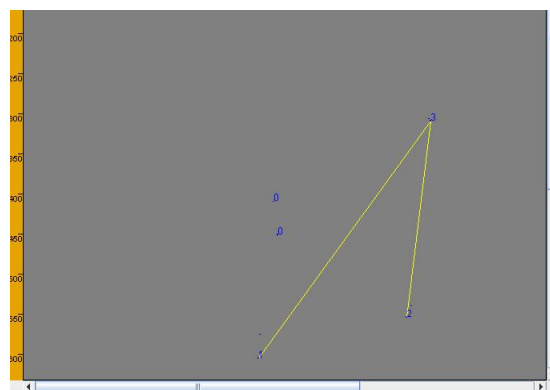


Figure 3.5: After mobility

- **Changing stat information:**

Glomosim generates the statistic file at the end of the simulation. If we want to add any statistic at any layer, then a variable is created and it is printed into the stat function of specific layer. Thus any statistic can be added into the stat file generated by the Glomosim. Figure 3.6 shows that a "prime" statistic is added at MAC layer.

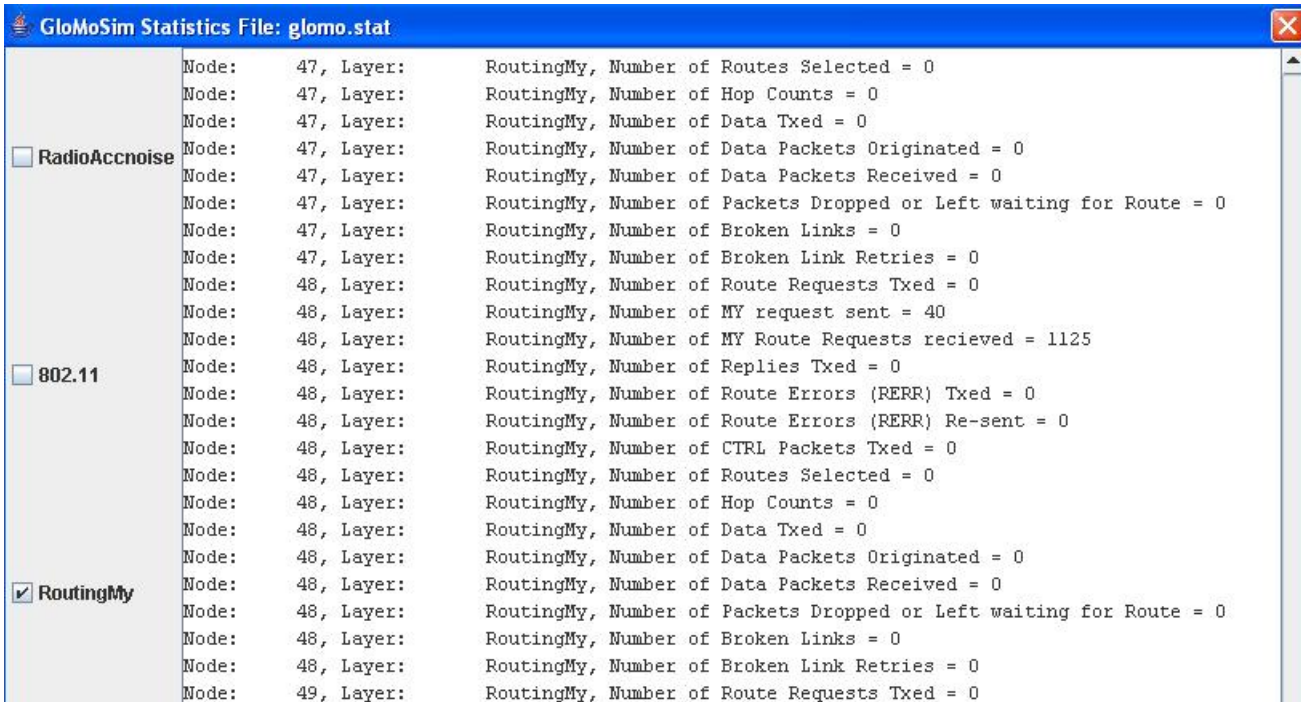
<input type="checkbox"/> RadioAccnoise	Node: 0, Layer:	802.11, prime: 61
	Node: 0, Layer:	802.11, pkts from network: 0
	Node: 0, Layer:	802.11, UCAST (non-frag) pkts sent to chanl: 10
	Node: 0, Layer:	802.11, BCAST pkts sent to chanl: 6
	Node: 0, Layer:	802.11, UCAST pkts rcvd clearly: 10
	Node: 0, Layer:	802.11, BCAST pkts rcvd clearly: 11
<input checked="" type="checkbox"/> 802.11	Node: 0, Layer:	802.11, retx pkts due to CTS timeout: 0
	Node: 0, Layer:	802.11, retx pkts due to ACK timeout: 0
	Node: 0, Layer:	802.11, pkt drops due to retx limit: 0
	Node: 0, Layer:	802.11, RTS Packets ignored due to Busy Channel 0
	Node: 0, Layer:	802.11, RTS Packets ignored due to NAV 0
	Node: 0, Layer:	802.11, RTS Packets ignored due to NAV 0
<input type="checkbox"/> NetworkIp	Node: 1, Layer:	802.11, prime: 65
	Node: 1, Layer:	802.11, pkts from network: 0
	Node: 1, Layer:	802.11, UCAST (non-frag) pkts sent to chanl: 10
	Node: 1, Layer:	802.11, BCAST pkts sent to chanl: 7
<input type="checkbox"/> TransportUdp	Node: 1, Layer:	802.11, UCAST pkts rcvd clearly: 10
	Node: 1, Layer:	802.11, BCAST pkts rcvd clearly: 15
	Node: 1, Layer:	802.11, retx pkts due to CTS timeout: 0
	Node: 1, Layer:	802.11, retx pkts due to ACK timeout: 0
	Node: 1, Layer:	802.11, pkt drops due to retx limit: 0
<input type="checkbox"/> TransportTcp	Node: 1, Layer:	802.11, RTS Packets ignored due to Busy Channel 0
	Node: 1, Layer:	802.11, RTS Packets ignored due to NAV 0
	Node: 1, Layer:	802.11, RTS Packets ignored due to NAV 0
	Node: 2, Layer:	802.11, prime: 65

Figure 3.6: Changing stat information



- How to hook a new protocol into Glomosim?

New protocol can be added into Glomosim by different ways. One way is to use the available User\_Protocol file and to put the functions and variables there. The other way is to create the .pc and .h file of new protocol and mark the entries into corresponding files in the code of Glomosim. Figure 3.7 shows the statistics of newly added network layer protocol named "MY". The protocol contains the code of aodv but it is named other way[19].



Protocol	Node	Layer	Statistic	Value
RadioAccnoise	47	Layer:	RoutingMy, Number of Routes Selected = 0	0
	47	Layer:	RoutingMy, Number of Hop Counts = 0	0
	47	Layer:	RoutingMy, Number of Data Txed = 0	0
	47	Layer:	RoutingMy, Number of Data Packets Originated = 0	0
	47	Layer:	RoutingMy, Number of Data Packets Received = 0	0
	47	Layer:	RoutingMy, Number of Packets Dropped or Left waiting for Route = 0	0
	47	Layer:	RoutingMy, Number of Broken Links = 0	0
802.11	47	Layer:	RoutingMy, Number of Broken Link Retries = 0	0
	48	Layer:	RoutingMy, Number of Route Requests Txed = 0	0
	48	Layer:	RoutingMy, Number of MY request sent = 40	40
	48	Layer:	RoutingMy, Number of MY Route Requests recieved = 1125	1125
	48	Layer:	RoutingMy, Number of Replies Txed = 0	0
	48	Layer:	RoutingMy, Number of Route Errors (RERR) Txed = 0	0
	48	Layer:	RoutingMy, Number of Route Errors (RERR) Re-sent = 0	0
RoutingMy	48	Layer:	RoutingMy, Number of CTRL Packets Txed = 0	0
	48	Layer:	RoutingMy, Number of Routes Selected = 0	0
	48	Layer:	RoutingMy, Number of Hop Counts = 0	0
	48	Layer:	RoutingMy, Number of Data Txed = 0	0
	48	Layer:	RoutingMy, Number of Data Packets Originated = 0	0
	48	Layer:	RoutingMy, Number of Data Packets Received = 0	0
	48	Layer:	RoutingMy, Number of Packets Dropped or Left waiting for Route = 0	0
RoutingMy	48	Layer:	RoutingMy, Number of Broken Links = 0	0
	48	Layer:	RoutingMy, Number of Broken Link Retries = 0	0
	49	Layer:	RoutingMy, Number of Route Requests Txed = 0	0

Figure 3.7: Statistic generated for new network layer protocol

## Chapter 4

# Implementation of LowestID and MaxDegree

### 4.1 Lowest ID and Max Degree Implementation

As described in the literature survey (section 2.3.1 and section 2.3.2) while forming the clusters, each node has information about its one hop neighbor. In Lowest ID scheme, the node with the lowest ID among its neighbors is selected as cluster head while in Max Degree(K-CONID), the node with the highest degree (connectivity) among its neighbors is selected as cluster head. If any node in the neighbor list has the same degree, the node with the lowestID among those nodes is selected as cluster head.

These algorithms basically generate dominating set of the graph where the network connections and nodes are included in the graph as  $G=(V,E)$  where  $V$  is the nodes of the network and  $E$  is connection between nodes.

As the cluster heads are selected they form the dominating set of the graph  $G$ . As described in the literature survey, Dominating set is the subset of graph  $G$  such that

all the other nodes are reachable from the nodes which are members of the dominating set.

These nodes of dominating set may not be connected so, the subset cannot be said as connected dominating set. It is required to have connected dominating set to create back bone of the network. So to do this, some common nodes which are neighbor of more than one cluster head node are selected as gateway into the network.

The proof of correctness of dominating set and back bone created.

- **Proof of correctness of Dominating set generated:**

As the cluster head makes the dominating set and the cluster heads are elected from one hop neighbor table so each node is directly connected with either of the cluster heads in the network.

- **Proof of the correctness of back bone created:**

As the cluster head are selected from one hop neighbor list, the distance between two cluster head is maximum two hop. So the node which connects to two cluster heads create a connected dominating set and provides back bone topology of network.

This chapter describes the implementation details of the clustering scheme in Glomosim simulator.

## 4.2 Data Structures Used

### 4.2.1 Control Packets generated

Following control packets are sent for the formation of Clusters.

- **MY\_nbrRequest Packet:** It is same like hello packet and contains its own ID and broadcasted in the network.

MY_nbrRequest Packet	<ul style="list-style-type: none"> <li>– Packet Type</li> <li>– Source Address</li> </ul>
----------------------	---

Table 4.1: Structure of nbrRequest Packet

- **MY\_STATUS Packet:** This is the packet used to propagate the status of the node itself. The packet contains address of the node which has propagated its own status and the type of itself(Cluster head or gateway).

MY_STATUS Packet	<ul style="list-style-type: none"> <li>– Packet Type</li> <li>– Source Address</li> <li>– Node Type</li> </ul>
------------------	--

Table 4.2: Structure of Status packet

- **MY\_TEMP Packet:** It is a temporary packet sent when any node selects the node other than itself as cluster head. This packet contains its type, address of the node which has generated the packet and address of the node which is made selected as cluster head.

MY_TEMP Packet	<ul style="list-style-type: none"> <li>– Packet Type</li> <li>– Source Address</li> <li>– Destination Address</li> </ul>
----------------	--

Table 4.3: Structure of Temp Packet

- **MY\_DEGREE Packet:** This packet is used to propagate the degree of node among its neighbors. Thus it contains Source Address and its degree.

MY_DEGREE Packet	<ul style="list-style-type: none"> <li>– Packet Type</li> <li>– Source Address</li> <li>– Degree</li> <li>– Serving Factor</li> </ul>
------------------	---

Table 4.4: Structure of Degree Packet

### 4.2.2 Data structure to store One Hop Neighbor information

For each node the list of one hop neighbor is stored in the form of link list. Each node stores the address of its neighbor, node type of neighbor, its degree and link to the next neighbor. The link list is sorted in ascending order.

MY_NT_Node	<ul style="list-style-type: none"> <li>• neighbor address</li> <li>• NodeType of neighbor</li> <li>• Degree of neighbor</li> <li>• Serving Factor of neighbor</li> <li>• Pointer to next neighbor</li> </ul>
------------	--

Table 4.5: Structure of Neighbor information stored at each node

### 4.3 Implementation of algorithm

To implement the algorithm some steps should be required to carry out in the simulator coding.

- Initializing the data structure used.
- Finding out the one hop neighbor by sending nbrRequest packet.
- Creating one hop neighbor table from each node.
- Defining the type of the node(Cluster head , Ordinary node or Gateway)
- Propagate the status into the network.

#### 4.3.1 Initializing the data structure used

Initially all nodes have does not have any information about any other nodes existing in the network. So each node initializes some information.

**Pseudo code for initializing parameters:**

```

InitmyNbrTable()
{
Total1HopNbr <- 0

HeadOfNbrTable <- NULL

SizeOfNbrTbl <- 0
}

```

Any information stored at node level is appended to the api.h structure of glosim. The information stored at layer level (i.e. at network level information) may change per time are added into network layer protocol.h file (here, my.h)

### 4.3.2 Creating one hop neighbor table

For creating the neighbor table, each node starts sending nbrRequest packets. Each node runs a procedure where it sends a nbrRequest for maximum four times.

```

InitiatemynbrRequest()
{
    if nbrreqcount < 4 then
        MY_nbrRequest packet is generated and broadcasted.
    }

```

The average no of neighbors detected during this are shown in Figure 4.1.

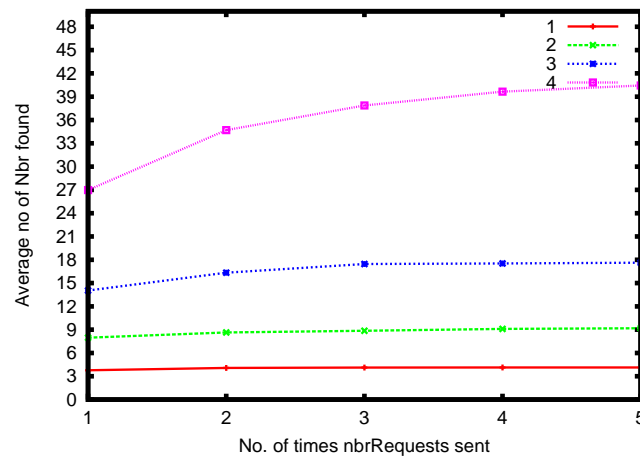


Figure 4.1: Effect of no. of nbrRequest on neighbor detection

As shown in the graph it can be seen that we get more average number of neighbor if the number of times nbrRequest sent is increased. But as the number of packets increase, it increases collision in the network.

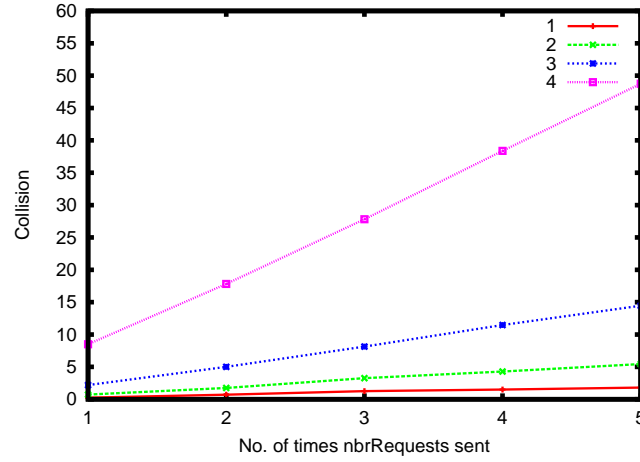


Figure 4.2: Effect of no. of nbrRequest on collision

If we consider the slope between each point for the specific number of nodes in the network then it is also increases and as we increase number of nodes in the network then also individual slope of the collision graph (for different size of the network) increases because of the increase of control packets in the network. So we selected to send the neighbor request four times which also balances the collision and more neighbor can also be found.

After sending the request Each node maintains a link list of its one hop neighbor's node address. So, there is a procedure running at every node which checks that the incoming neighbor request is duplicate or not. If the packet is duplicate, then the node discards it otherwise it inserts into the neighbor table according to ascending order of the nodeID in the list.

Thus there are four procedures running to perform this task and to create one hop neighbor table.

- **HandlemynbrRequest():**

Takes the incoming nbrRequest packet, extracts the sender ID and gives it for checking of duplication.



- **CheckNbrExists():**

The sender's Id extracted from `HandlemynbrRequest`, is compared with already existing node IDs of neighbor table. If the match is found, the procedure returns FALSE otherwise returns TRUE.

- **InsertintonbrTable():** If `ChecknbrExists` procedure returns FALSE it shows that neighbor request is fresh one not duplicate so this function inserts the new neighbor's address into ascending order of node ID into neighbor table.

### 4.3.3 Defining type of the node

As once each node has its neighbor's list it selects which node role it will have in the network.

For the `LowestID`, node checks its own ID with neighbors. If its own ID is less than all neighbors, then it would declare itself as Cluster Head. For that it generates `MY_STATUS` packet and propagates its status into network. Thus the below procedure are used for defining the type and propagating the status.

- **DefineType():**

This algorithm is used to define the status of any node as cluster head.

- **DefineGW():**

This algorithm checks the node is gateway or not. The node whose more than one neighbors are of type cluster head is selected as gateway nodes.

- **PropagateStatus():**

This function propagates the node status among neighbors.

For `MaxDegree`, one more round to send the Degree packet is performed. Then the degree of the node is compared with its neighbor's degree. For sending degree packet below procedures runs at each node.

- **SendDegreePacket():**

This function generates and broadcasts `MY_DEGREE` packet.

- **HandleDegreePacket():**

The degree packets are also sent for four times to maximally sending the information about node's degree to its neighbors. So this function checks for duplication of degree packet and appropriately inserts degree information into neighbor table at corresponding place. For this it uses **InsertmyDegree()** procedure.

#### 4.3.4 Updating cluster structure

As the MANET has dynamic environment and nodes are moving, it is required to update the cluster structure timely. We have applied 1 minute interval to update the cluster structure. So all the procedures are repeated at every one minute interval. Before updating the data all the data structures information stored are deleted using the procedure **FlushAllTable()**.

## Chapter 5

# Analysis of Results And Proposed Approach

Implementing Lowest ID and MaxDegree can be analyzed by the number of clusters created in the network. The simulation parameters are given below.

### 5.1 Results of LowestID & MaxDegree Clustering Algorithm

We have implemented LowestID & MaxDegree clustering algorithm in the Glomosim. Formation of cluster is done according to the parameter selected and to maintain the cluster, the network structure is updated at 1 minute interval.

Simulation parameters are listed in Table 5.1 .

Simulation Area	2000×2000 meters
No. of Nodes	50,100,150,200,250
Node Placement	random
Mobility model	random way point
Mobility min-max Speed	0-5(meter/second)
Simulation Time	10 minutes

Table 5.1: Simulation Parameter

The no of leading nodes found in both the algorithms are shown in Figure 5.1.

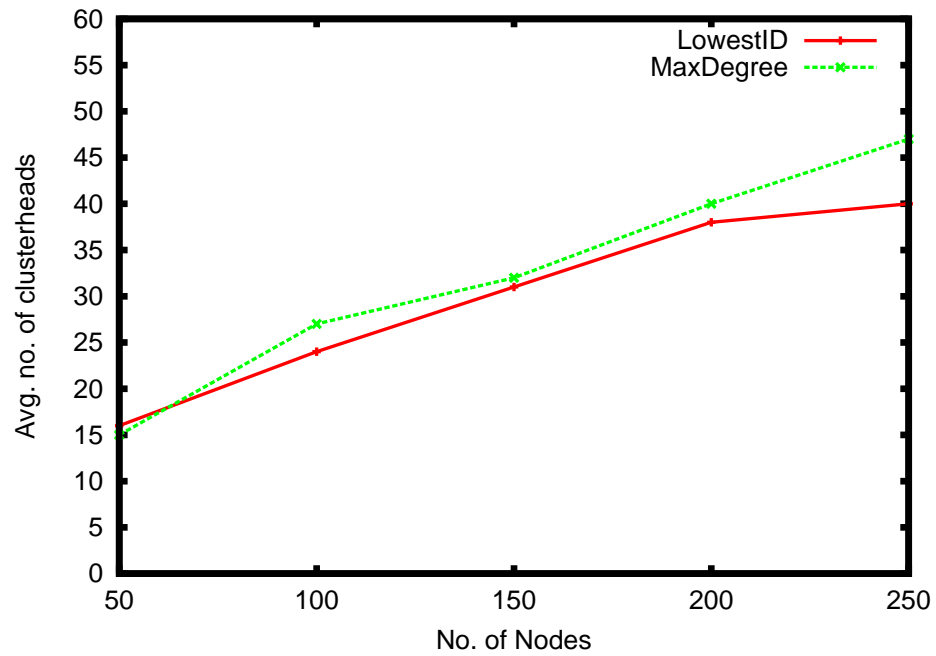


Figure 5.1: Average number of cluster head in different size of network

As shown in Figure approximately 30% nodes are found as the leading nodes in the network. So there can be reduction in the network links accordingly.

## 5.2 Load balancing in LowestID and MaxDegree

Here we have calculated the number of times a node serve as cluster head in the network during the simulation and found the difference between the maximum number which a node has become cluster head and the minimum number for which any node in the network become cluster head.

For the lowest ID the difference is more compared to the MaxDegree as the node having minimum ID moves anywhere in the network, It is selected as the cluster head i.e. leading node. So, energy starvation occurs for the lowest ID node. Similarly in the MaxDegree, the node which moves always in the dense area always becomes a cluster head in the network. Thus In Lowest Id the probability of a node of becoming a cluster head depends on its ID and in MaxDegree it depends on the mobility of the node.

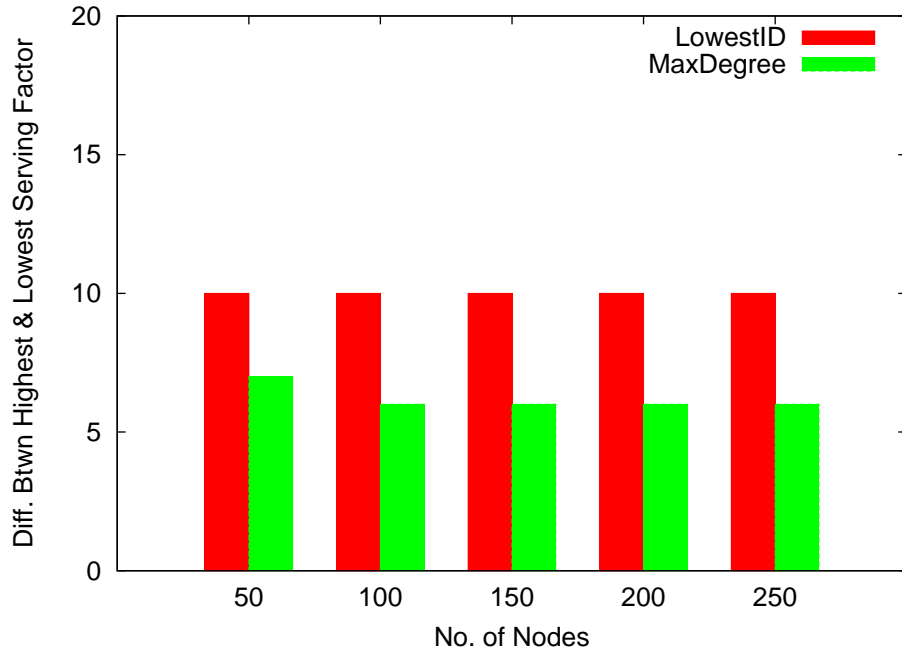


Figure 5.2: Comparison of serving factor in LowestID and MaxDegree

### 5.3 Extension to MaxDegree

We get good load balancing factor in MaxDegree as compared to the LowestID but this still depends on the dynamic positioning of the nodes. So It may be possible that this factor leads to higher values. If the Load balancing can be made on the static factor, it is possible to provide equal chance to each node to become the cluster head.

For this purpose, each node maintains the history about how many times it has served as cluster head during simulation. So when the next time reclustering occurs, the algorithm chooses the node as the cluster head which had minimally served the network as leading node. The results for this algorithm is shown in the Figure 5.3 with comparison to lowestID and MaxDegree.

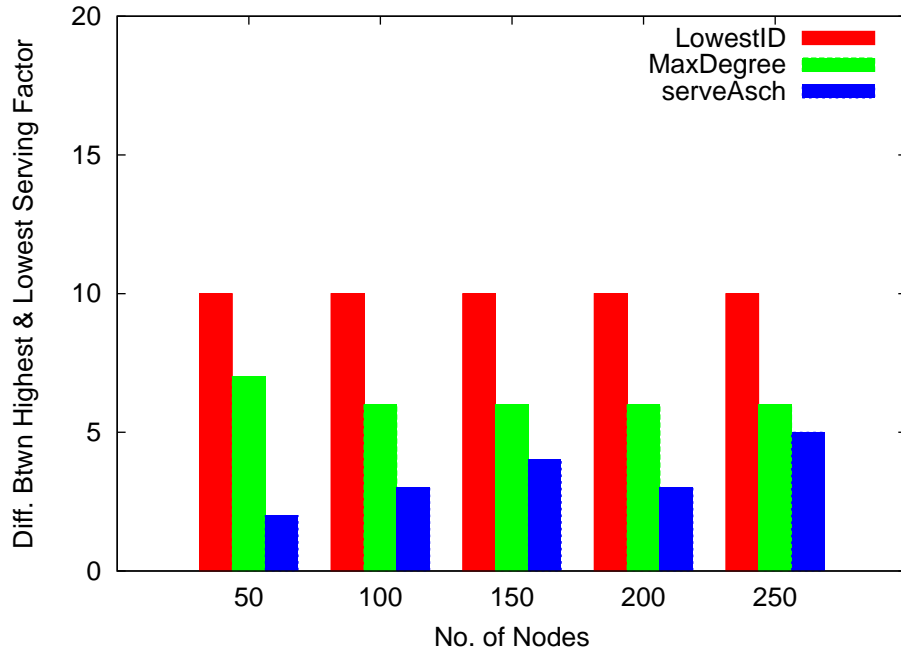


Figure 5.3: Comparison of serving factor

This Extension in the algorithm can be helpful to avoid more energy consumption of any specific node and thus saves energy of the node. The clusterhead observation

of the new algorithm is compared with the lowestID and Maxdegree which shows that it may lead to some more number of leading nodes but also provides the load balancing in the network.

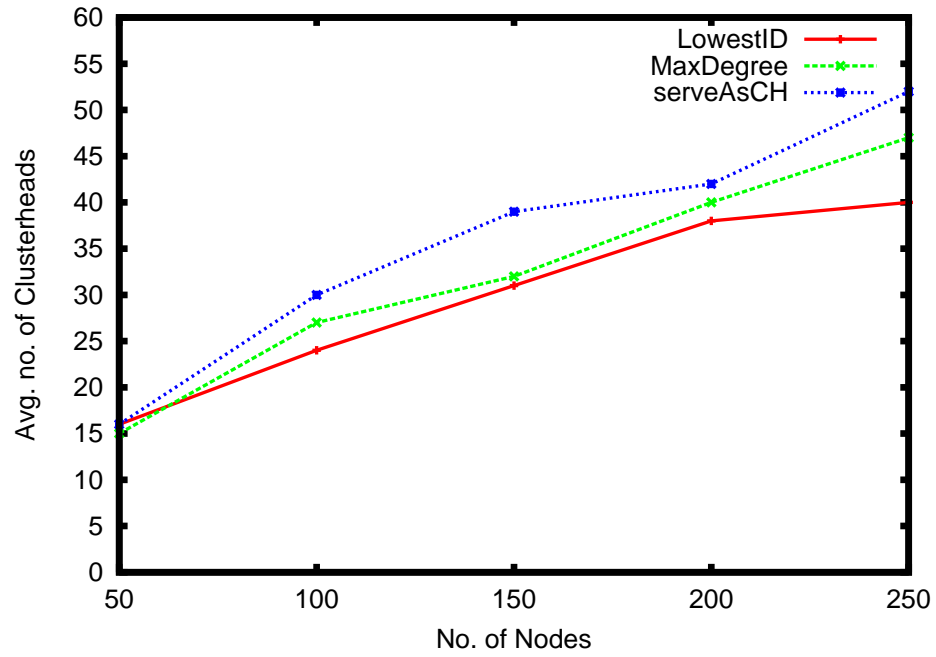


Figure 5.4: Average number of cluster head in different size of network

## Chapter 6

### Conclusion and Future work

The extended Max Degree algorithm provides load balancing and thus also saves the energy of the nodes in the ad hoc network by assigning cluster head role equally to each node in the network. This would preserve the bias ness of becoming cluster head which exists in Lowest ID algorithm and also it makes the decision of taking cluster head role independent of node placement which exists in MaxDegree algorithm.

To maintain the cluster, Algorithm updates the network structure information at specified time interval. The algorithm can be enhanced to provide some more refined parameters to handle mobility. Also the algorithm provides a back bone structure of network by managing the links between nodes so by using this a cluster based routing can also be done.



## Web References

- [1] <http://www.comp.nus.edu.sg/tanhweex/research/GuidetoGloMoSim.pdf>
- [2] <http://www.home.mylkcن.net/gaoanyu/glomosim.html>
- [3] <http://www.cpe.ku.ac.th/anان/courses/204529/document/GlomoSim.pdf>
- [4] <http://pcl.cs.ucla.edu>
- [5] <http://www.icis.ntu.edu.sg/glomosim/how-to-hook-in-glomosim.pdf>

# References

- [1] Damianos Gavalas, Grammati Pantziou, Charalampos Konstantopoulos, Basilis Mamalis, "Lowest-ID with Adaptive ID Reassignment: A Novel Mobile Ad-Hoc Networks Clustering Algorithm", *Wireless Pervasive Computing*, 1st International Symposium, April 2006.
- [2] Geng Chen, Fabian Garcia Nocetti, Julio Solano Gonzalez and Ivan Stojmenovic, "Connectivity Based k-hop Clustering in Wireless Networks", System Sciences HICSS , *Proceedings of the 35th Annual Hawaii International Conference*, pp: 2450-2459, August 2002.
- [3] AlanD.Amis, RaviPrakash, ThaiH.P.Vuong, Dung T.Huynh, "Max-Min D - Cluster Formationin Wireless Ad Hoc Networks", 2000.[Online]Available: cite-seer.ifi.unizh.ch/amis00maxmin.html
- [4] SupengLeng, YanZhang, Hsiao-HwaChen, LirenZhang, KeLiu , "A Novel k-Hop Compound Metric Based Clustering Scheme for AdHoc Wireless Networks", *Wireless Communications*, IEEE Transactions, Vol. 8, pp. 367-375, February 2009.
- [5] Mainak chatterjee, Sajal k.Das, damla turgut," WCA :A Weighted Clustering Algorithm for Mobile AdHoc Networks ",*Cluster Computing*, vol.5, pp.193-204, 2002.
- [6] Jing An, Chang Li, Bin Li ,"A Improved Weight Based Clustering Algorithm In Mobile Ad Hoc Networks", *Information, Computing and Telecommunication*, IEEE Youth Conference, pp. 220-223, 2009.
- [7] Deniz Cokuslu, Kayhan Erciyes, Orhan Dagdeviren, "A Dominating Set Based Clustering Algorithm for Mobile Adhoc Networks", [online]Available: cite-seer.ist.psu.edu/viewdoc/summary?doi=10.1.1.64.3549
- [8] Stefano Basagni," Distributed Clustering For Ad Hoc Networks", *Parallel Architectures, Algorithms, and Networks*, pp. 310-315, 1999.
- [9] rhan Dagdeviren, Kayhan Erciyes and Deniz Cokuslu ," Merging Clustering Algorithms in Mobile Ad hoc Networks", [online]Available: [http://web.iyte.edu.tr/kayhanerciyes/erciyes\\_alg.pdf](http://web.iyte.edu.tr/kayhanerciyes/erciyes_alg.pdf)

- [10] Doug Willard, "Invited Paper: Network Management Architecture for the Objective Airborne Network", *Military Communications Conference*, Vol. 1, pp. 388-393, 2004.
- [11] "Simple Network Management Protocol", *Internetworking Technologies Handbook*
- [12] Compilation by Jorge Nuevo,"A Comprehensible GloMoSim Tutorial",INRS, March 2003.
- [13] LucHogie, Pascal Bouvry & Frederic Guinand, "An Overview of MANETs Simulation", [online]Available: cite-seerx.ist.psu.edu/viewdoc/download?doi=10.1.1.106
- [14] Yvonne Rooney,"Extending the Range of Wireless LANs using AODV",Dublin City University , School Of Electronic Engineering
- [15] Richard A. Meyer , "PARSEC User Manual", UCLA Parallel Computing Laboratory, Release 1.1 ,January 1999.
- [16] Muhammad Azizur Rahman, ,Algirdas Paktas ,Frank Zhigang Wang , "Network Modelling And Simulation Tools", *Simulation Modelling Practiceand Theory*, vol. 17, pp. 1011-1031, 2009
- [17] Xiang Zeng , Rajive Bagrodia , Mario Gerla , "GloMoSim: A Library for Parallel Simulation of Large-scale Wireless Network" , Department of Computer Science University of California, Los Angeles.
- [18] Lokesh Bajaj, Mineo Takai, Rajat Ahuja, Ken Tang, Rajive Bagrodia, Mario Gerla , "GloMoSim: A Scalable Network Simulation Environment" , Computer Science Department University of California, Los Angeles.
- [19] Jian ZHAO , "How to hook your designed routing protocol into GlomoSim".
- [20] Dr.Mehdi Dehghan, Amir Darehshoorzadeh , "Global Mobile Information System Simulator"

# Index

Abstract, v

Acknowledgements, vi

Certificate, iv