

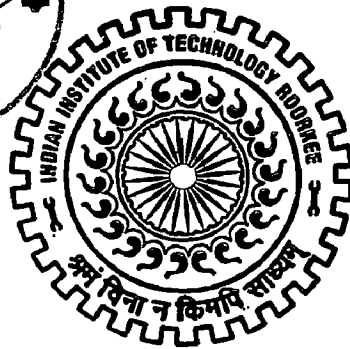
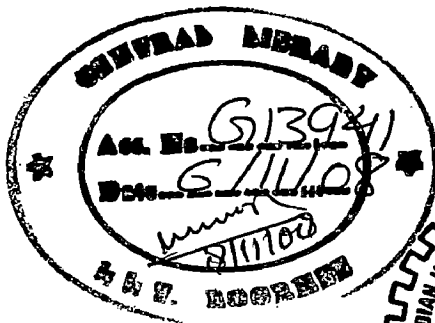
OLSR BASED INTERVAL ADAPTIVE ROUTING WITH EFFICIENT MPR's TO MINIMIZE CONTROL OVERHEAD

A DISSERTATION

*Submitted in partial fulfillment of the
requirements for the award of the degree
of*
MASTER OF TECHNOLOGY
in
COMPUTER SCIENCE AND ENGINEERING

By

ANIL KUMAR GOLLA



**DEPARTMENT OF ELECTRONICS AND COMPUTER ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY ROORKEE
ROORKEE - 247 667 (INDIA)**

JUNE, 2008

CANDIDATE'S DECLARATION

I hereby declare that the work, which is being presented in the dissertation entitled “**OLSR based Interval Adaptive Routing with Efficient MPRs to Minimize Control Overhead**” towards the partial fulfillment of the requirement for the award of the degree of **Master of Technology in Computer Science and Engineering** submitted in the Department of Electronics and Computer Engineering, Indian Institute of Technology Roorkee, Roorkee (India) is an authentic record of my own work carried out during the period from July 2007 to June 2008, under the guidance of **Dr. A. K. Sarje, Professor, Department of Electronics and Computer Engineering, IIT Roorkee.**

I have not submitted the matter embodied in this dissertation for the award of any other degree or diploma.

Date: 20/06/2008

Place: Roorkee

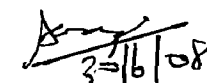

(ANIL KUMAR GOLLA)

CERTIFICATE

This is to certify that the above statement made by the candidate is correct to the best of my knowledge and belief.

Date:

Place: Roorkee


3-16/08
(Dr. A. K. Sarje)

Professor

Department of Electronics and Computer Engineering

IIT Roorkee – 247 667

ACKNOWLEDGEMENTS

I would like to take this opportunity to extend my heartfelt gratitude to my guide and mentor **Dr. A. K. Sarje**, Professor, Department of Electronics and Computer Engineering, Indian Institute of Technology Roorkee, for his trust in my work, his esteemed guidance, regular source of encouragement and assistance throughout this dissertation work. I would state that the dissertation work would not have been in the present shape without his inspirational support and I consider myself fortunate to have done my dissertation under him.

I also extend my sincere thanks to **Dr. D. K. Mehra**, Professor and Head of the Department of Electronics and Computer Engineering, and **Dr. Padam Kumar**, Professor, Department of Electronics and Computer Engineering, and Head, Institute Computer Centre for providing facilities for the work.

Finally, I would like to say that I am indebted to my parents for everything that they have done for me. All of this would have been impossible without their constant support.

ANIL KUMAR GOLLA

Abstract

Mobile Ad-hoc networks are dynamic in nature. The protocols for the MANET's are either proactive and reactive. In the proactive routing focus is particularly on Optimized Link state routing (OLSR) which is widely adopted in Mobile Ad-hoc networks. The Optimized Link State routing protocols produce high routing control overhead. An OLSR based Interval Adaptive Routing with efficient MPRs to minimize control overhead is developed for the mobile Ad-hoc network to minimize the routing overhead by adaptively maximizing the control message broadcasting interval while satisfying its local mobility. The Optimized Link State Routing protocol (OLSR) uses special nodes called Multipoint Relay (MPR) nodes to broadcast control messages within the network. These Multipoint Relay (MPR) nodes act as anchor nodes which diffuse effectively the control messages over the Mobile Ad-hoc Network . If the neighbors of the node in the Mobile Ad-hoc Network change a little then the rate at which control(Hello) packets are diffused via network is reduced there by reducing the control over head. Multipoint relays offer an optimized way of flooding packets in a radio network. However, this technique requires the last hop knowledge to decide whether or not a flooding packet is retransmitted, a node needs to know from which node the packet was received. An algorithm for computing an optimized connected dominating set multipoint relays , OLSR based Interval adaptive routing protocol which reduce the control overhead and the end to end delays are developed. The performance is evaluated by simulation in NS2 and the working of the protocol is compared with routing protocol techniques such as OLSR and AODV. Thus it is showed that the algorithm simulation demonstrate that OLSR based Interval Adaptive Routing with efficient MPRs to minimize control overhead effectively reduces the routing overhead and gains a better performance.

CONTENTS

| | |
|--|------------|
| CANDIDATE'S DECLARATION..... | i |
| ACKNOWLEDGEMENTS..... | ii |
| ABSTRACT..... | iii |
| TABLE OF CONTENTS..... | iv |
| FIGURES..... | vi |
| ACRONYMS..... | vii |
| | |
| CHAPTER 1: INTRODUCTION AND STATEMENT OF THE PROBLEM..... | 1 |
| 1.1 Introduction..... | 1 |
| 1.2 Statement of the Problem..... | 4 |
| 1.3 Motivation | 4 |
| 1.4 Organization of the Dissertation | 4 |
| CHAPTER 2: BACKGROUND AND LITERATURE REVIEW..... | 6 |
| 2.1 Mobile Adhoc Network..... | 6 |
| 2.2 Overview of Routing Protocols in MANET..... | 8 |
| 2.3 Link State Routing | 9 |
| 2.4 Optimized Link State Routing Functioning | 11 |
| 2.4.1 Multi Point Relays..... | 12 |
| 2.4.2 MPR Set Computation..... | 14 |
| 2.4.3 Packet format and forwarding..... | 17 |
| 2.4.4 Message Header..... | 18 |
| 2.4.5 Processing & forwarding..... | 20 |
| 2.4.6 Information Repositories..... | 21 |
| 2.4.6.1 Neighborhood Information Base | 21 |
| 2.4.6.2 Topology Information Base | 22 |
| 2.5 RESEARCH GAPS | 24 |

| | |
|--|-----------|
| CHAPTER 3: PROPOSED STRATEGY TO REDUCE CONTROL OVERHEAD..... | 26 |
| 3.1 Control over head in Link State Routing..... | 26 |
| 3.2 Proposed OLSR based Interval Adaptive Routing with Efficient MPR's..... | 28 |
| | |
| CHAPTER 4: SYSTEM DESIGN AND IMPLEMENTATION..... | 33 |
| 4.1 System Design components..... | 33 |
| 4.1.1 Network Interfaces | 34 |
| 4.1.2 Outgoing Packets..... | 36 |
| 4.1.3 Incoming Packets..... | 37 |
| 4.1.4 Adding user level agents to a Mobile Node | 38 |
| 4.2 Implementation..... | 38 |
| 4.2.1 Computing of the MPR set..... | 38 |
| 4.2.2 Adjustments in Hello and Topology Timers..... | 39 |
| 4.2.3 Packet Processing and forwarding | 41 |
| 4.3 Simulation Parameters..... | 44 |
| 4.4 Trace | 44 |
| | |
| CHAPTER 5: RESULTS AND DISCUSSION..... | 49 |
| 5.1 Results..... | 49 |
| | |
| CHAPTER 6: CONCLUSIONS AND FUTURE WORK..... | 61 |
| 6.1 Conclusions..... | 61 |
| 6.2 Suggestions for Future Work..... | 61 |
| | |
| REFERENCES..... | 62 |

LIST OF FIGURES

| Fig.No. | Name | Page No. |
|----------------|--|-----------------|
| 1.1 | Mobile Nodes with their communication ranges in MANET | 3 |
| 2.1 | Wireless Networks | 6 |
| 2.2 | Simple Adhoc Network with Three Mobile Nodes | 7 |
| 2.3 | Classification of Adhoc Routing protocols | 8 |
| 2.4(a) | Pure Flooding Technique | 10 |
| 2.4(b) | Using MPR Technique | 10 |
| 2.5 | A Mobile Adhoc Network | 15 |
| 2.6 | Multipoint Relays for the Nodes of a Mobile Adhoc Network | 16 |
| 3.1 | Dynamic and Static Region in a Mobile Ad-Hoc Network | 26 |
| 4.1 | Logical view of Mobile Nodes connecting to channel | 33 |
| 4.2 | Mobile Node Interfaces | 35 |
| 5.1 | Mobility vs Control Overhead | 49 |
| 5.2 | Mobility vs End to End delay | 50 |
| 5.3 | Packet id vs End to End delay | 51 |
| 5.4 | Packet Receive Time vs No. of Intermediate Nodes | 52 |
| 5.5 | Packet ID vs No. of Intermediate Nodes receiving packets | 53 |
| 5.6 | Packet Receive Time at destination Nodes vs End to End Delay | 54 |
| 5.7 | Packet send time at source node vs End to End delay | 55 |
| 5.8 | Packet Drop Time vs Sequence Number of Dropped packet | 56 |
| 5.9 | Packet Send time at source node vs No. of Intermediate Nodes.(rec) | 57 |
| 5.10 | Packet Send time at source node vs No. of Intermediate Nodes(rec) | 58 |
| 5.11 | Simulation Time vs Throughput of Generating packets | 59 |
| 5.12 | Simulation Time vs Throughput of Receiving packets | 59 |
| 5.13 | Simulation Time vs Throughput of Sending packets | 60 |

ACRONYMS

| | |
|-------|--|
| MANET | Mobile Adhoc Network |
| OLSR | Optimized Link State Routing |
| LSR | Link State Routing |
| N1 | One Hop Neighbor |
| N2 | Two Hop Neighbor |
| MPR | Multi Point Relay |
| CDS | Connected Dominating Set |
| DS | Dominating Set |
| AODV | Adhoc On-Demand Distance Vector Routing Protocol |
| FSR | Fish Eye State Routing |
| ZRP | Zone Routing Protocol |
| IETF | Internet Engineering Task Force |
| ID | Internet Draft |
| RFC | Request for Comments |
| IANA | Internet Assigned Numbers Authority |
| MID | Multiple Interface Declaration |
| LL | Link Layer |
| TTL | Time to Live |
| NS2 | Network Simulator 2 |

CHAPTER 1

INTRODUCTION AND STATEMENT OF THE PROBLEM

1.1 Introduction

Mobile Ad Hoc networks (MANETs)[1] gained much attention in recent years due to their self-organizing and infrastructure less characteristics. Each node in a MANET can act as a router/host to receive and forward packets, following seamless communications between devices. Hence, MANETs have great application potential in various scenarios such as battle field communications, emergency services, disaster recovery, environmental monitoring, personal entertainment and mobile conferencing . Broadcast is an important data transmission method used in MANETs to diffuse data and control messages. The goal of the efficient routing protocol is to maximize the node reachability in the network while keeping the computation and communication overheads to minimum[2]. However, the wireless nature of MANETs make it difficult to design an efficient routing protocol algorithm.

Every message sent by a node can be heard by all its adjacent nodes. In the efficient routing protocol algorithm only a subset of nodes in the network are needed to relay a broadcast message. Furthermore, in mobile ad hoc network nodes can randomly move around, leave the network or switch off, and new nodes may join the network unexpectedly. These characteristics cause the network topology to change frequently. Therefore, a routing algorithm requires the global information of a network which may be unstable and complex[1].

An approach for doing efficient broadcast is to choose a small subset of all available nodes, called connected dominating set (CDS)[2] in the network based on local information such as the local topology. A dominating set (DS)[2] is a subset of nodes in the network where every node in the network is either a member of the subset, or a

neighbor of at least one member of the subset. A DS is called a CDS if all nodes in the DS are connected. Only nodes in a CDS can retransmit packets while other nodes can only receive them. In this approach, a node determines the forwarding state of itself or its neighbors only based on its neighborhood information. A CDS constructed in such a localized manner can adapt to frequent topology changes, thus ensuring the property of stability.

The Multipoint Relays (MPR)[3] concept is a distributed localized broadcast algorithm, which is computationally lightweight. In this algorithm, each node collects the two-hop neighborhood information (such as node ID) from its one hop neighbors and determines the forwarding state of its one-hop neighbors based on this information.

Specifically, a node knows its one-hop neighbors and neighbors of these one-hop neighbors after it collects the two-hop neighborhood information. Then it selects a subset of nodes from its one-hop neighbors as the forwarding nodes to cover all its two-hop neighbors.

The original MPR algorithm is source dependent, that is, an MPR[2][3] will only forward a packet that comes from its selectors (nodes that select it as an MPR). A selector and its MPRs form a local CDS to cover nodes within two hops away from the selector, and eventually, all nodes in the network are covered by a number of local CDSs. This rebroadcast process requires the last hop knowledge to indicate the source of a packet, which increases the complexity of the algorithm.

To solve this problem, an efficient MPR algorithm is proposed, which computes the CDS and redundant MPR set.

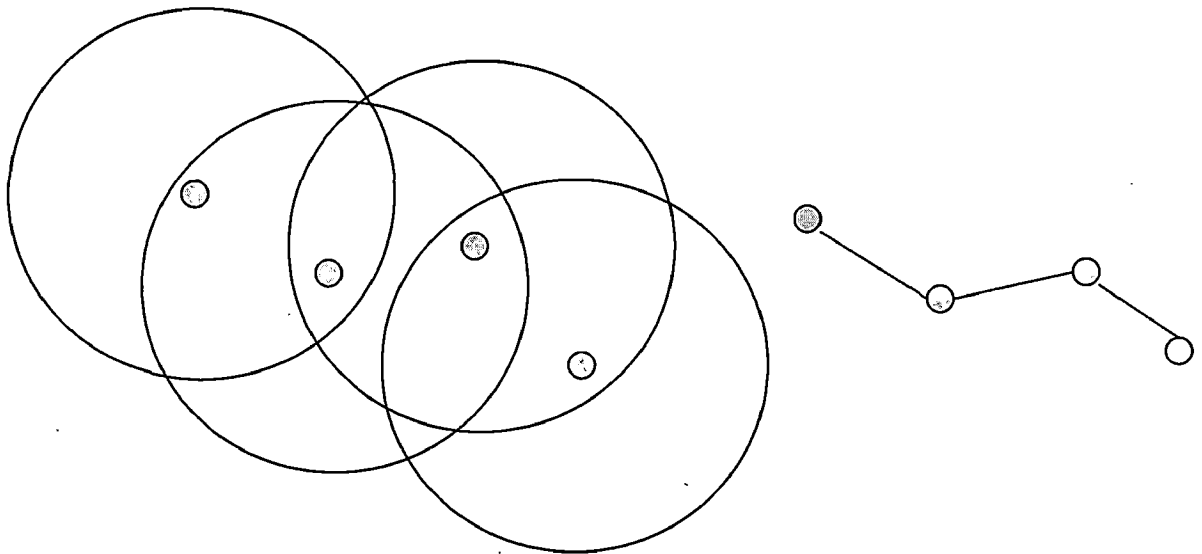


Figure 1.1 Mobile Nodes with their Communication Ranges in MANET.

Multi-hop wireless ad-hoc networks can be quickly deployed and nodes adapt themselves to network topology changes. Such a network does not have base stations unlike other types of wireless networks (e.g.; cellular networks,...).

The nodes organize themselves to route packets in a multihop fashion from a source to a destination as shown in the Figure 1.1. One of the best known Link State ad-hoc routing protocols is OLSR[3].

There different ways to reduce the control overhead in the Link State Routing protocols are

- (1) If Topology change little then TC (Topology Control) Interval is prolonged.
- (2) If Neighbors change little the Hello_Interval is prolonged

In this dissertation, An OLSR based Interval Adaptive Routing with efficient MPRs Routing protocol for MANET's that provides for reduction of the control overhead is proposed and validated.

1.2 Statement of the Problem

1. To formulate an OLSR based Interval Adaptive Routing with efficient MPR's Routing protocol for MANET's.
 - (i) Efficient throughput in MANET Routing protocols,
 - (ii) Reduction in the control overhead by adjusting the frequency at which Hello and Topology control packets are diffused in the MANET.
 - (iii) To study the end to end delays .

1.3 Motivation

In view of the tremendous growth in the wireless networks the routing protocols are designed to maintain network connectivity, throughput , robustness and minimal end to end delay. In order to maintain routes at any instant proactive routing protocols are suitable. These proactive or link state routing protocols compute the route table and have the route from source to destination at any instant. OLSR is a pure link state protocol which optimizes the control overhead by the use of Multi Point relays.

Control overhead is high in OLSR due to transmission of Hello messages for neighbor detection and topology control messages for routing table calculations. In dense wireless networks with mobility the Control overhead is very high. So, reduction of control overhead is necessary as the mobile nodes have constraint's such as limited power, band width etc., . Thus it is necessary to reduce the control overhead in OLSR.

1.4 Organization of the Dissertation

This report comprises of including this chapter that introduces the topic and states the problem. The rest of the dissertation report is organized as follows.

Chapter 2 Gives an overview of the MANET Routing protocols classifies them according to the various criteria. Also, taxonomy of the existing MANET Routing protocols is discussed in brief. It discusses the related work based on the Optimized Link state Routing protocol and research gaps .

Chapter 3 Gives an overview of the proposed OLSR based Interval Adaptive Routing with efficient MPR' s Routing protocol to minimize the control overhead in the link state Routing.

Chapter 4 Explains in detail the System Design and Implementation of the OLSR based Interval Adaptive Routing with efficient MPR' s to minimize control overhead.

Chapter 5 Explains about the results and discussion.

Chapter 6 Conclusions and the future work.

CHAPTER 2

BACKGROUND AND LITERATURE REVIEW

2.1 Mobile Ad hoc Network

Mobile Ad hoc networks are wireless, mobile networks that can be set up anywhere and anytime without the aid of any established infrastructure or centralized administration[1]. It is an autonomous system in which mobile hosts connected by wireless links are free to move randomly and often act as routers at the same time. The traffic types in ad hoc networks are different from those in the networks that have infrastructure. Basic traffic types in ad-hoc networks are as follows:

Peer-to-Peer: Communication between two nodes which are within one hop[1].

Remote-to-Remote: Communication between two nodes beyond a single hop but which maintain a stable route between them. This may be the result of several nodes staying within communication range of each other in a single area or possibly moving as a group.

Dynamic in Nature: This occurs when nodes are dynamic and moving around. Routes are reconstructed frequently. This results in a poor connectivity and network activity in short bursts.

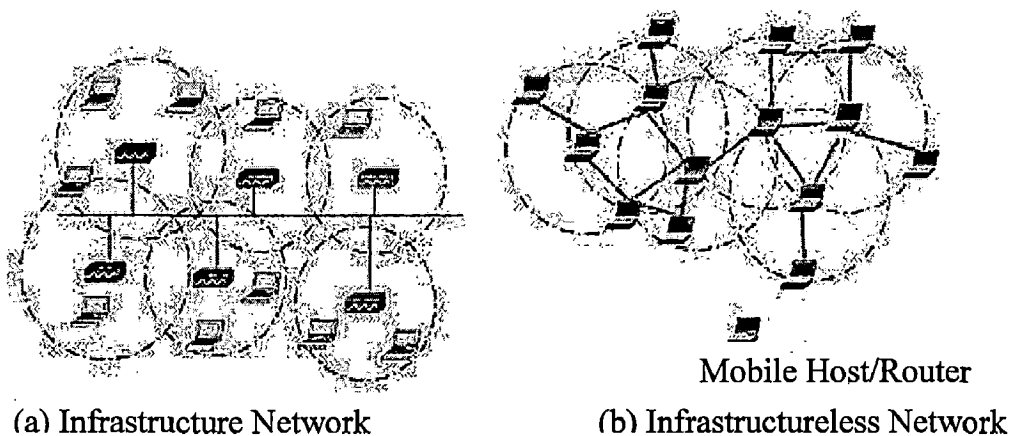


Figure 2.1 Wireless Networks

Because of the limited range of each host's wireless transmission, to communicate with hosts outside its transmission range, a host needs to support the aid of its nearby hosts in

forwarding packets to the destination. However, since there is no established infrastructure such as base stations, each host has to act as a router for itself. A routing protocol for ad hoc networks is executed on every host and is therefore subject to the limit of the resources and more prone to errors at each mobile host.

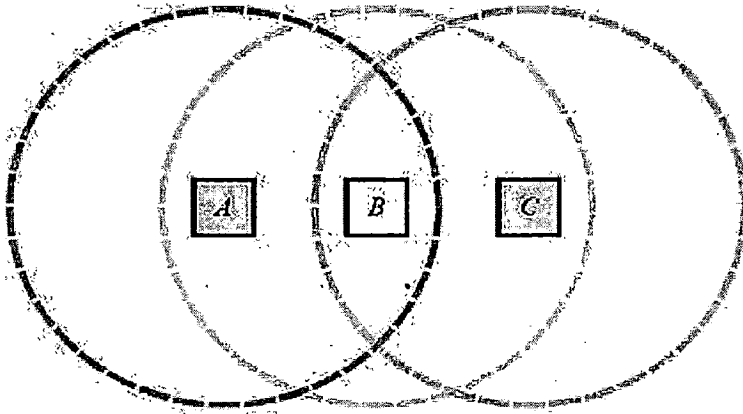


Figure 2.2 : Simple Adhoc Network with three mobile nodes A,B and C

In the MANET as shown in the Figure 2.2 , A and B nodes are in communication range with each other. B and C are with in the range. In order for A to communicate with the C it establishes a route via B.

A good routing protocol should minimize the computing load on the host as well as the traffic overhead on the network[4]. Therefore, a number of routing protocols are proposed for mobile ad hoc networks, by the Internet Engineering Task Force (IETF) Mobile Ad hoc Network (MANET) working group. These routing protocols can be classified into three main categories: the proactive, reactive, and hybrid protocols[1]. Proactive routing[3] attempts to maintain optimal routes to all destinations at all times, regardless of whether they are needed. To support this, the routing protocol propagates information updates about a network's topology throughout the network. In contrast, reactive or on-demand routing protocols determine routes to given destinations only when there is data to send to destinations. If a route is unknown, the source node initiates

a search to find one. There are also a few hybrid protocols, such as ZRP[5] that combine proactive and reactive routing strategies.

2.2 Overview of Routing Protocols in MANET.

The MANET Routing protocols can be classified as shown in the figure 2.3

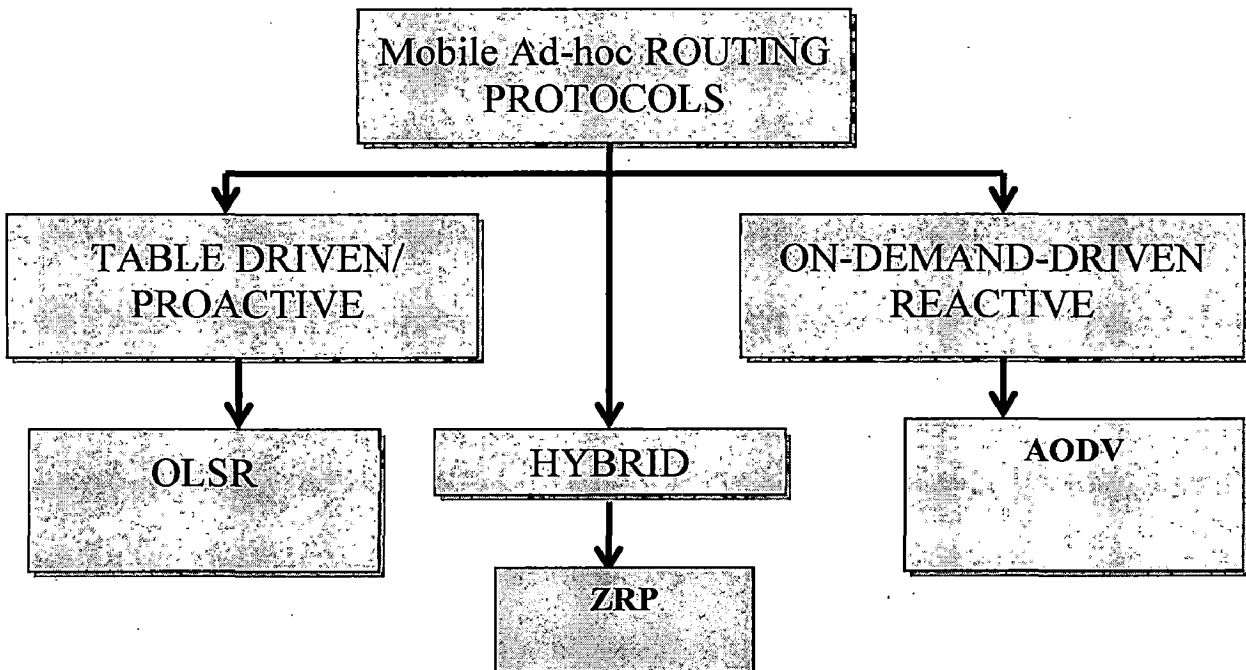


Figure 2.3: Classification of Ad-hoc Routing protocols

Reactive Routing Protocols

The Reactive routing protocols are designed to overcome the maintenance of unused routes. Routing information is acquired only when there is a need for it. The needed routes are calculated on demand[1] (AODV[6]). This saves the overhead of maintaining unused routes at each node, but on the other hand the latency for sending data packets will considerably increase. A long delay can arise before data transmission because it has to wait until a route to the destination is acquired. As reactive routing

protocols flood the network to discover the route, they are not optimal in terms of bandwidth utilization, but they scale well in the frequency of topology change.

Proactive Routing Protocols

The nodes maintain a table of routes to every destination in the network, for this reason they periodically exchange messages (FSR[7]; OLSR). At all times the routes to all destinations are ready to use and as a consequence initial delays before sending data are small. Keeping routes to all destinations up-to-date even if they are not used, is a disadvantage with regard to the usage of bandwidth and of network resources. It is also possible that the control traffic delays data packets, because queues are filled with control packets and there are more packet collisions due to more network traffic.

Hybrid Routing protocols

The Hybrid Routing protocols[1][5] adopt the characteristics of both the Reactive and Proactive Routing Protocols.

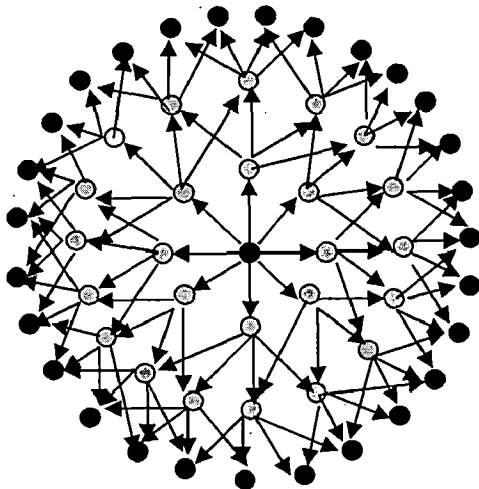
2.3 Link State Routing

The link-sensing is performed by every *node* in the network (i.e. nodes which are prepared to forward packets). The basic concept of link-state routing is that every node constructs a topology of the network[3], showing which nodes are connected to which other nodes. Each node independently calculates the best *next hop* from it for every possible destination in the network.

The Optimized Link State Routing (OLSR) is a table-driven, proactive routing protocol[1] developed for MANETs. It is an optimization of pure link state protocols that reduces the size of control packets as well as the number of control packet transmissions required. OLSR reduces the control traffic overhead by using Multipoint Relays (MPR)[2][3], which is the key idea behind OLSR. An MPR is a node's one-hop neighbor

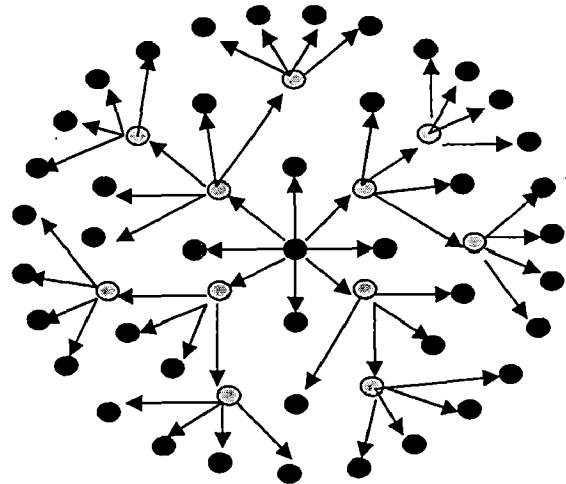
which has been chosen to forward packets. Instead of pure flooding of the network, packets are forwarded by a node's MPRs. This delimits the network overhead, thus being more efficient than pure link state routing protocols.

OLSR is well suited to large and dense mobile networks. Because of the use of MPRs, the larger and more dense a network, the more optimized link state routing is achieved. MPRs help providing the shortest path to a destination. The only requirement is that all MPRs declare the link information for their MPR selectors (i.e., the nodes which have chosen them as MPRs). The network topology information is maintained by periodically exchange link state information. If more reactivity to topological changes is required, the time interval for exchanging of link state information can be reduced[4].



24 retransmissions to diffuse a message up to 3 hops

⊗ Retransmission



11 retransmission to diffuse a message up to 3 hops

⊗ Retransmission node - MPR

Figure 2.4(a) Pure Flooding Technique

Figure 2.4(b) Using MPR Technique

The Main address , Neighbor node, 2-hop neighbor node ,MPR and MPR selector in OLSR are defined as follows[2]

Main address: The main address of a node, which is used in OLSR control traffic as the “originator address” of all messages emitted by this node.

Neighbor node: A node ‘X’ is a neighbor node of node ‘Y’ if node ‘Y’ can hear node ‘X’ . If ‘X’ is a neighbor of ‘Y’ then a link exists between an OLSR interface on node ‘X’ and an OLSR interface on ‘Y’.

2-hop neighbor: A node heard by a neighbor.

Multipoint relay (MPR): A node which is selected by its 1-hop neighbor, node ‘X’, to "re-transmit" all the broadcast messages that it receives from ‘X’, provided that the message is not a duplicate, and that the time to live field of the message is greater than one.

Multipoint relay selector : A node which has selected its 1-hop neighbor, node ‘X’, as its multipoint relay, will be called a multipoint relay selector of node ‘X’.

2.4 Optimized Link State Routing Functioning

OLSR is a proactive routing protocol for mobile ad hoc networks. The protocol inherits the stability of a link state algorithm and has the advantage of having routes immediately available when needed due to its proactive nature. OLSR minimizes the overhead from flooding of control traffic by using only selected nodes, called MPRs, to retransmit control messages. This technique significantly reduces the number of retransmissions required to flood a message to all nodes in the network.

OLSR protocol optimize the reactivity to topological changes by reducing the maximum time interval for periodic control message transmission. Furthermore, as OLSR

continuously maintains routes to all destinations in the network, the protocol is beneficial for traffic patterns where a large subset of nodes are communicating with another large subset of nodes, and where the source and destination pairs are changing over time[4]. The protocol is particularly suited for large and dense networks, as the optimization done using MPRs. The larger and more dense a network, the more optimization can be achieved. OLSR works in a completely in distributed manner and does not depend on any central entity. The protocol does not require reliable transmission of control messages each node sends control messages periodically, and can therefore sustain a reasonable loss of such messages. Such losses occur frequently in radio networks due to collisions or other transmission problems. Also, OLSR does not require sequenced delivery of messages. Each control message contains a sequence number which is incremented for each message. Thus the recipient of a control message can, if required, easily identify which information is more recent - even if messages have been re-ordered while in transmission.

2.4.1 Multipoint Relays

The simplest way of broadcasting a packet to all nodes in an ad hoc network is basic flooding, call pure flooding[2].

Pure flooding forwarding rule: a node retransmits if it has not already received the packet. Clearly all nodes reachable from the source will receive the packet. However, every node will retransmit the packet when it is possible[2], to greatly reduce the number of retransmissions. Several techniques have been proposed to optimize flooding with two hop neighborhood knowledge (*i.e.* a node knows its neighbors and the neighbors of its neighbors). Link state routing protocols usually provide this information. It is usually obtained by regularly emitting Hello packets containing lists of neighbors. Based on graph theory, the set of nodes that will retransmit a given broadcast packet (including the source) must form a connected dominating set. A dominating set is a set of nodes such that any node in the network is neighbor of some element of the set. It is connected if the subgraph formed by this set is connected. The connectedness of the dominating set

insures that all nodes of the connected dominating set will receive the packet and will thus be able to retransmit it.

The dominating set property insures that all nodes will receive the packet (assuming no transmission error).

The selection of the connected dominating set must be distributed. Based on two hop neighborhood knowledge, a node has to decide whether or not it is in the dominating set.

If a connected dominating set has been elected, then the forwarding rule becomes

Connected dominating set forwarding rule : a node retransmits if it has not already received the packet and it is in the connected dominating set. On the other hand, the multipoint relay technique has been proposed to optimize flooding when the last hop information is given. The idea behind this technique is to compute some kind of local dominating sets.

Each node computes a multipoint relay set with the following properties:

- The multipoint relay set is included in the neighborhood of the node, the element of the multipoint relay set are called *multipoint relays* (or MPR for short) of the node,
- Each two hop neighbor of the node has a neighbor in the multipoint relay set.

The multipoint relay set and the node forms a dominating set of the two hop neighborhood of the node. More formally, let $N(E)$ denote the set of all nodes that are in a given set E or have a neighbor in E . E covers a set F when $F \subset N(E)$. Let $N_1(E) = N(E) - E$ denote the nodes at distance 1 from E and $N_2(E) = N(N(E)) - N(E)$ denote the nodes at distance 2 from E . When $E = \{x\}$ contains only one node, $N_1(\{x\})$ is the neighborhood of x and $N_2(\{x\})$ is the two hop neighborhood of x . A MPR set M of a

node x is thus any subset of $N_1(\{x\})$ such that $N_2(\{X\}) \subset N(M)$. M is a subset of neighbors that covers the two hop neighborhood of x or equivalently M which is a dominating set of the subgraph induced by $N(N(\{x\}))$. Multipoint relay selector of a node x a node which has selected node x as multipoint relay. The following forwarding rule allows to reach all the nodes in the network .

Multipoint relay flooding forwarding rule: A node retransmits if it has not already received the packet and it is a multipoint relay of the last emitter[2]. The smallest the multipoint relay sets are, the fewer retransmissions will occur. For computing a multipoint relay set with minimum size , there exists good heuristics based on preferring neighbors with large degree as multipoint relays. The only knowledge assumed for a given node is two hop neighborhood and the list of neighbors that have selected the node as multipoint relay (such neighbors are called multipoint relay selectors). The MPR relay algorithm does not need any distributed knowledge of the global network topology. The MPR algorithm for mobile ad hoc networks needs just local updates at each detected topology change.

2.4.2 MPR set computation

In practice the following greedy algorithm [3] works very well for computing the MPR set of a node x . If v is a two hop neighbor of x , connecting neighbors are the nodes that are connected to both x and v . An empty set is assigned as MPR set.

- Step 1: Find the two-hop neighbors that have only one connecting neighbor. Put in the MPR set the connecting neighbors in $N(x)$ that connect u to these two-hop neighbors. (Notice that any MPR set must contain these connecting neighbors.)
- Step 2: Add in the MPR set the neighbor node that covers the largest number of two-hop neighbors of x that are not yet covered by the current MPR set.

- Repeat Step 2 until all two-hop neighborhood is covered. Algorithm provides an MPR set whose size is at most $\log m$ larger than the optimal MPR set, where m is the maximum degree of a node.

In OLSR each node learns its 2-hop neighborhood by sending HELLO messages. Each message contains the list of known neighbors.

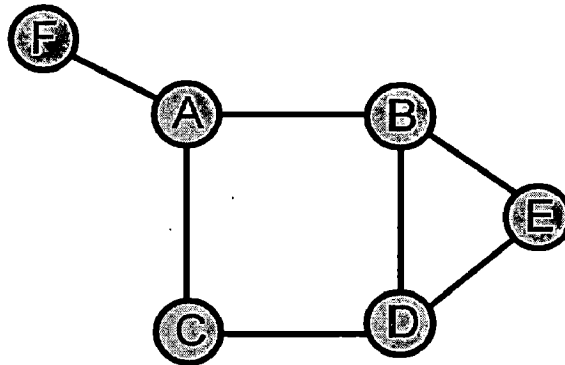


Figure 2.5 A Mobile Ad-hoc Network

- A sends empty HELLO_A{ },
- A receives HELLO_B{A},
- A receives HELLO_F{A}.
- A receives HELLO_C{A},
- A sends HELLO_A{B,C,F}, thus F learns 2 hop neighbors: B,C.

In this manner all nodes learn their 2-hop neighborhood, and also have a partial view of the 2-hop topology.

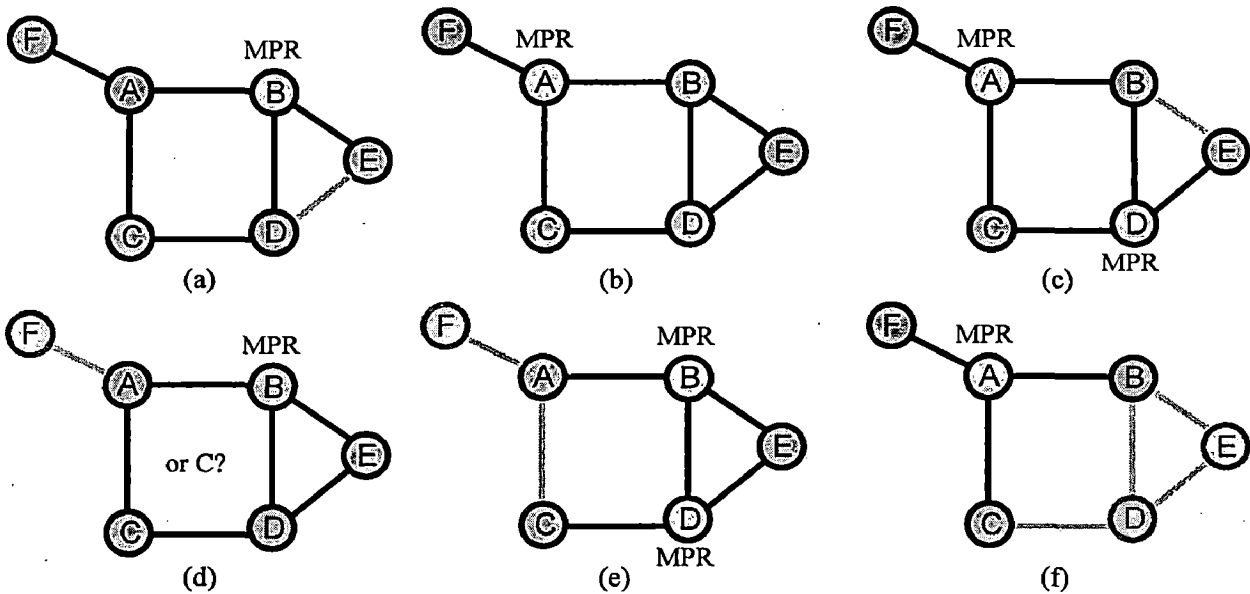


Figure 2.6 MPRs for the Nodes of the Mobile Ad-hoc Network

The multipoint relays minimize the overhead of flooding. Messages in the network by reducing redundant retransmissions in the same region. Each node in the network selects a set of nodes in its symmetric 1-hop neighborhood which may retransmit its messages. This set of selected neighbor nodes is called the "Multipoint Relay" (MPR) set of that node. The neighbors of node N which are not in its MPR set, receive and process broadcast messages but do not retransmit broadcast messages received from node N. Each node selects its MPR set from among its 1-hop symmetric neighbors. This set is selected such that it covers (in terms of radio range) all symmetric strict 2-hop nodes.

The MPRs of A,B,C,D,E and F are B,A,{A and B},{B or C},{B and D},and A respectively from figure 2.6(a),(b),(c),(d),(e) and (f) respectively.

From the figure 2.7 only nodes A,B,D are selected as MPR. But not all MPR retransmit a flooded packet a node only re-transmits a packet if it is an MPR of the last emitter.

A flooding by node F requires 3 packets = optimal case.

A flooding by node E requires 4 packets actually 5 if node C was chosen as MPR of node D. Hence performance heavily depends on MPR selection

The MPR set of N , denoted as $MPR(N)$, is then an arbitrary subset of the symmetric 1-hop neighborhood of N which satisfies the following condition every node in the symmetric strict 2-hop neighborhood of N must have a symmetric link towards $MPR(N)$. The smaller a MPR set, the less control traffic overhead results from the routing protocol. Each node maintains information about the set of neighbors that have selected it as MPR. This set is called the "Multipoint Relay Selector set" (MPR selector set) of a node. A node obtains this information from periodic HELLO messages received from the neighbors.

A broadcast message, intended to be diffused in the whole network, coming from any of the MPR selectors of node N is assumed to be retransmitted by node N , if N has not received it yet. This set can change over time (i.e., when a node selects another MPR-set) and is indicated by the selector nodes in their HELLO messages.

The core functionality of OLSR specifies the behavior of a node, equipped with OLSR interfaces participating in the MANET and running OLSR as routing protocol. This includes a universal specification of OLSR protocol messages and their transmission through the network, as well as link sensing, topology diffusion and route calculation.

2.4.3 Packet Format

- **Protocol and Port Number**

Packets in OLSR communicate using UDP. Port 698 has been assigned by IANA for exclusive usage by the OLSR protocol[3]

- **Link Sensing**

Link Sensing[3] is accomplished through periodic emission of HELLO messages over the interfaces through which connectivity is checked. A separate HELLO message is generated for each interface. Resulting from Link Sensing is a local link set[3][4][9], describing links between "local interfaces" and "remote interfaces" - i.e., interfaces on

neighbor nodes. If sufficient information is provided by the link-layer, this is utilized to populate the local link set instead of HELLO message[3] exchange.

- **Neighbor detection**

Given a network with only single interface nodes, a node may deduct the neighbor set[4][9] directly from the information exchanged as part of link sensing the "main address" of a single interface node is the address of the only interface on that node.

In a network with multiple interface nodes, additional information is required in order to map interface addresses to main addresses and to the nodes. This additional information is acquired through multiple interface declaration (MID) messages.

- **MPR Selection and MPR Signaling**

The MPR set of a node is computed such that it, for each interface, The MPR selection for a node is to select a subset of its neighbors such that a broadcast message, retransmitted by these selected neighbors, will be received by all nodes 2 hops away.

The information required to perform this calculation is acquired through the periodic exchange of HELLO messages.

2.4.4 Message Header

The Message Header Field[3] consist of the following attributes

- **Message Type**

This field indicates which type of message is to be found in the MESSAGE part. Message types in the range of 0-127 are reserved for messages (Eg., Hello Messages, TC etc.,)

- **Vtime**

The field indicates for how long time after reception a node MUST consider the information contained in the message as valid, unless a more recent update to the information is received. The validity time is represented by its mantissa (four highest bits of Vtime field) and by its exponent (four lowest bits of Vtime field). In other words:

$$\text{validity time} = C * (1 + a/16) 2^b \text{ [in seconds]}$$

where a is the integer represented by the four highest bits of Vtime field and b the integer represented by the four lowest bits of Vtime field and C is the scaling factor.

- **Message Size**

Size of this message, counted in bytes and measured from the beginning of the "Message Type" field and until the beginning of the next "Message Type" field (or - if there are no following messages - until the end of the packet).

- **Originator Address**

Contains the main address of the node, which has originally generated this message. The Originator Address field is not changed in retransmissions.

- **Time To Live**

This field contains the maximum number of hops a message will be transmitted. Before a message is retransmitted, the Time To Live is decremented by 1. A Node receiving a message with a Time To Live equal to 0 or 1, the message is not retransmitted. Normally, a node would not receive a message with a TTL of zero. Thus, by setting the field, the originator of a message can limit the flooding radius.

- **Hop Count**

This field contains the number of hops a message has attained. Before a message is retransmitted, the Hop Count must be incremented by 1. Initially, the Hop count is set to '0' by the originator of the message.

- **Message Sequence Number**

The "originator" node will assign a unique identification number to each message. This number is inserted into the Sequence Number field of the message. The sequence number is increased by 1 (one) for each message originating from the node. Message sequence numbers are used to ensure that a given message is not retransmitted more than once by any node.

2.4.5 Processing and forwarding

The processing and forwarding messages are two different actions, conditioned by different rules. Processing relates to using the content of the messages, while forwarding is related to retransmitting the same message for other nodes of the network. Messages with known message types are not forwarded "blindly" by MPR algorithm. Forwarding (and setting the correct message header in the forwarded, known, message) is the responsibility of the algorithm specifying how the message is to be handled and, if necessary, retransmitted. Thus enables a message type to be specified such that the message can be modified while in transit (e.g., to reflect the route the message has taken). It also enables bypassing of the MPR flooding mechanism[10][11] if for some reason classical flooding of a message type is required, the algorithm which specifies how such messages should be handled will simply rebroadcast the message, regardless of MPRs. The REQUIRED message types for the core functionality of OLSR are

- HELLO-messages, performing the task of link sensing, neighbor detection and MPR signaling,
- TC-messages, performing the task of topology declaration (advertisement of link states).
- MID-messages, performing the task of declaring the presence of multiple interfaces on a node.

2.4.6 Information Repositories

Through the exchange of OLSR control messages, each node accumulates information about the network, which is stored as follows

2.4.6.1 Neighborhood Information Base

The neighborhood information base stores information about neighbors, 2-hop neighbors, MPRs and MPR selectors.

Neighbor Set

A node records a set of "neighbor tuples" ($N_neighbor_main_addr$, N_status , $N_willingness$), describing neighbors. $N_neighbor_main_addr$ is the main address of a neighbor, N_status specifies if the node is NOT_SYM or SYM. $N_willingness$ is an integer between 0 and 7, and specifies the node's willingness to carry traffic on behalf of other nodes.

2-hop Neighbor Set

A node records a set of "2-hop tuples" ($N_neighbor_main_addr$, N_2hop_addr , N_time), describing symmetric (and, since MPR links by definition are also symmetric, thereby also MPR) links between its neighbors and the symmetric 2-hop neighborhood. $N_neighbor_main_addr$ is the main address of a neighbor, N_2hop_addr is the main address of a 2-hop neighbor with a symmetric link to $N_neighbor_main_addr$, and N_time specifies the time at which the tuple expires and MUST be removed. In a node, the set of 2-hop tuples are denoted the "2-hop Neighbor Set".

MPR Set

A node maintains a set of neighbors which are selected as MPR. Their main addresses are listed in the MPR Set.

MPR Selector Set

A node records a set of MPR-selector tuples (MS_main_addr, MS_time), describing the neighbors which have selected this node as a MPR. MS_main_addr is the main address of a node, which has selected this node as MPR. MS_time specifies the time at which the tuple expires and must be removed. In a node, the set of MPR-selector tuples are denoted the "MPR Selector Set".

2.4.6.2 Topology Information Base

Each node in the network maintains topology information about the network. This information is acquired from TC-messages and is used for routing table calculations. Thus, for each destination in the network, at least one "Topology Tuple" (T_dest_addr, T_last_addr, T_seq, T_time) is recorded. T_dest_addr is the main address of a node, which may be reached in one hop from the node with the main address T_last_addr. Typically, T_last_addr is a MPR of T_dest_addr. T_seq is a sequence number, and T_time specifies the time at which this tuple expires and must be removed. In a node, the set of Topology Tuples are denoted the "Topology Set". A node maintains a set of neighbor tuples, based on the link tuples. This information is updated according to changes in the Link Set. The Link Set keeps the information about the links, while the Neighbor Set keeps the information about the neighbors. There is a clear association between those two sets, since a node is a neighbor of another node if and only if there is at least one link between the two nodes. As links in an ad-hoc network can be either unidirectional or bidirectional, a protocol for determining the link status is needed.

HELLO messages are broadcasted periodically for neighbor sensing. When a node receives a HELLO message in which its address is found, it registers the link to the source node as symmetric. As an example of how this protocol works, consider two nodes A and B which have not yet established links with each other. Initially A broadcasts an empty HELLO message. When B receives this message and does not find its own address, it registers in the routing table that the link to A is asymmetric. Then B

broadcasts a HELLO message declaring A as an asymmetric neighbor. Upon receiving this message and finding its own address, A registers the link to B as symmetric. A then broadcasts a HELLO message declaring B as a symmetric neighbor, and B registers A as a symmetric neighbor upon reception of this message.

Information about the network topology is extracted from *topology control* (TC) packets. These packets contain the MPR Selector set of a node, and are broadcast by every node in the network, both periodically and when changes in the MPR Selector set are detected. The packets are flooded in the network using the multipoint relaying mechanism. Every node in the network receives such TC packets, from which they extract information to build a topology table.

Main Addresses and Multiple Interfaces

For single OLSR interface nodes, the relationship between an OLSR interface address and the corresponding main address is trivial the main address is the OLSR interface address. For multiple OLSR interface nodes, the relationship between OLSR interface addresses and main addresses is defined through the exchange of Multiple Interface Declaration (MID) messages. Each node with multiple interfaces announce, periodically, information describing its interface configuration to other nodes in the network. This is accomplished through flooding a Multiple Interface Declaration message to all nodes in the network through the MPR flooding mechanism. Each node in the network maintains interface information about the other nodes in the network. This information acquired from MID messages, emitted by nodes with multiple interfaces participating in the MANET, and is used for routing table calculations. Specifically, multiple interface declaration associates multiple interfaces to a node (and to a main address) through populating the multiple interface association base in each node.

The routing table is based on the information contained in the local link information base and the topology set. Therefore, if any of these sets are changed, the

routing table is recalculated to update the route information about each destination in the network. The route entries are recorded in the routing table in the following format:

1. R_dest_addr R_next_addr R_dist R_iface_addr
2. R_dest_addr R_next_addr R_dist R_iface_addr

Each entry in the table consists of R_dest_addr, R_next_addr, R_dist, and R_iface_addr. Such entry specifies that the node identified by R_dest_addr is estimated to be R_dist hops away from the local node, that the symmetric neighbor node with interface address R_next_addr is the next hop node in the route to R_dest_addr, and that this symmetric neighbor node is reachable through the local interface with the address R_iface_addr. Entries are recorded in the routing table for each destination in the network for which a route is known. All the destinations, for which a route is broken or only partially known, are not recorded in the table.

The routing table is updated when a change is detected in either

- the link set,
- the neighbor set,
- the 2-hop neighbor set,
- the topology set,
- the Multiple Interface Association Information Base,

The routing table is recalculated in case of neighbor appearance or loss, when a 2-hop tuple is created or removed, when a topology tuple is created or removed or when multiple interface association information changes. The update of this routing information does not generate or trigger any messages to be transmitted, neither in the network, nor in the 1-hop neighborhood

2.5 Research Gaps

Link Sensing

There exists a research gap in Link sensing as Link Sensing is accomplished through periodic emission of HELLO messages over the interfaces through which

connectivity is checked. A separate HELLO message is generated for each interface. Resulting from Link Sensing is a local link set, describing links between "local interfaces" and "remote interfaces" - i.e., interfaces on neighbor nodes. The Link sensing timer interval can be prolonged based on the Mobility of the nodes in the Ad-hoc Network.

Neighbor detection

Given a network with only single interface nodes, a node may deduct the neighbor set directly from the information exchanged as part of link sensing the "main address"[3] of a single interface node is the address of the only interface on that node. In a network with multiple interface nodes, additional information is required in order to map interface addresses to main addresses and to the nodes. This additional information is acquired through multiple interface declaration (MID) [2][3] messages.

MPR Selection and MPR Signaling

The MPR set of a node is computed such that it, for each interface, The MPR selection for a node is to select a subset of its neighbors such that a broadcast message, retransmitted by these selected neighbors, will be received by all nodes 2 hops away. The information required to perform this calculation is acquired through the periodic exchange of HELLO messages.

The proposed OLSR based Interval Adaptive Routing with efficient MPR' s Routing protocol effectively sense the links for efficient MPR computation and neighborhood detection.

CHAPTER 3

Proposed Strategy to Reduce Control Overhead in OLSR

As part of this dissertation , The proposed OLSR based Interval Adaptive Routing with Efficient MPR's to Minimize control overhead is presented.

3.1 Control Overhead in Link State Routing

Optimized Link Stating Routing (OLSR) optimizes the traditional link state protocol by exploiting multipoint relays (MPRs)[4]. It reduces the size of the control messages rather than declaring all links, a node declares only a subset of links with its neighbors, namely the links to those nodes which are its MPR selectors. OLSR minimizes flooding of control traffic by using MPRs to diffuse its messages. This technique significantly reduces the number of retransmissions in a flooding or broadcast procedure.

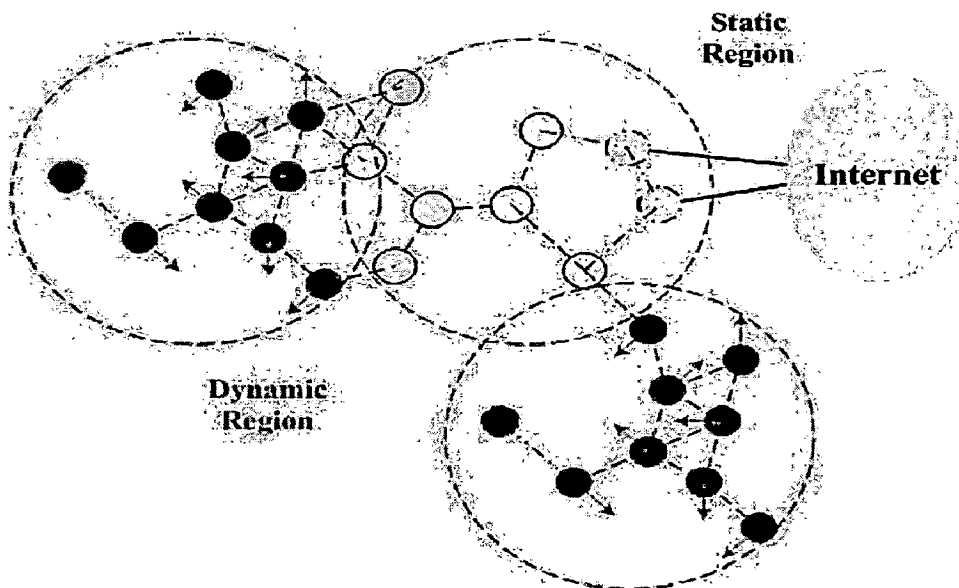


Figure 3.1 Dynamic and Static regions in a Mobile Adhoc Network

In Link State Routing protocols including FSR[7] and OLSR, the control message intervals are constant, regardless of the local mobility[3]. Similar to other routing protocols in wireless networks [7], the performance of LSR is very sensitive to the control message interval in different network conditions. In high mobility conditions, the routes need to be updated more often and the control message interval is shortened to improve performance. The short control message interval will lead to high routing overhead. In slow mobility conditions, the frequent generation of control messages may create excessive and unnecessary overhead since the routes do not need to be updated as often. As shown in Figure 3.1 , the stationary network prefers a long interval while the highly dynamic network the opposite. Hence, it is required to dynamically set the control message interval to be optimal in the specific network condition, the routing overhead can be minimized without jeopardizing the routing reliability.

For traditional LSR with totally N nodes when retransmission and packet reception are error free, the number of transmissions of a TC message is exactly N .

If h is the rate of hello transmission per node and p the rate of TC generation, then the actual routing overhead in traditional LSR[4] is

$$hN + PN^2 \quad \dots\dots\dots (3.1)$$

The routing overhead of FSR[7] is

$$hN + N \cdot \sum_{i=1}^N P_i \quad \dots\dots\dots (3.2)$$

where p_i is decreasing as the distance from the source is increasing, i.e $P_i = P_{default} / hops$

According to [2], the routing overhead of OLSR is

$$hN + pcN^{4/3} \quad \dots\dots\dots (3.3)$$

where c is a constant and $cN^{4/3}$ indicates the average number of retransmission in an MPR flooding.

LSR with optimal control message intervals is integrated with OLSR, the routing overhead is

$$\sum_{i=1}^N h_i + N \sum_{i=1}^{cN^{4/3}} P_i \quad \dots\dots\dots(3.4)$$

Where h_i and p_i are optimal according to the local mobility. For any $h_i \leq h_{default}, P_i \leq P_{default}$ where $h_{default}$ is default rate of hello transmission per node and $p_{default}$ is default rate of Topology control generation.

Performance of LSR is very sensitive to the control message interval in different network conditions. In high mobility[12] conditions, the routes need to be updated more often and the control message interval must be shortened to improve performance. The short control message interval leads to high routing overhead.

3.2 OLSR based Interval Adaptive Link State Routing with efficient MPR's

The Optimized Link State Routing use the following control messages

Hello Message

- Periodically for neighbor sensing
- Contain 1-hop neighbors
- Learn topology up to 2 hops

- Select MPRs to cover 2-hop neighbors

TC – Topology Control Message

- Periodically via MPRs for diffusing link-state
- Contain only MPR selectors
- Update topology table
- Calculate routing table

The Setting emission intervals and holding times in the OLSR are defined as follows

The Vtime in the message header and the Htime in the HELLO message are the fields which hold information about the values in mantissa and exponent format (rounded up).

$$value = C * (1 + a/16) * 2^b \text{ [in seconds]}$$

$$C = 1/16 \text{ seconds (equal to 0.0625 seconds)}$$

C is a scaling factor for the "validity time" calculation ("Vtime" and "Htime" fields in message headers in the OLSR). The "validity time" advertisement is designed such that nodes in a network may have different and individually emission intervals, while still interoperate fully. For protocol functioning and interoperability to work. The advertised holding time must always be greater than the refresh interval of the advertised information to allow for reasonable packet loss. The constant C is set to the suggested value. In order to achieve interoperability, C is same on all nodes. 'validity time' ("Vtime" and "Htime" of packets) to compensate timer resolution, at least in the case where "validity time" could be shorter than the sum of emission interval and maximum expected timer error.

where a is the integer represented by the four highest bits of the field and b the integer represented by the four lowest bits of the field. Given one of the above holding times, a way of computing the mantissa/exponent representation of a number T (of seconds) is

the following find the largest integer 'b' such that $T/C \geq 2^b$ compute the expression $16*(T/(C*(2^b))-1)$, which may not be a integer, and round it up. This results in the value for 'a' if 'a' is equal to 16 increment 'b' by one, and set 'a' to 0 now, 'a' and 'b' should be integers between 0 and 15, and the field will be a byte holding the value $a*16+b$ For instance, for values of 2 seconds, 6 seconds, 15 seconds, and 30 seconds respectively, a and b would be (a=0,b=5), (a=8,b=6), (a=14,b=7) and (a=14,b=8) respectively.

Emission Intervals

HELLO_INTERVAL = 2 seconds

REFRESH_INTERVAL = 2 seconds

TC_INTERVAL = 5 seconds

Holding Time

NEIGHB_HOLD_TIME = 3 x REFRESH_INTERVAL

TOP_HOLD_TIME = 3 x TC_INTERVAL

The following adjustments are done for control overhead minimization

The Hello timer interval and the Topology control timer are adaptively adjusted based on the neighbor hood change.

- Hello_Timer: HELLO_INTERVAL
- TC_Timer: TC_INTERVAL

Adjustments in the Hello timers

If the neighbors of the node change a little the hello interval can be prolonged. There by the control overhead of diffusing Hello messages can be reduced. Similarly if the topology changes a little then Topology control interval can be prolonged.

- Neighbors change little then the Hello_Interval is prolonged
- Topology change little then the TC_Interval is prolonged
- $\text{Hello_Interval} = \text{Hello_FK} * \text{Hello_Interval_Unit}$
- $\text{TC_Interval} = \text{TC_FK} * \text{TC_Interval_Unit}$
 - Hello_Interval_Unit = 2
 - TC_Interval_Unit = 5
- Adjust Hello_FK, TC_FK automatically.

In slow mobility conditions, the frequent generation of control messages may create excessive and unnecessary overhead since the routes do not need to be updated as often. In a large-scale Wireless Network with hybrid mobility, it is very difficult to select a universal control message interval that will work for every node since the mobility condition can be very diverse. The stationary network prefers a long interval while the highly dynamic network the opposite. Hence, if we can dynamically set the control message interval to be optimal in the specific network condition, the routing overhead can be minimized [13].

Multipoint Relays

The idea of multipoint relays is to minimize the overhead of flooding. Messages in the network by reducing redundant retransmissions in the same region[11][12]. Each node in the network selects a set of nodes in its symmetric 1-hop neighborhood which may retransmit its messages. This set of selected neighbor nodes is called the "Multipoint Relay" (MPR) set of that node. The neighbors of node N which are not in its MPR set, receive and process broadcast messages but do not retransmit broadcast messages received from node N. Each node selects its MPR set from among its 1-hop symmetric neighbors. This set is selected such that it covers (in terms of radio range) all symmetric strict 2-hop nodes. The MPR set of N, denoted as $\text{MPR}(N)$, is then an arbitrary subset of the symmetric 1-hop neighborhood of N which satisfies the following condition every node in the symmetric strict 2-hop neighborhood of N must have a symmetric

link towards MPR(N). The smaller a MPR set the less control traffic overhead results from the routing protocol. [2] gives an analysis and example of MPR selection algorithms. Each node maintains information about the set of neighbors that have selected it as MPR. This set is called the "Multipoint Relay Selector set" (MPR selector set) of a node. A node obtains this information from periodic HELLO messages received from the neighbors.

A broadcast message, intended to be diffused in the whole network, coming from any of the MPR selectors of node N is assumed to be retransmitted by node N, if N has not received it yet. This set can change over time (i.e., when a node selects another MPR-set) and is indicated by the selector nodes in their HELLO messages.

Thus dynamically adjusting the Timer Intervals and Selection of the Efficient MPR's reduces the control overhead in the dense mobile Adhoc networks.

CHAPTER 4

System Design and Implementation

4.1 System Design Components:

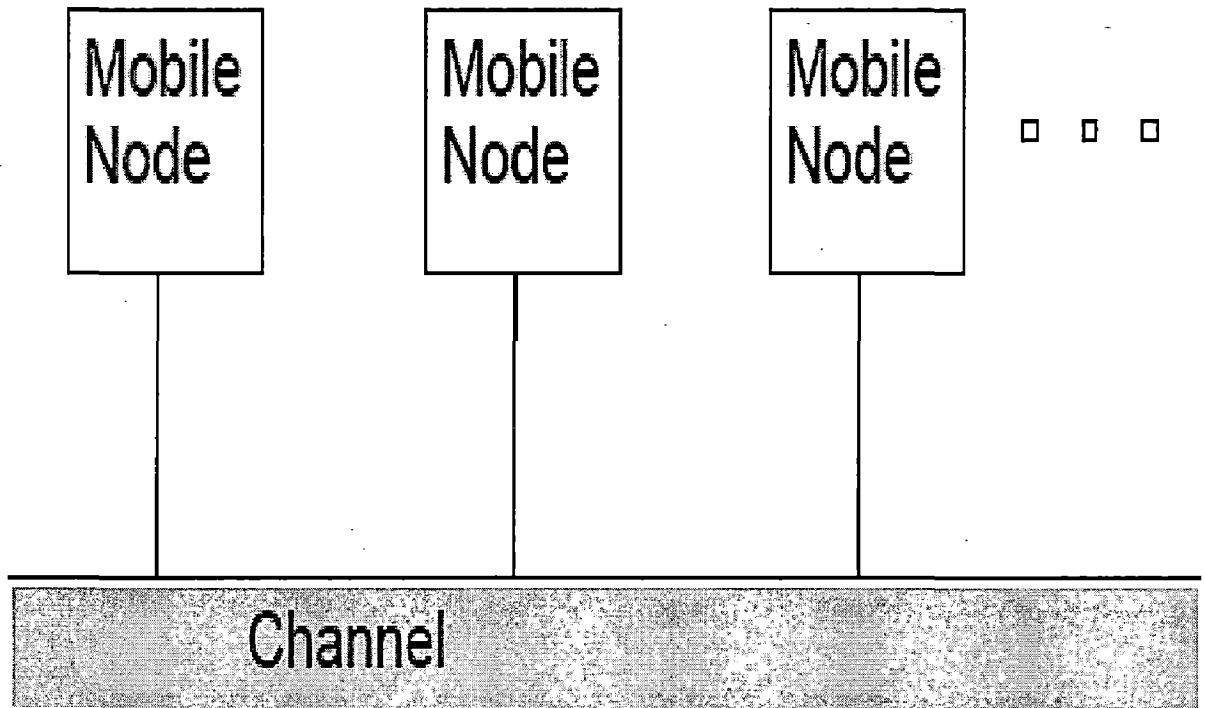


Figure 4.1 : Logical view of the Mobile Nodes connecting to the channel

Figure 4.1 shows a logical overview of nodes connected to the channel. These are connected together using the CMU Monarch extensions[2] to *ns*. Each *mobile node* is an independent entity that is responsible for computing its own position and velocity as a function of time. Each mobile node can have one or more *network interfaces*, each of which is attached to a *channel*. Channels are the conduits that carry packets between mobile nodes. When a mobile node transmits a packet onto a channel, the channel distributes a copy of the packet to all the other network interfaces on the channel. These interfaces then use a *radio propagation model* to determine if they are actually able to

receive the packet. Typically, all the network interfaces of the same type should be connected to one channel. A mobile node can have more than one type of network interface, and these interfaces can be connected into different channels. Conceptually, a channel represents a particular radio frequency with a particular modulation[17].

4.1.1 Network Interfaces

Shared Media This network interface implements a shared media model where, subject to collisions and the propagation model, each node can overhear packets transmitted by the others.

Link Layer The LL used by mobilenode 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.

Interface Queue The PriQueue 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[17].

Mac Layer NS-2 has the implementation of IEEE 802.11 distributed coordination function (DCF) from CMU[17].

Network Interfaces The Network layer serves as a hardware interface which is used by mobilenode to access the channel. The wireless shared media interface is implemented as class Phy/WirelessPhy. This interface subject to collisions and the radio propagation

model receives packets transmitted by other node interfaces to the channel..

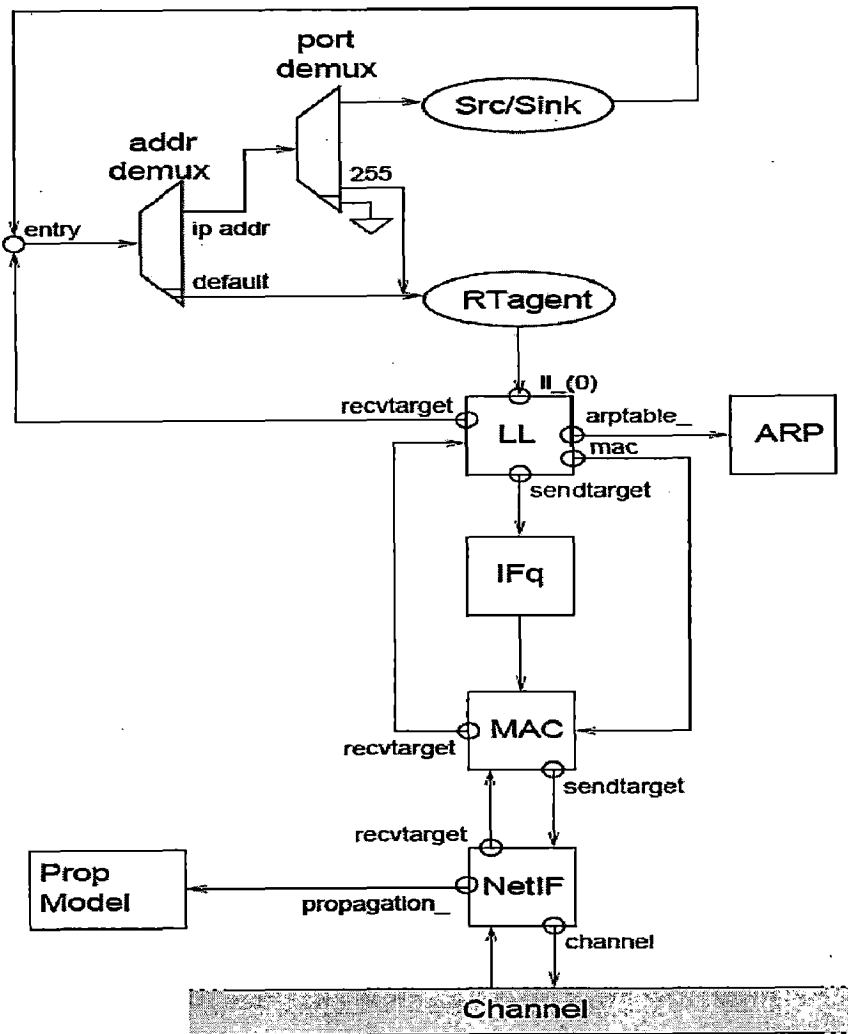


Figure 4.2 Mobile Node Interfaces

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

Antenna An omni-directional antenna is used by mobilenodes.

```
$ns_ node-config -adhocRouting $opt(adhocRouting)
                 -llType $opt(ll)
                 -macType $opt(mac)
                 -ifqType $opt(ifq)
                 -ifqLen $opt(ifqlen)
                 -antType $opt(ant)
                 -propInstance [new $opt(prop)]
                 -phyType $opt(netif)
                 -channel [new $opt(chan)]
                 -topoInstance $topo
                 -wiredRouting OFF
                 -agentTrace ON
                 -routerTrace OFF
                 -macTrace OFF
```

The above API configures for a mobilenode with all the given values of adhoc-routing protocol, network stack, channel, topography, propagation model, with wired routing turned on or off (required for wired-cum-wireless scenarios) and tracing turned on or off at different levels (router, mac, agent).

4.1.2 Outgoing Packets

Packets sent by a source on the mobile node are handed to the mobile node's entry point, which passes them to the address demultiplexer. The address demux tests the destination IP address of the packet, and, if it matches the node's own address, passes the packet up to the port demux. Since packets sent by a source on this node are typically destined to another node, most packets will match the default target of the address demux and be handed down to the routing protocol. The routing protocol is responsible for setting the `next_hop` field in the packet's common header to the address of the next node that should process the packet, and then passing the packet down to the link layer (LL). If the next

hop address is an IP address, the LL object queries the ARP object to translate the IP address to a hardware address. If the ARP object does not already have a mapping for the IP address, it will queue the packet and instruct the LL object to send an ARP Request packet with a broadcast hardware address in order to perform address resolution. Once the hardware address of a packet's next hop is known, the packet is inserted into the interface queue (IFq). The media access control object (MAC) takes packets from the head of the interface queue and sends them to the networking interface (NetIF) when appropriate. The network interface stamps the common header of the packet with properties such as the power and position of the transmitting interface, and then passes the packet to the channel, where a copy is made for all the other interfaces on the channel.

4.1.3 Incoming Packets

A copy of each packet sent onto a channel is delivered to each of the network interfaces on the channel at the time at which the first bit of the packet would begin arriving at the interface in a physical system, based on the distance between the nodes and the speed of light. Each network interface stamps the packet with the receiving interface's properties and then invokes the propagation mode. The propagation model uses the transmit and receive stamps and the properties of the receiving interface to determine the power with which the interface will receive the packet. The receiving network interfaces then use their properties to determine if they actually successfully receive the packet, and hand the packet to their MAC layer if appropriate. If the MAC layer receives the packet collision- and error-free, it passes the packet to the mobile node's entry point. If this is the final destination of the packet, the address demux will pass the packet to the port demux, which will hand the packet to the proper sink agent. If this is not the packet's final destination, it will go to the default target of the address demux, and the routing agent at this node will be called on to assign the packet a next hop and pass the packet back to the link layer.

4.1.4 Adding User Level Agents to a Mobile Node

There are several ways to attach the agents which send and receive packets to the MobileNodes that will carry them around and provide them with network layer services. In the CMU Monarch[2] simulation scheme, commands that create, configure, and control these agents in a *communication pattern* file.

```
set cbr_(4) [$ns_ create-connection CBR $node_(6) CBR $node_(7) 0]
```

create-connection is a method of the simulator object that creates the connection. The method first creates the CBR source and sink, and then calls the attach method of the nodes to associate them with the respective node.

4.2 Implementation

4.2.1 Computing of the MPR Set.

- I) Start with an MPR set made of all members of Neighbors with N_willingness [3] equal to WILL_ALWAYS

- II) Calculate The degree of a 1-hop neighbor node 'A' (where 'A' is a member of Neighbor set), which is defined as the number of symmetric neighbors of node 'A', excluding all the members of Neighbors and excluding the node performing the computation, where y is a member of N, for all nodes in N.

- III) Add to the MPR set those nodes in Neighbors, which are the nodes to provide reachability to a node in 2-hop neighbor . For example, if node b in 2-hop neighbor is reachable only through a symmetric link to node a in Neighbor , then add node a to the MPR set. Remove the nodes from 2-hop neighbor which are now covered by a node in the MPR set.

- IV) While there exist nodes in 2-hop neighbor which are not covered by at least one node in the MPR set

- a) For each node in N , calculate the reachability, i.e., the number of nodes in 2-hop neighbor which are not yet covered by at least one node in the MPR set, and which are reachable through this 1-hop neighbor.
 - b) Select as a MPR the node with highest $N_willingness$ among the nodes in Neighbor set with non-zero reachability. In case of multiple choice select the node which provides reachability to the maximum number of nodes in 2-hop neighbor set. In case of multiple nodes providing the same amount of reach ability, select the node as MPR whose one- hop degree ('A') is greater. Remove the nodes from N_2 which are now covered by a node in the MPR set.
- V) A node's MPR set is generated from the union of the MPR sets for each interface. As an optimization, process each mobile node, 'A', in the MPR set in increasing order of $N_willingness$. If all nodes in N_2 are still covered by at least one node in the MPR set excluding node 'A', and if $N_willingness$ of node y is smaller than $WILL_ALWAYS$, then node 'A' is removed from the MPR set.

4.2.2 Adjustments in the Hello and Topology timers

- Neighbors change little->Hello_Interval prolong
- Topology change little->TC_Interval prolong

If neighbors change a little then the hello message timer interval can be prolonged. If route table do not change then the topology message control interval are prolonged. There by reducing the number of hello and topology control messages diffused in the network.

Algorithm

- i) In the processing of the received Hello Packets neighbor list of node 'N' is updated.

If neighbors list is not changed

hello_count is incremented by 1

- a) If(hello_count > Neighborcount)

hello_frequency is incremented

- b) If(hello_frequency is greater than MAX_hello frequency)

Hello frequency is assigned the max_hello_frequency value

Else

hello_count is initialized to 0

hello_frequency is assigned to 1

Topology control frequency is assigned to 1

- ii) In the processing of the topology messages of node N

Update the routing table

If routing table is not changed

Increment the TC_frequency

- (a) If TC_frequency is greater than the MAX_TC_frequency

TC_frequency is assigned the value of MAX_TC_frequency

Else

TC_frequency is assigned 1

Thus the above algorithm computes the hello and topology control frequency which are used in the determination of the Hello_Interval and TC_Interval as follows.

- $\text{Hello_Interval} = \text{Hello_frequency} * \text{Hello_Interval_Unit}$
- $\text{TC_Interval} = \text{TC_frequency} * \text{TC_Interval_Unit}$
 - Hello_Interval_Unit = 2
 - TC_Interval_Unit = 5
- Adjust Hello_frequency, TC_frequency automatically.

4.2.3 Packet processing and forwarding

Upon receiving a basic packet, a node examines each of the "message headers". Based on the value of the "Message Type" field, the mobile node can determine the fate of the message.

A mobile node may receive the same message several times. Thus, to avoid re-processing of some messages which were already received and processed, each node maintains a duplicate Set. In this set, the node records information about the most recently received messages where duplicate processing of a message is to be avoided. For such a message, a node records a duplicate tuple (D_addr, D_seq_num, D_retransmitted, D_iface_list, D_time), where D_addr is the originator address of the message, D_seq_num is the message sequence number of the message, D_retransmitted is a boolean indicating whether the message has been already retransmitted, D_iface_list is a list of the addresses of the interfaces on which the message has been received and D_time specifies the time at which a tuple expires and are removed. In a node, the set of Duplicate Tuples are denoted the "Duplicate set".[3][4]

"Originator Address" is be used for the main address of the node which sent the message. The term "Sender Interface Address" is used for the sender address (given in the IP header of the packet containing the message) of the interface which sent the message. The term "Receiving Interface Address" is used for the address of the interface of the node which received the message.

Thus, upon receiving a basic packet, a mobile node performs the following tasks for each encapsulated message:

- 1 If the packet contains no messages (i.e., the Packet Length is less than or equal to the size of the packet header), the packet is silently be discarded. For IPv4 addresses, this implies that packets, where the Packet Length < 16 MUST silently be discarded.

2 If the time to live of the message is less than or equal to '0' (zero), or if the message was sent by the receiving node (i.e., the Originator Address of the message is the main address of the receiving node): the message is silently be dropped.

3 Processing condition:

3.1 if there exists a tuple in the duplicate set, where:

$D_addr == \text{Originator Address, AND}$

$D_seq_num == \text{Message Sequence Number}$

then the message has already been completely processed and is not be processed again.

3.2 Otherwise, if the node implements the Message Type of the message, the message is processed according to the specifications for the message type.

4 Forwarding condition:

4.1 if there exists a tuple in the duplicate set, where:

$D_addr == \text{Originator Address, AND}$

$D_seq_num == \text{Message Sequence Number,}$

AND

the receiving interface (address) is in D_iface_list

then the message has already been considered for forwarding and is not retransmitted again.

4.2 Otherwise:

4.2.1

If the node implements the Message Type of the message, the message is considered for forwarding according to the specifications for the message type.

4.2.2

Otherwise, if the node does not implement the Message Type of the message, the message is processed according to the default forwarding algorithm described below.

Default Forwarding Algorithm

The default forwarding algorithm is the following:

1 If the sender interface address of the message is not detected to be in the symmetric 1-hop neighborhood of the node, the forwarding algorithm is silently stop here (and the message is not forwarded).

2 If there exists a tuple in the duplicate set where

$D_addr == \text{Originator Address}$

$D_seq_num == \text{Message Sequence Number}$

Then the message will be further considered for forwarding if and only if $D_retransmitted$ is false, AND the (address of the) interface which received the message is not included among the addresses in D_iface_list

3 Otherwise, if such an entry doesn't exist, the message is further considered for forwarding.

If after those steps, the message is not considered for forwarding, then the processing i.e., steps 4 to 8 are ignored, otherwise, if it is considered for forwarding then the following algorithm is used

4 If the sender interface address is an interface address of a MPR selector of this node and if the time to live of the message is greater than '1', the message is retransmitted

5 If an entry in the duplicate set exists, with same Originator Address, and same Message Sequence Number, the entry is updated as follows:

$D_time = \text{current time} + \text{DUP_HOLD_TIME.}$

The receiving interface (address) is added to D_iface_list .

$D_retransmitted$ is set to true if and only if the message will be retransmitted according to step 4. Otherwise an entry in the duplicate set is recorded with

$D_addr = \text{Originator Address}$

$D_seq_num = \text{Message Sequence Number}$

$D_time = \text{current time} + \text{DUP_HOLD_TIME.}$

D_iface_list contains the receiving interface address.


$D_{\text{retransmitted}}$ is set to true if and only if the message will be retransmitted according to step 4.

If, and only if, according to step 4, the message must be retransmitted then

- 6 The TTL of the message is reduced by one.
- 7 The hop-count of the message is increased by one
- 8 The message is broadcast on all interfaces.

4.3 Simulation Parameters

Table No. 4.1

| Parameter | Value | Description |
|---------------------|------------------|--|
| Simulator | ns-2 | Simulation tool |
| Number of Nodes | 50 | Network nodes |
| Communication Range | 100 Meters | Mobile Node can communicate within this Range |
| Max Speed | 10 Meters/sec | Mobile Nodes at a Maximum speed of 10 meters/sec. |
| Simulation time | 0-100,0-600 Sec | Simulation duration |
| Traffic | UDP | Constant Bit rate |
| No. of Flows | 50 | No. of Mobile Nodes |
| Ground | 1000x1000 meters |  X RNDY TOPOGRAPHY |

4.4 Trace File

Event type In the traces, the first field 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 energy level
- **-Nl:** trace level, such as AGT, RTR, MAC
- **-Nw:** reason for the 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_RETRY_COUNT_EXCEEDED
- **"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_ARP_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 towards 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:** dst's ethernet address
- **-Ms:** src's ethernet address
- **-Mt:** ethernet type

Packet info at "Application level" The packet information at application level[18] consists of the type of application like ARP, TCP, the type of adhoc routing protocol like OLSR based interval adaptive routing protocol is traced. This field consists of a leading "-P" and list of tags for different application is listed as below:

- **-P arp** Address Resolution Protocol. Details for ARP is given by the following tags:
 - **-Po:** ARP Request/Reply
 - **-Pm:** src mac address
 - **-Ps:** src address
 - **-Pa:** dst mac address
 - **-Pd:** dst address
- **-P polsr** This denotes the adhoc routing protocol (proposed OLSR).

Information on proposed OLSR is represented by the following tags:

- **-Pn:** how many nodes traversed
- **-Pq:** routing request flag
- **-Pi:** route request sequence number
- **-Pp:** routing reply flag
- **-Pl:** reply length
- **-Pe:** src of srcrouting->dst of the source routing
- **-Pw:** error report flag
- **-Pm:** number of errors
- **-Pc:** report to whom
- **-Pb:** link error from linka->linkb
- **-P cbr** Constant bit rate.

Information about the CBR application is represented by the following tags:

- **-Pi:** sequence number
- **-Pf:** how many times this pkt was forwarded
- **-Po:** optimal number of forwards
- **-P tcp** Information about TCP flow is given by the following subtags:

- **-Ps:** seq number
- **-Pa:** ack number
- **-Pf:** how many times this pkt was forwarded
- **-Po:** optimal number of forwards

Generation of node-movement and traffic-connection for wireless scenarios

Normally for large topologies, the node movement and traffic connection patterns are defined in separate files for convenience. These movement and traffic files may be generated using CMU's movement- and connection-generators[14].

MobileNode Movement

Some of the node movement files can be represented by 670x670-50-600-20-*. These files define a topology of 670 by 670m where 50 nodes move with a speed of 20m/s with pause time of 600s. The information regarding number of hops between the nodes is fed to the central object "GOD" . Next each node is a speed and a direction to move to. The generator for creating node movement files can be generated in the following way [19]

```
./setdest -n <num_of_nodes> -p <pausetime> -s <maxspeed> -t <simtime> -x <maxx> -y
<maxy> > <outdir>/<scenario-file>
```

Nodes are created from 0 instead of 1 as was in the original

Generating traffic pattern files

The Traffic files are very important for establishing communication between nodes

The traffic generator are called cbrgen.tcl and tcpgen.tcl. They are used for generating CBR and TCP connections respectively.

The CBR connections are created by

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

The TCP connections are created by running

```
ns tcpgen.tcl [-nn nodes] [-seed seed] [-mc connections] [-rate rate]
```


CHAPTER 5

RESULTS AND DISCUSSION

An analysis of the simulation experiments carried out in ns-2 [2] are as follows.

5.1 Results

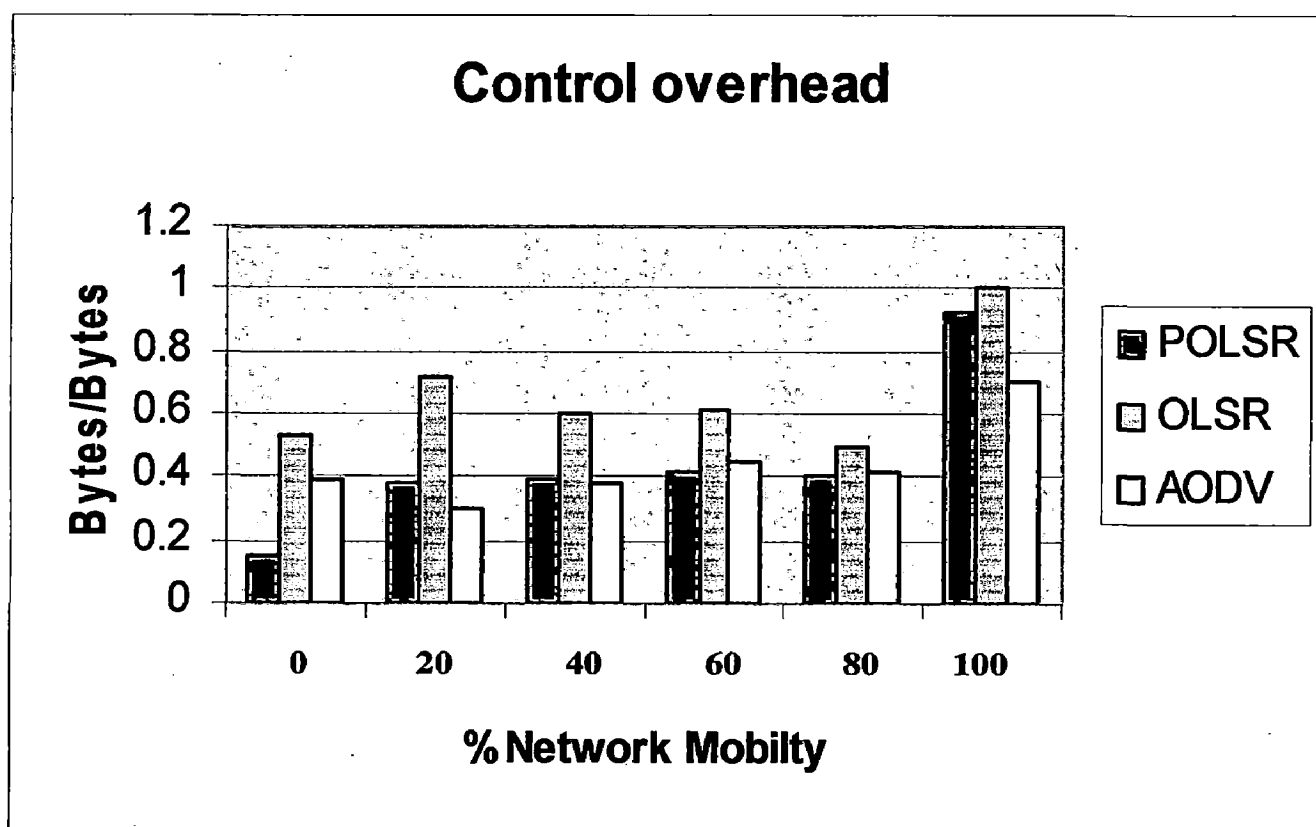


Figure 5.1 Mobilty vs control overhead

Figure 5.1 compares the proposed OLSR based Interval Adaptive Routing with efficient MPR's routing protocol(POLSR) with OLSR and AODV. And it is clear that in high mobility Ad-hoc networks the proposed OLSR based Interval Adaptive Routing with efficient MPR' s Routing protocol outperforms the OLSR and AODV. The good performance of the proposed protocol comes in view of the Interval Adaptive and Efficient selection of the MPR's .

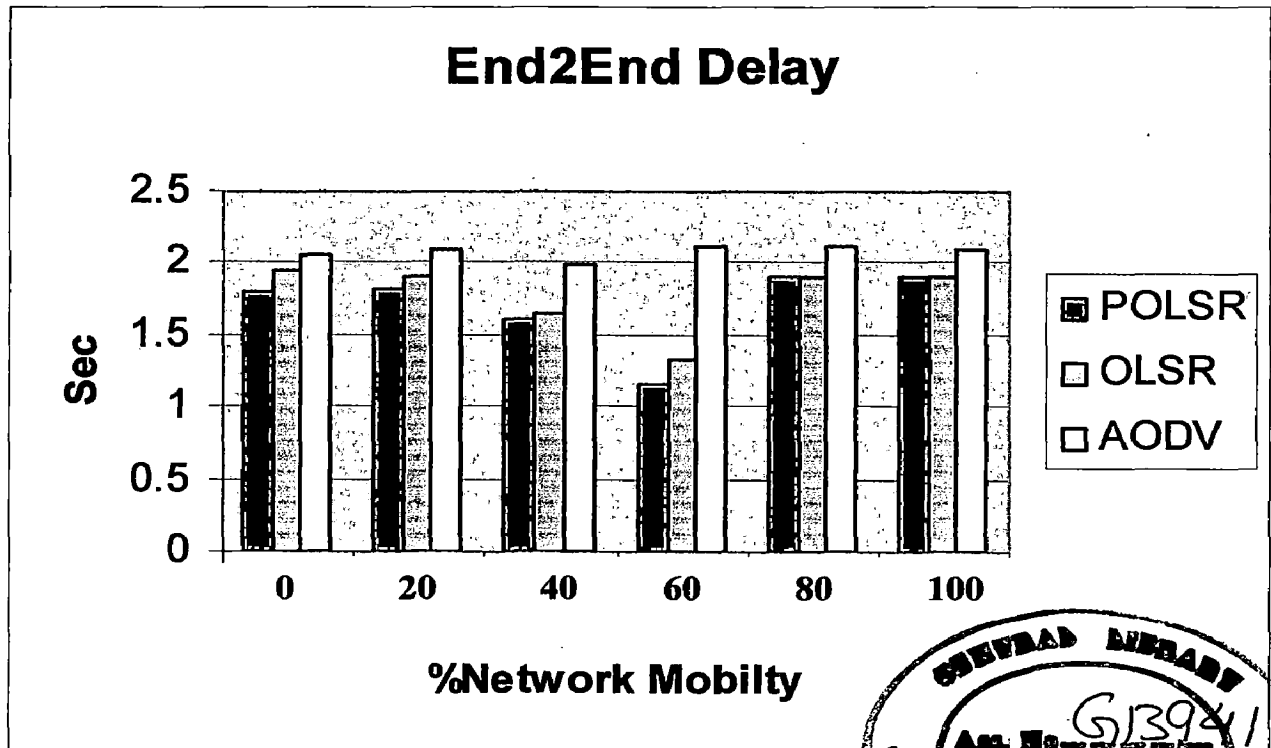


Figure 5.2 Mobilty vs end to end delay



Figure 5.2 illustrates The end to end delay is reduced with the help of the proposed OLSR based Interval Adaptive Routing with efficient MPR's Routing protocol (POLSR). The comparison between the proposed routing protocol , OLSR and the AODV are depicted as shown in the figure 5.2. The proposed protocol performs well in case of high mobility also due to the increased availability of reliable routes in the network, which reduces the end to end delay in the network. The proposed protocol dynamically set the control message interval to be optimal there by minimizing the routing overhead without jeopardizing the reliability of the routes.

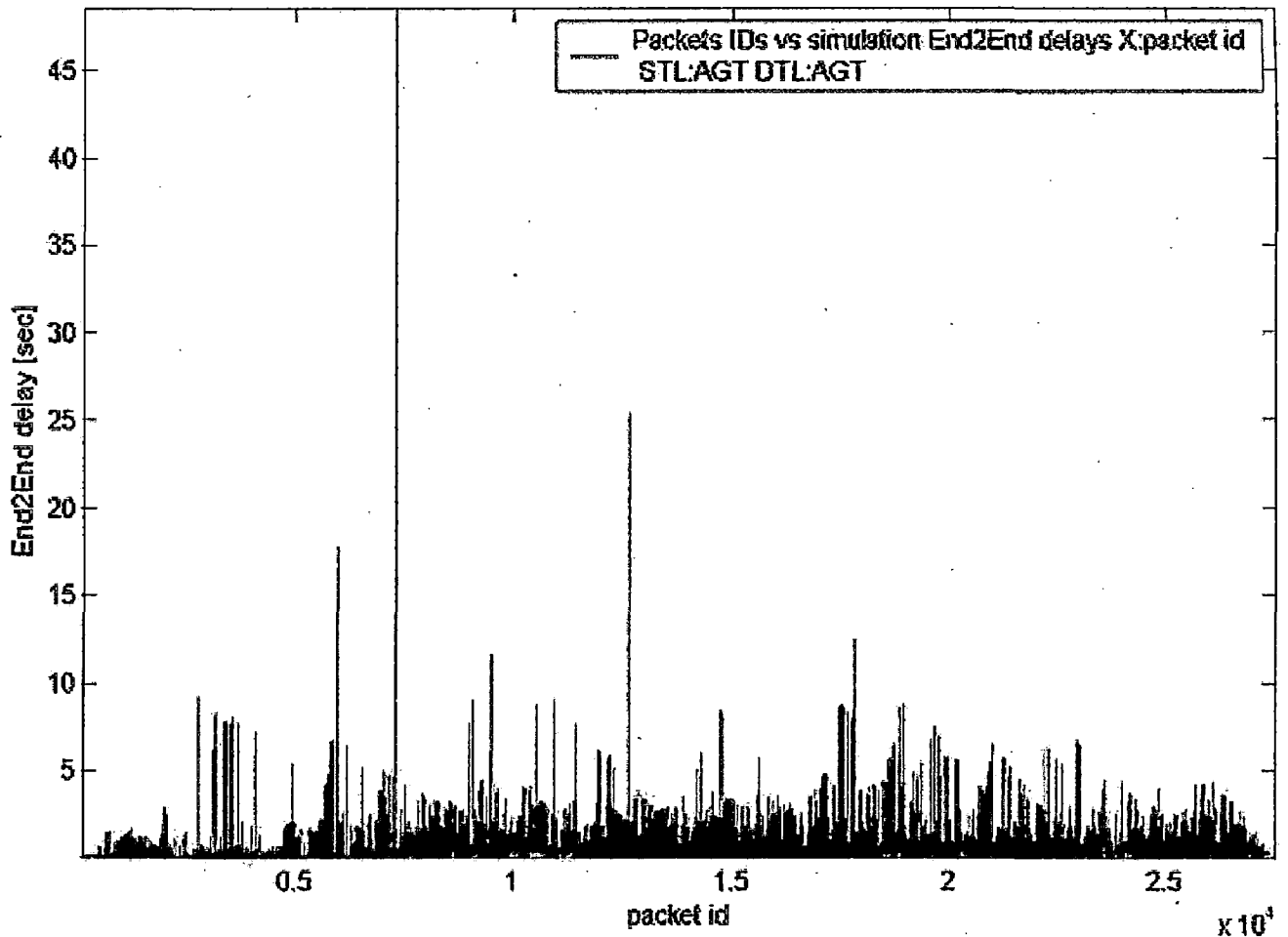


Figure 5.3 packet id vs end to end delay

Figure 5.3 illustrates the packet id and End to End Delay of the proposed OLSR based Interval Adaptive Routing with efficient MPR' s. The packets id are plotted against the corresponding end to end delays. The OLSR based Interval Adaptive Routing with efficient MPR' s Routing protocol computes the reliable routes dynamically there by decreasing the average End to End delay.

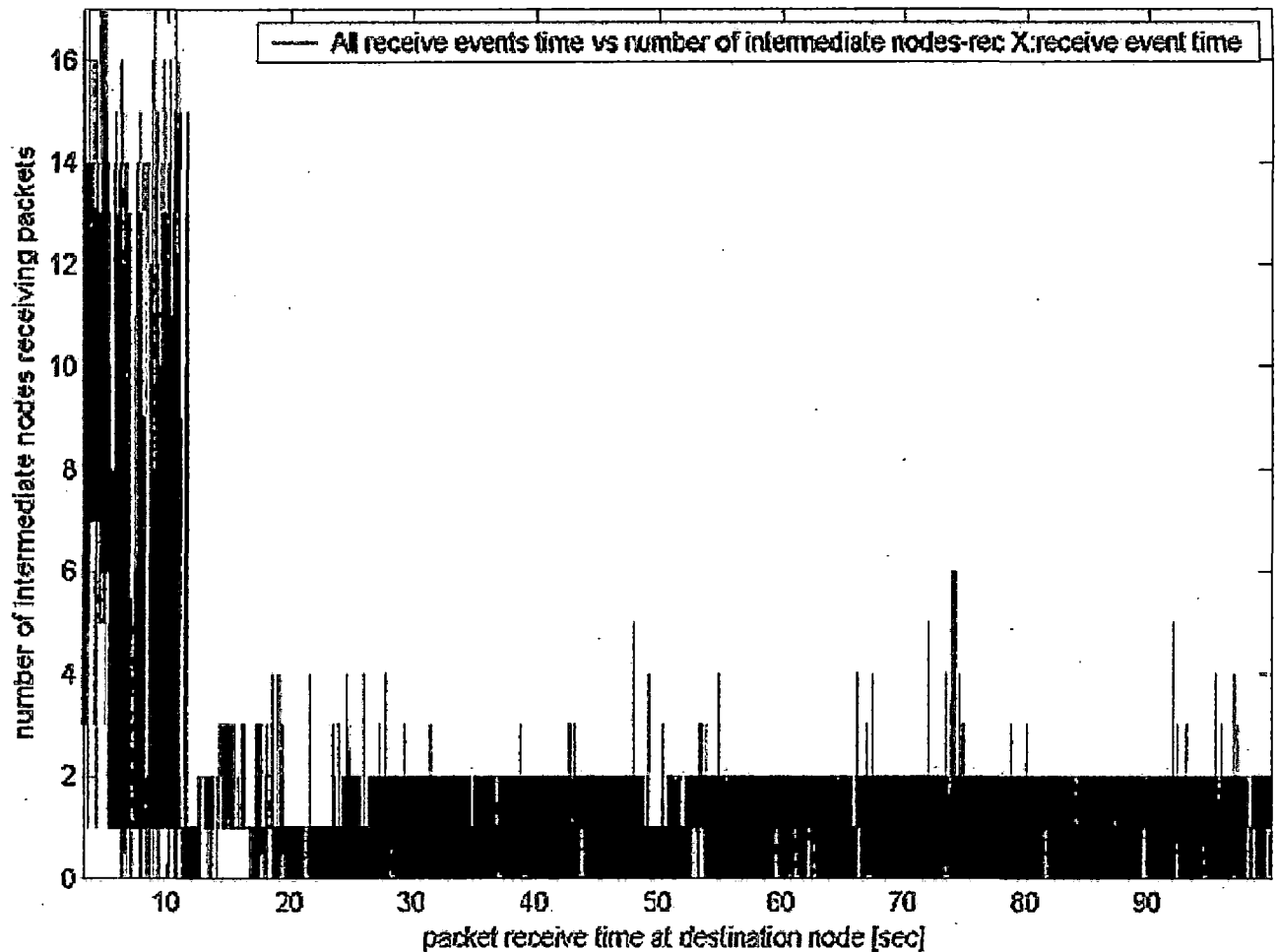


Figure 5.4 Packet receive time vs No. of intermediate Nodes

Figure 5.4 illustrates the number of intermediate nodes involved in receiving packets in the proposed OLSR based Interval Adaptive Routing with efficient MPR's to Minimize Control over head. The proposed algorithm computes MPR set and thus reliable routes are established. The message forwarding is done through the MPR's by which number of intermediate nodes receiving the packets are drastically reduced.

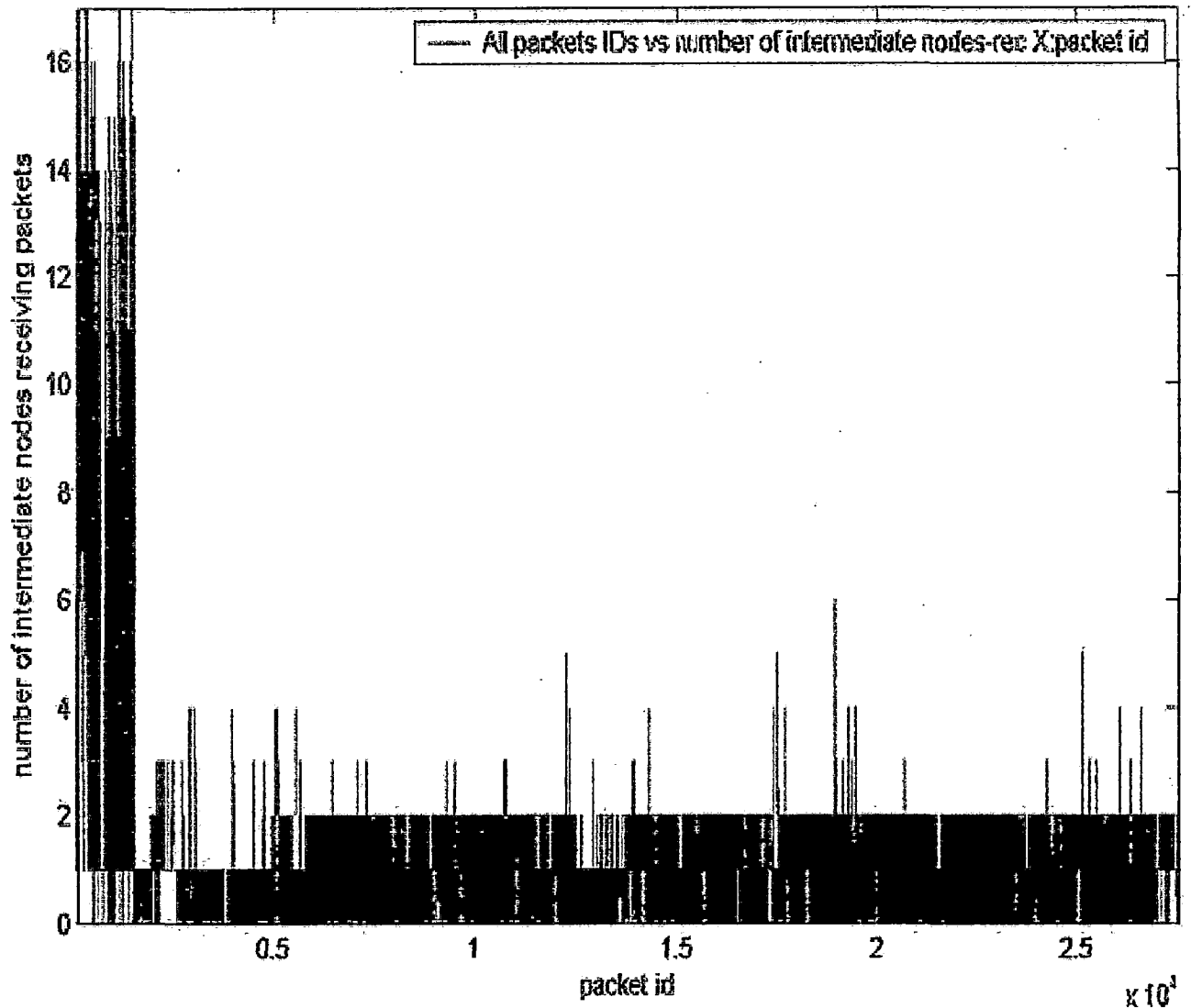


Figure 5.5 Packet id vs No. of intermediate nodes receiving packets

Figure 5.5 illustrates the number of intermediate nodes involved in receiving packet id of the proposed OLSR based Interval Adaptive Routing with efficient MPR's to minimize control overhead. The proposed algorithm computes MPR set and thus reliable routes are established. The message forwarding is done through the MPR's there by average number of intermediate nodes receiving the packets are drastically reduced

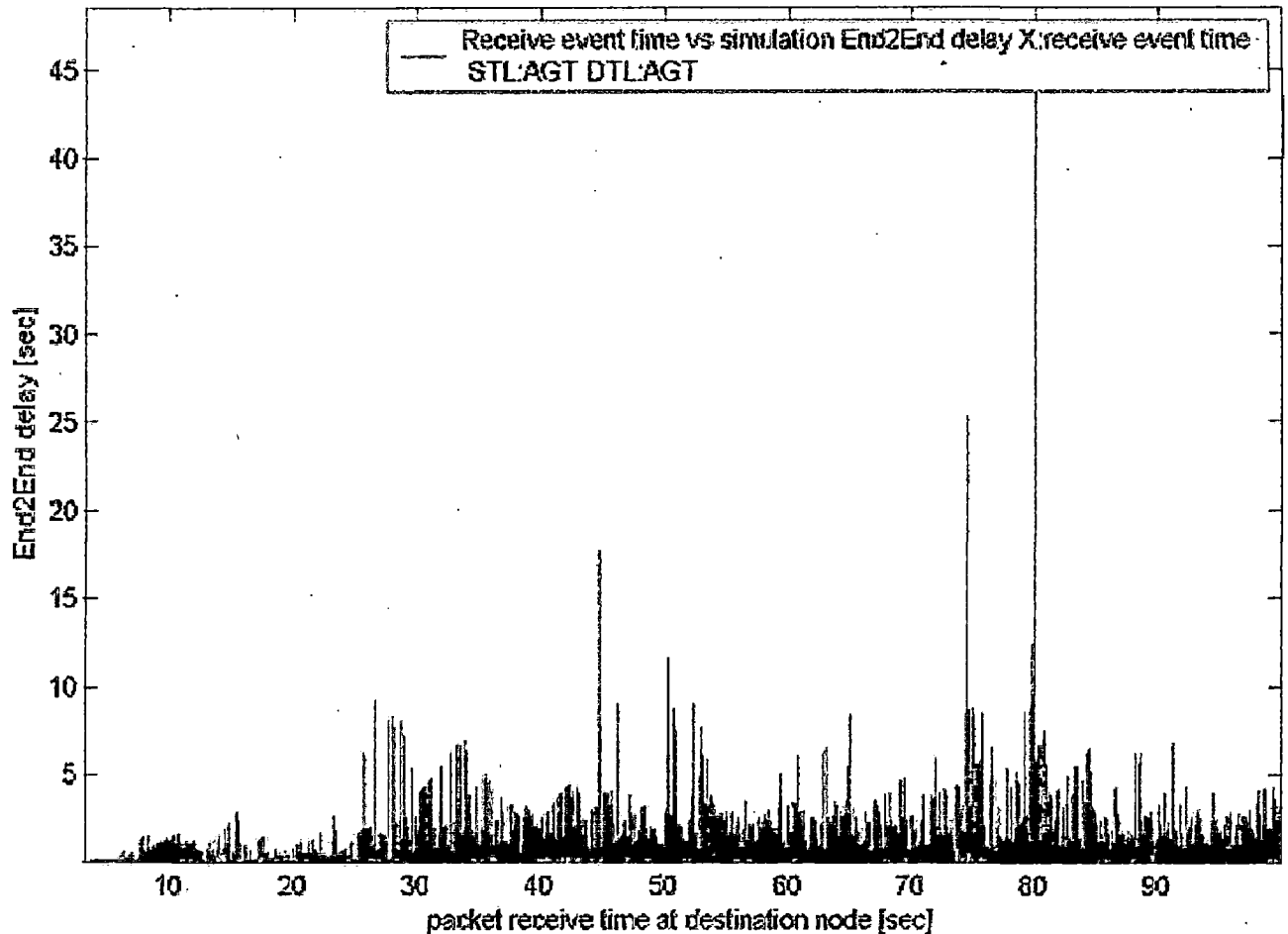


Figure 5.6 Packet receive time at destination node vs end to end delay

Figure 5.6 illustrates the packet receive time at the destination node and End to End Delay of the OLSR based Interval Adaptive Routing with efficient MPR's . The packets receive time are plotted against the corresponding end to end delays. The OLSR based Interval Adaptive Routing with efficient MPR' s Routing protocol computes the reliable routes there by in dynamic environment also there exists a reliable routing path at any instant.

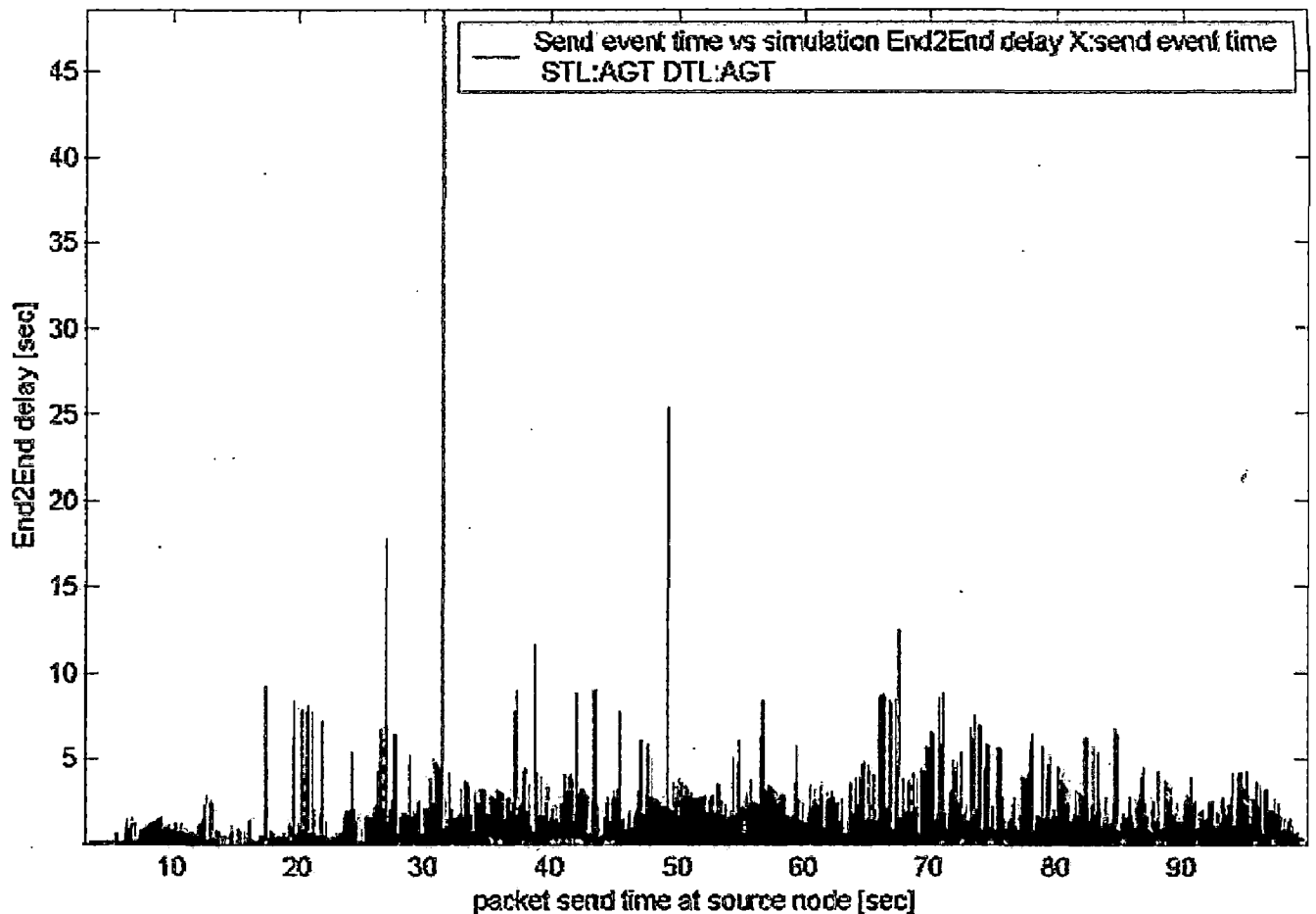


Figure 5.7 packet send time at source node vs end to end delay

Figure 5.7 illustrates the packet send time at the source node and End to End Delay of the OLSR based Interval Adaptive Routing with efficient MPR' s. The packets send time at the source node are plotted against the corresponding end to end delays. The OLSR based Interval Adaptive Routing with efficient MPR' s Routing protocol computes the reliable routes there by in dynamic environment also there exists a reliable routing path from the source to destination at any Instant.

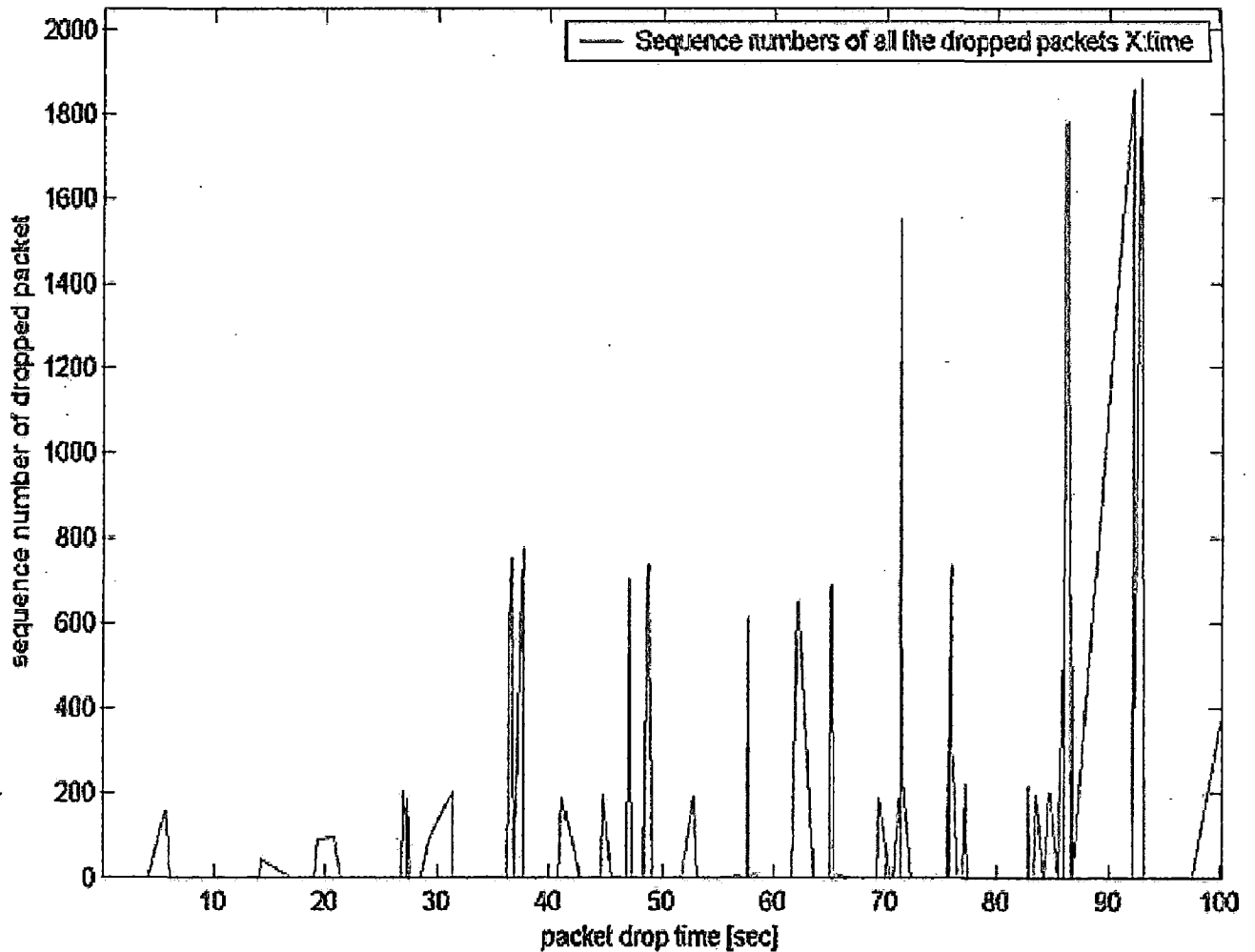


Figure 5.8 Packet drop time vs sequence number of the dropped packet

Figure 5.8 illustrates the packet drop time and sequence number of the dropped packets of the OLSR based Interval Adaptive Routing with efficient MPR's. The packets id are plotted against the corresponding end to end delays. The OLSR based Interval Adaptive Routing with efficient MPR's Routing protocol computes the MPR's and the messages are broadcasted through the MPR's. Initially MPR's are not computed and the packets are forwarded entire through the MANET. After computation of the MPR set the control messages are broadcasted via MPR's.

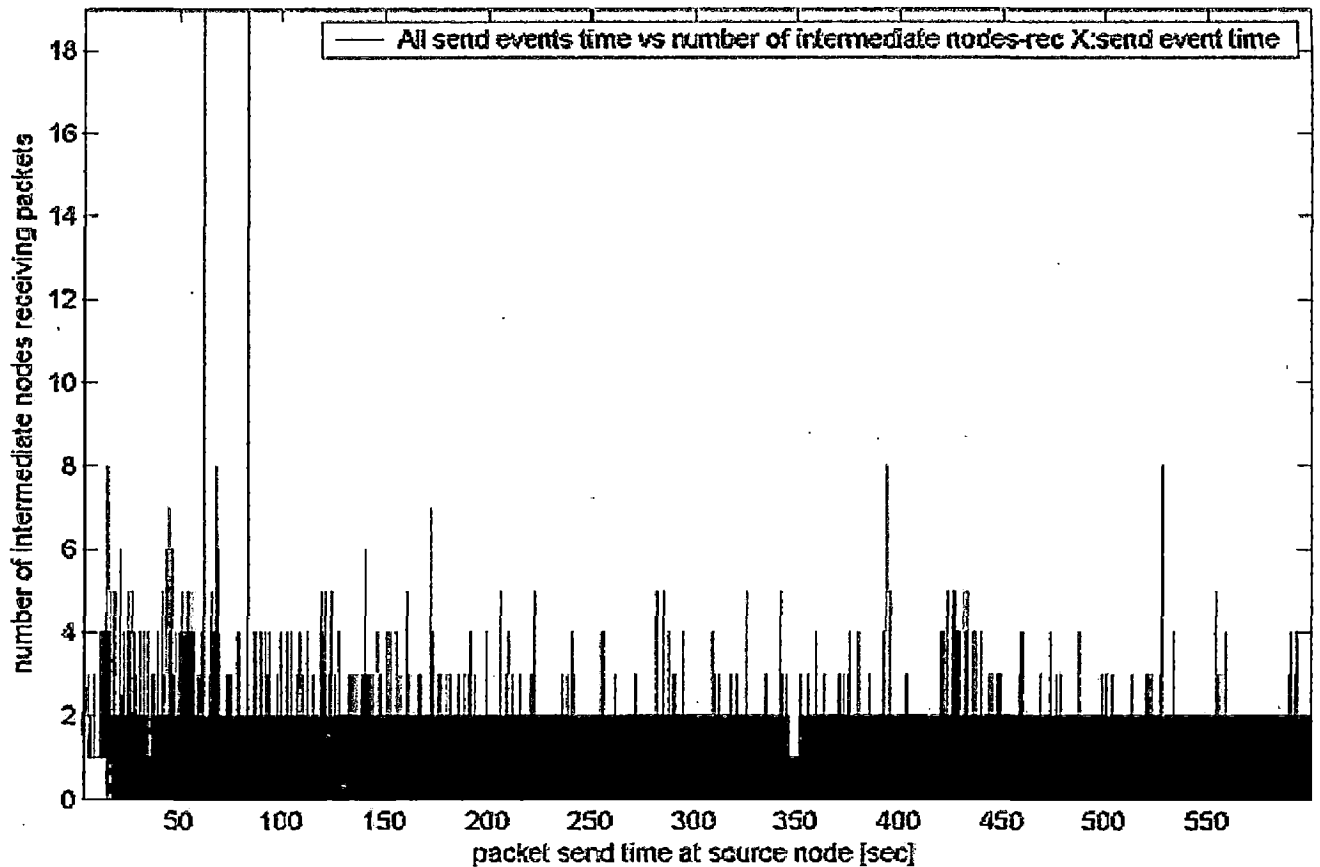


Figure 5.9 Packet send time at source node vs No. of Intermediate Nodes

Figure 5.9 illustrates the OLSR based Interval Adaptive Routing with efficient MPR's simulation for 600 sec. Thus the graph between Packet send time at the source node and number of Intermediate nodes is stable when compared to the OLSR protocol illustrated in the figure 5.10. the proposed routing protocol computes and has reliable routes, as mobility of the nodes effect a little due to adaptive frequency of diffusing Hello and topology control messages for neighbor sensing and route table computation. The average Number of Intermediate nodes involved in packet send is minimized when compared to the OLSR.

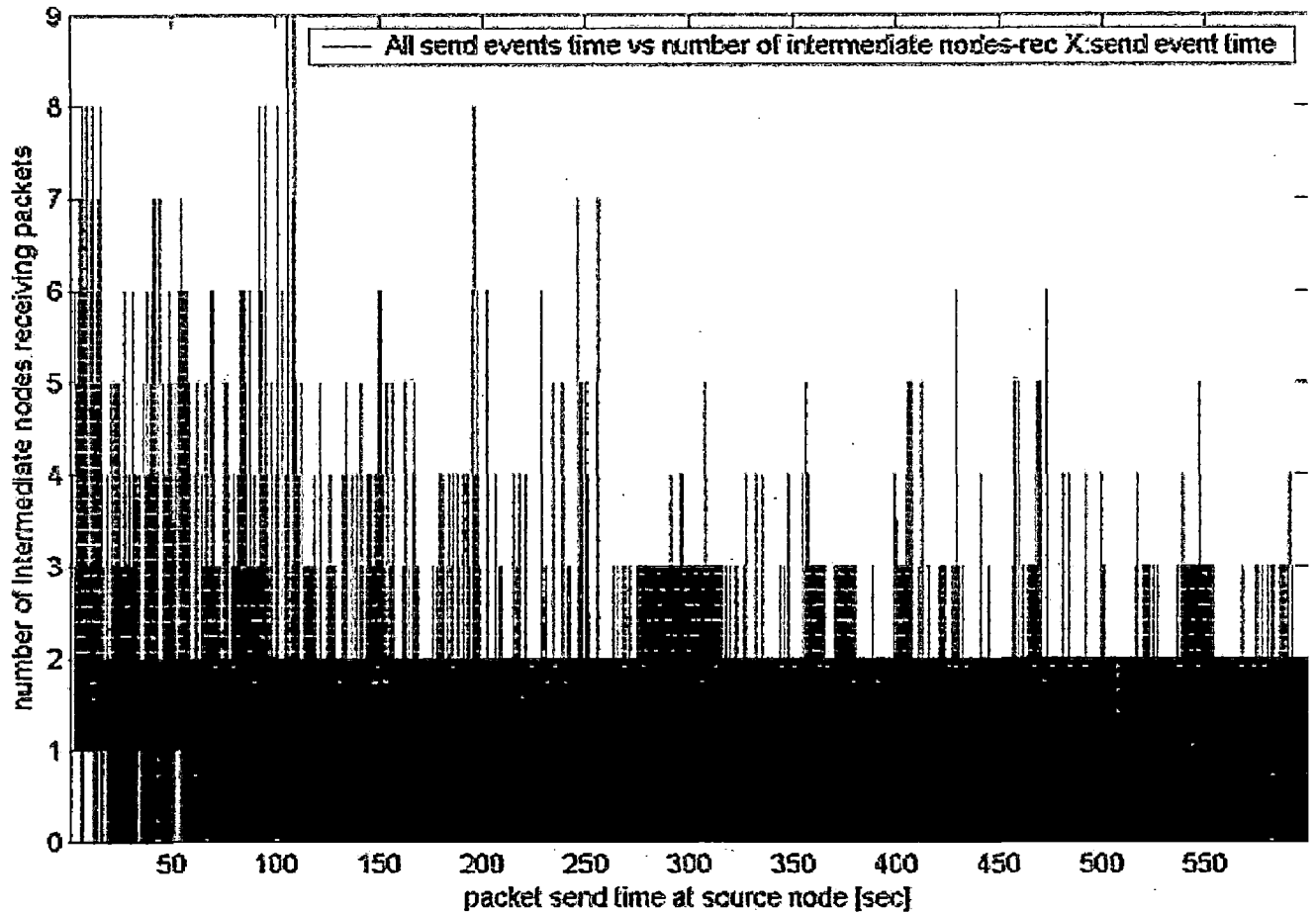


Figure 5.10 Packet send time at source node vs No. of Intermediate Nodes receiving packets

Figure 5.10 illustrates OLSR Packet send time at source node and Number of Intermediate Nodes receiving packets. The OLSR do not compute the routes based on the Mobility of the nodes. OLSR computes the routes based on fixed timer intervals. So, as the mobility increases the network topology changes and average number of Intermediate Nodes receiving the packets increases.

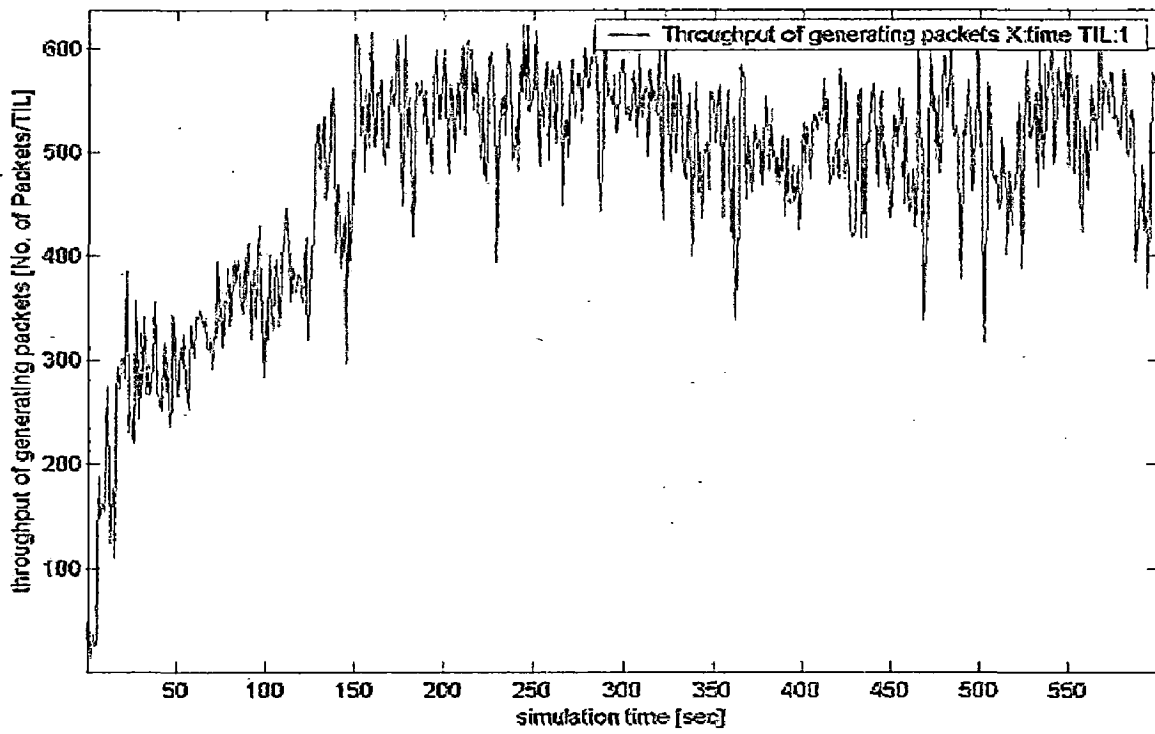


Figure 5.11 Simulation time vs Through put of generating Packets

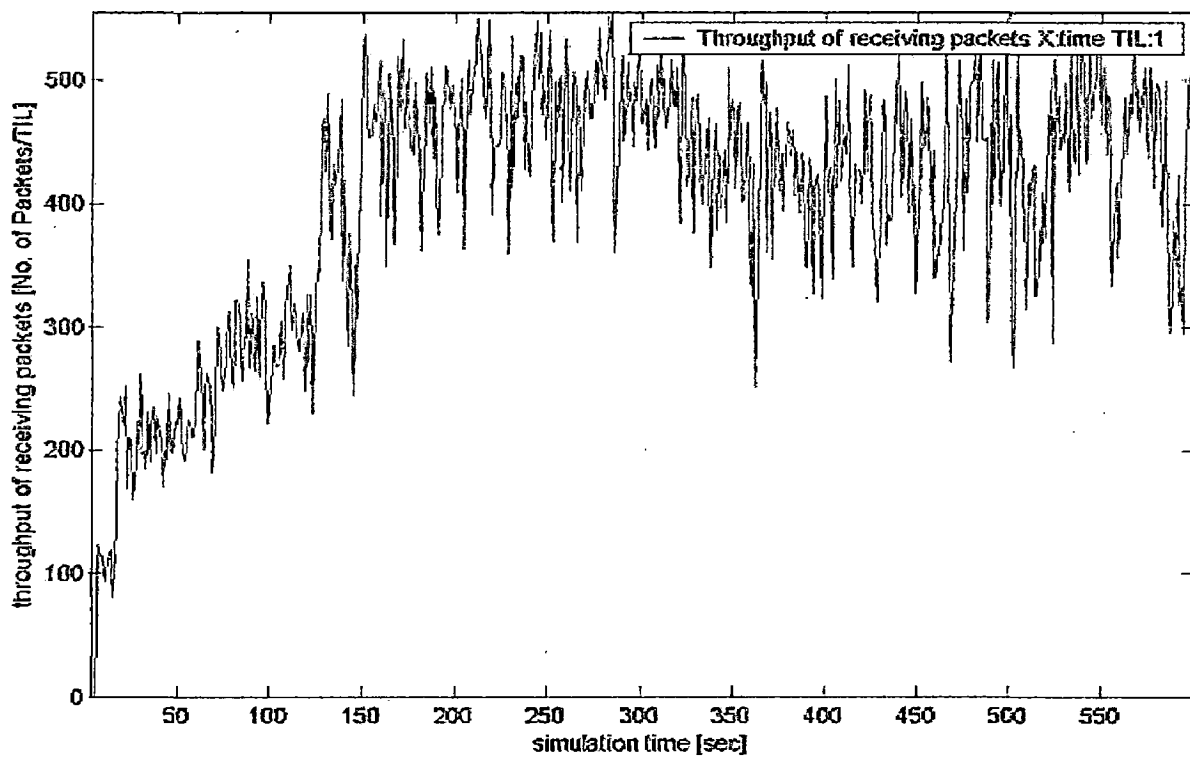


Figure 5.12 Simulation time vs Through put of receiving packets

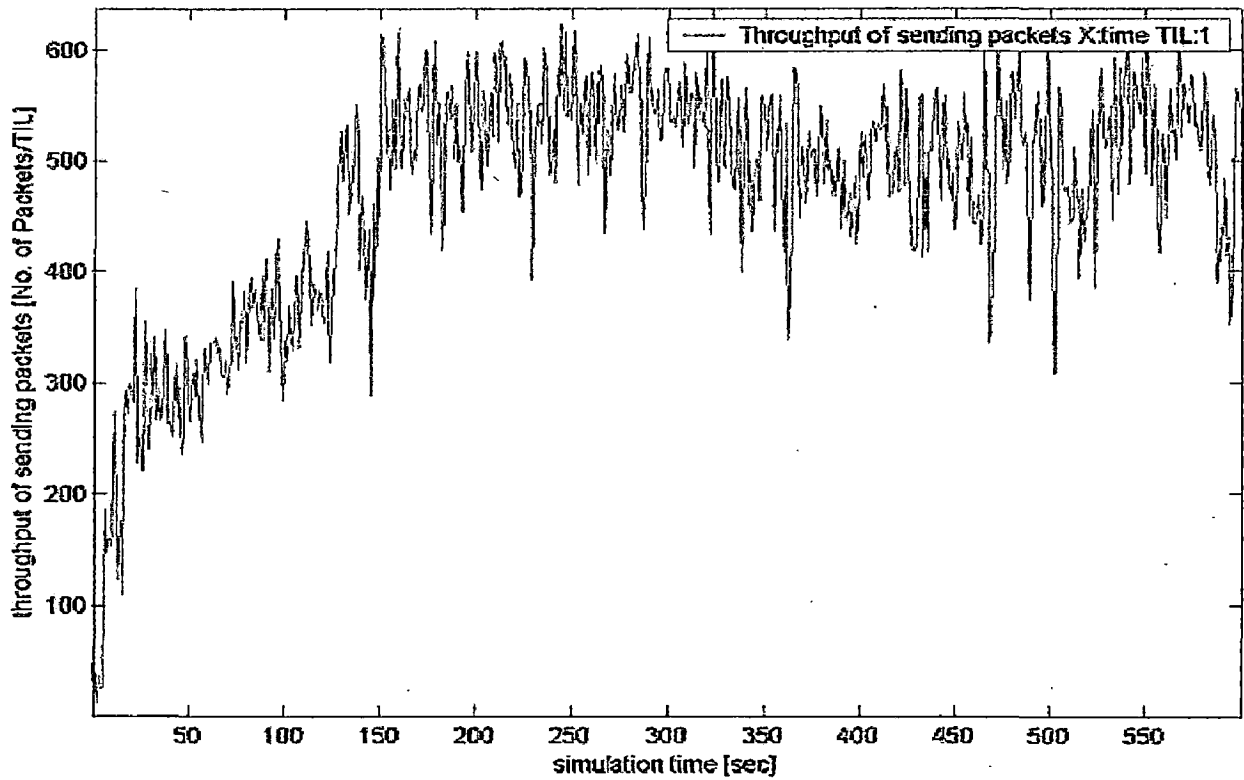


Figure 5.13 simulation time vs throughput of sending packets

The above figures 5.11,12,13, depicts the throughput of generated packets , receiving and sending packets .The through put of generating , receiving and sending packets increases as the simulation time progresses and has a little affect on the mobility of the nodes. This is due to availability of the routes at any Instant and up to date routing tables.

CHAPTER 6

CONCLUSIONS AND FUTURE WORK

6.1 Conclusions

Routing overhead which is a severe problem for link state routing in wireless networks, especially in large-scale wireless Ad hoc networks can be reduced using OLSR based Interval Adaptive Routing with efficient MPR's .

This Dissertation proposes an OLSR based Interval Adaptive Routing with efficient MPR's algorithm to minimize the routing overhead by dynamically maximizing the control message interval while satisfying the local network mobility, a mechanism to sense a mobile node using local link information, and provide a solution to enhance reliability of routes.

In this proposed OLSR based Interval Adaptive Routing with efficient MPR' s routing protocol. If a node becomes mobile the designed algorithm enables the network to access Efficient MPRs set. This approach to the mobile node adds a degree of reliability to routes. The data packets can be directly transmitted without incurring any delay because of redundant and available MPR.

Thus the proposed Routing algorithm analysis and simulation demonstrate that algorithm is effective outperforms OLSR and AODV routing protocols.

6.2 Future Works

The OLSR based Interval Adaptive Routing with efficient MPR's can be merged with the FSR and the other routing protocols Techniques. This approach is useful in VANET's

REFERENCES

- [1] M.S. Corson and J.Macker (Jan 1999) "Mobile ad hoc networking (MANET) routing protocol performances issues and evaluation considerations", RFC 2501.
- [2] P. Jacquet, A. Laouiti, P. Minet, and L. Viennot, "Performance of multipoint relaying in ad hoc mobile routing protocols," Proc. of IFIPE *Networking 2002, pages 387-398, 2002*
- [3] T. Clausen and P. Jacquet, "Optimized Link State Routing Protocol: (OLSR)," RFC 3626. 2003.
- [4] W. Jiang, Y. Zhu, and Z. Zhang "Routing overhead minimization in Large scale wireless sensor networks" IEEE 2007 pages 1270-1274, *2007*
- [5] Z.J.Haas , M . R .Pearlman , and P .Samar "The Zone Routing Protocol for MANET draft-ietf-manet-zone-zrp-04.txt" 2004
- [6] C. Perkins, E. B.Royer, and S. Das, "Ad hoc On-demand Distance Vector (AODV) Routing," rfc 3561 July 2003
- [7] M. Gerla, X. Hong, and G. Pei, "Fisheye State Routing Protocol (FSR) for Ad hoc networks ," available at pub/id/draft-ietf-manet-fsr-03.txt, 2002
- [8] R. Draves, J. Padhye, and B. Zill, "Routing in multi-radio, multi-hop wireless mesh networks," IEEE Proc. of MobiCom 2004
- [9] A Laouiti, A Qayyum and L Viennot (2001) Multipoint relaying: "An efficient Technique for flooding in mobile wireless networks", Proc. in 35th Annual Hawaii International conference on System Sciences (HICSS'2001), IEEE Computer Society 2001 , *pages 1-19, 2001* .

-
- [10] J. P. Singh, N. Bambos "Proposal and Demonstration of Link Connectivity Assessment based Enhancements to Routing in Mobile Ad-hoc Networks" IEEE edition pages 2834-2838 March 2003
- [11] A.Y.Khan, S. Rashid , A. Iqbal "Mobility vs. Predictive MPR Selection for Mobile Ad Hoc Networks using OLSR" IEEE 2005 International Conference on Emerging Technologies . pages 52-57. 2005
- [12] T. Camp, J. Boleng, V. Davies, A survey of mobility for ad hoc network research, *Wireless Communication & Mobile Computing (WCMC): Special issue on Mobile Ad Hoc Networking: Research, Trends and Applications*, pages. 483- 502, 2002.
- [13] G. Ying, T. Kunz, and L. Lamont, "Quality of service routing in ad-hoc networks using OLSR," *Proceedings of 36'th Hawaii International Conference on System Sciences*, Jan, 2003.
- [14] T. Camp, J. Boleng, V. Davies, Mobility models for ad hoc network simulations, *Wireless Comm. and Mobile Computing (WCMC)*, special issue on mobile ad hoc networking, vol. 2, no. 5, pages. 483-502, 2002.
- [15] A. Jardos, E.M. Belding-Royer, K. Almeroth, S. Suri, Towards realistic mobility models for mobile ad hoc networks, *Proceedings of ACM MobiCom*, pages. 217-229, 2003.
- [16] D. M. Blough, M. Leoncini, G. Resta, and P. Santi, "The lit k-neigh protocol for symmetric topology control in ad hoc networks," in *MobiHoc '03: Proceedings of the 4th ACM international symposium on Mobile ad hoc networking & computing*. New York, NY, USA: ACM Press, , pages. 141–152 , 2003
- [17] NS-2. available at <http://www.isi.edu/nsnam/ns/>

- [18] M. Greis. Tutorial for the Network Simulator "*ns*". Available at www.isi.edu/nsnam/ns/tutorial/index.html.

- [19] The VINT Project. The ns Manual, December 2003. <http://www.isi.edu/nsnam/ns/ns-documentation.html>.