INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps.

ProQuest Information and Learning
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA
800-521-0600

UMI®

Performance Analysis of Optimized Link State Routing (OLSR) Protocol

Yubo Liu

A Thesis

in

The Department

of

Electrical and Computer Engineering

Presented in Partial Fulfillment of the Requirements
for the Degree of Master of Applied Science at
Concordia University
Montréal, Québec, Canada

September 2002

© Yubo Liu, 2002

National Library of Canada
Acquisitions and Bibliographic Services
395 Wellington Street
Ottawa ON K1A 0N4
Canada

Bibliothèque nationale du Canada
Acquisitions et services bibliographiques
395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence

Our file Notre référence

The author has granted a non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of this thesis in microform, paper or electronic formats.

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de cette thèse sous la forme de microfiche/film, de reproduction sur papier ou sur format électronique.

The author retains ownership of the copyright in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

L'auteur conserve la propriété du droit d'auteur qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

0-612-72912-5

Canada

ABSTRACT

Performance Analysis of Optimized Link State Routing (OLSR) Protocol

Yubo Liu

The mobile ad-hoc network (MANET) is referred to as autonomous system of mobile routers, connected by wireless links. MANET can be applied when networks need to be rapidly deployable, without prior planning or any existing infrastructure.

OLSR protocol is an optimization of the pure link state protocol for mobile ad-hoc networks. It concentrates on functions at the IP layer while any protocol such as IEEE 802.11 can be used in the underlying MAC layer. Multipoint Relays (MPR) are introduced so as to minimize flooding of control messages. This technique significantly reduces the number of retransmission and network congestion.

In this work we emphasize the performance analysis of OLSR. A detailed protocol simulation model is established in C++. The main performance criteria are throughput, overhead, buffer overflow, transfer delay, etc. We give herein detailed evaluation and explanation of the simulation results.

ACKNOWLEDGEMENTS

First of all, I would like to express my deepest grief to my father who passed during my thesis. I lost the most beloved person in the world. No word can express my heart. But one thing is sure that he will be in my heart forever.

Secondly, I am overwhelmed with gratitude to my supervisor Dr. A. K. Elhakeem. He gives me the best advice and support for my thesis. He encourages me to do the perfect work.

At last, I am very happy and proud to mention my wife and my lovely daughter. Without their support, I can't fulfill my thesis.

Thanks for all my best friends who gave me kindly help.

List of Acronyms and Abbreviations

ABF	Average Buffer overflow probability
ACK	Acknowledgement
ADTD	Average of Data Transfer Delay
AHC	Average Hop Counter
AODV	Ad hoc On Demand Distance Vector Routing
AP	Access Point
APB	Average of Packets in Buffer
AQD	Average of Queuing Delay
BBIR	Base Band Infra Red
BEB	Binary Exponential Back off
BRP	Bordercast Resolution Protocol
BSS	Basic Service Set
CLR	Clear
CPU	Central Process Unit
CRC	Cyclic Redundancy Check
CSMA/CA	Carrier Sense Multiple Access with Collision Avoidance
CTS	Confirm To Send
DECT	Digital European Cordless Telecommunications
DIFS	Distributed Inter Frame Space
DS	Distributed System
DSDV	Destination Sequenced Distance Vector
DSSS	Direct Sequence Spread Spectrum
DSR	Dynamic Source Routing
ESS	Extended Service Set
ETSI	European telecommunication Standards Institute
EY-NPMA	Elimination Yield Non-pre-emptive Priority Multiple Access
FDMA	Frequently Division Multiple Access
FHSS	Frequency Hopping Spread Spectrum
FIFO	First In First Out
FSR	Fisheye State Routing

GMSK	Gaussian Minimum Shift keying
GSM	Global System for Mobile communication
HID	Hierarchical ID
HIPERLAN	High PErformance Radio Local Area Network
HSR	Hierarchical State Routing
IARP	Intra-zone Routing Protocol
IERP	Inter-zone Routing Protocol
IBSS	Independent Basic Service Set
IEEE	Institute of Electrical and Electronic Engineering
IETF	Internet Engineering Task Force
IP	Internet Protocol
ISM	Industrial, Scientific, Medical
IR	InfraRed
ITU	International Telecommunications Union
LAN	Local Area Network
LS	Link State
LSA	Link State Advertisement
MAC	Medium Access Control
MANET	Mobile Ad-hoc Networks
MPR	Multi-Point Relay
MPR_S	Multi-Point Relay Selector
MSDU	MAC Service Data Unit
MSSN	Multipoint relay Selector Sequence Number
OLSR	Optimized Link State Routing
PDA	Personal Digital Assistant
PHY	Physical
QOS	Quality of Service
QRY	Query
RREP	Route Reply
RREQ	Route Request
RTS	Request To Send
SIFS	Short Inter Frame Space
SS	Stations Service

STA	Stations
TBRPF	Topology Broadcast based on Reverse-Path Forwarding
TC	Topology Control
TDMA	Time Division Multiple Access
TORA	Temporally-Odered Routing Algorithm
UPD	Update
VBF	Variance of Buffer overflow probability
VDTD	Variance of Data Transfer Delay
VHC	Variance of Hop Counter
VPB	Variance of Packets in buffers
VQD	Variance of Queuing Delay
WLAN	Wireless Local Area Networks
ZRP	Zone Routing Protocol

Table of Contents

List of Acronyms and Abbreviations v
Table of Contents viii
List of Figures xi

1 Introduction 1
1.1 Wireless data network 1
1.1.1 General 1
1.1.2 Usage 4
1.1.3 Characteristics 4
1.1.4 MANET working group 5
1.2 Wireless data network standards 6
1.2.1 Network standard category 6
1.2.1.1 TDMA type networks 7
1.2.1.2 CSMA type networks 7
1.2.1.2.1 IEEE Standards 8
1.2.1.2.2 ETSI Standards 9
1.2.2 IEEE 802.11 standard overview 9
1.2.3 ETSI's HIPERLAN type 1 standard overview 16
1.2.3.1 HIPERLAN's random access protocol with priorities 17
1.2.3.2 Routing in HIPERLAN 18

2 Mobile Ad-hoc (MANET) routing protocols 20
2.1 Conventional protocols 20
2.1.1 Link State 21
2.1.2 Distance Vector 22
2.1.3 Source Routing 22
2.1.4 Flooding 22
2.2 Desirable properties 23
2.3 MANET routing protocols 25
2.4 Fisheye State Routing Protocol (FSR) for Ad Hoc Networks 25
2.5 Ad Hoc On Demand Distance Vector (AODV) Routing 28
2.5.1 Description 28
2.5.2 Properties 31
2.6 Dynamic Source Routing Protocol (DSR) 31
2.6.1 Description 31
2.6.2 Properties 33
2.7 Zone Routing Protocol (ZRP) 34
2.7.1 Description 34
2.7.2 Properties 36
2.8 Temporally-Ordered Routing Algorithm (TORA) 37

2.8.1 Description 37
2.8.2 Properties 38
2.9 Hierarchical State Routing (HSR) 39
2.10 Destination Sequenced Distance Vector(DSDV) 42
2.10.1 Description 42
2.10.2 Properties 43
2.11 IP Flooding in Ad hoc Mobile Networks 43
2.12 Topology Broadcast based on Reverse-Path Forwarding (TBRPF) 44
2.13 Optimized Link State Routing Protocol 45
2.13.1 Multi Point Relay 46
2.13.2 Routing 48
2.14 Some performance results 48

3 OLSR simulation model 53
3.1 Input-Output Parameter and Definition 53
3.1.1 Input Parameter and Definition 53
3.1.2 Output Parameters and Definitions 55
3.1.2.1 Average and variance of data transfer delay 55
3.1.2.2 Average and variance of queuing delay 56
3.1.2.3 Average and variance of hop counter 56
3.1.2.4 Average and variance of buffer overflow probability 57
3.1.2.5 Average and variance of number of messages in the buffer 58
3.1.2.6 Throughput 59
3.1.2.7 Generation overhead 59
3.1.2.8 Transmission overhead 59
3.2 Simulation procedure 60
3.2.1 Simulation assumptions 60
3.2.2 Simulation data structure 61
3.2.2.1 Message data structure 61
3.2.2.2 Node data structure 63
3.2.2.3 Table data structure 64
3.2.2.3.1 Neighbor table 64
3.2.2.3.2 2-hop neighbor table 64
3.2.2.3.3 MPR table 65
3.2.2.3.4 MPR Selector table 65
3.2.2.3.5 Topology table 66
3.2.2.3.6 Routing table 67
3.2.3 Simulation model description 68
3.2.3.1 Network generator 68
3.2.3.2 Message generating 68
3.2.3.2.1 Hello generating 68
3.2.3.2.2 TC generating 68
3.2.3.3 Neighbor sensing 69
3.2.3.4 Multipoint relay selection 69
3.2.3.5 Multipoint relay information declaration 70
3.2.3.6 Routing table calculation 71

4 OLSR simulation results and evaluation 73
4.1 Network performance under different sending control message period 73
4.2 Network performance under different data time out 79
4.3 Network performance under different entry hold time 84
4.4 Network performance under different load condition 89
4.5 Network performance under different data transfer number of hops 93
4.6 Network performance under different node move probability 97
4.7 Performance comparison of OLSR with other routing protocols 101

5 Conclusions and future work 103

Reference

List of Figures

Figure 1-1 Example of a simple ad-hoc network with three participating nodes. 3
Figure 1-2 Block diagram of a mobile node acting both as host and as router. 3
Figure 1-3 Block diagram of a multi-interface node acting both as hosts and as router..... 4
Figure 1-4 IEEE 802.11 MAC frame format 10
Figure 1-5 IEEE 802.11 MAC frame---frame control 10
Figure 1-6 Complete IEEE 802.11 architecture 11
Figure 1-7 Data transmission and ACK in IEEE 802.11 without RTS/CTS 14
Figure 1-8 Data transmission and ACK in IEEE 802.11 with RTS/CTS 14
Figure 1-9 Hidden node problem 16
Figure 1-10 Messages with shorter deadlines have higher priority..... 18
Figure 1-11 EY-NPMA in HIPERLAN 18
Figure 1-12 Hop by hop forwarding in HIPERLAN..... 19
Figure 2-1 AODV Messages. 30
Figure 2-2 Example network of ZRP. 36
Figure 2-3 Directed acyclic graph rooted at destination. 38
Figure 2-4 An example of physical/virtual clustering 41
Figure 2-5 A small network with full flooding and MPR flooding. 48
Figure 2-6 Control O/H versus traffic pairs fixed area 50
Figure 2-7 Control O/H Versus mobility (100 pairs) 51
Figure 2-8 Average delay versus mobility (100 pairs) 51
Figure 2-9 Average hops versus mobility (100 Pairs) 52
Figure 2-10 Control O/H versus number of nodes.....52
Figure 3-1 Example of different addresses 61
Figure 4-1 buffer full probability—different sending control message period..... 74
Figure 4-2 messages in buffer—different sending control message period..... 74
Figure 4-3 data transfer hops—different sending control message period 76
Figure 4-4 throughput and overhead—different sending control message period..... 76
Figure 4-5 data message queue delay—different sending control message period 78
Figure 4-6 successful data transfer delay—different sending control message period..... 78
Figure 4-7 buffer full probability—different data time out..... 80
Figure 4-8 messages in buffer—different data time out 80
Figure 4-9 data transfer hops—different data time out..... 81
Figure 4-10 throughput and overhead—different data time out 81
Figure 4-11 data queue delay—different data time out..... 83
Figure 4-12 successful data transfer delay—different data time out..... 83
Figure 4-13 buffer full probability—different entry hold time 85
Figure 4-14 messages in buffer—different entry hold time..... 85
Figure 4-15 data transfer hops—different entry hold time..... 86
Figure 4-16 throughput and overhead—different entry hold time 86
Figure 4-17 data queue delay—different entry hold time 88
Figure 4-18 successful data transfer delay—different entry hold time 88
Figure 4-19 buffer full probability—different load condition 90

Figure 4-20 messages in buffer—different load condition 90
Figure 4-21 data transfer hops—different load condition 91
Figure 4-22 throughput and overhead—different load condition 91
Figure 4-23 data queuing delay—different load condition 92
Figure 4-24 successful data transfer delay—different load condition 92
Figure 4-25 buffer full probability—different data transfer Max hop 94
Figure 4-26 messages in buffer—different data transfer Max hop 94
Figure 4-27 data transfer hops—different data transfer Max hop 95
Figure 4-28 throughput and overhead-- different data transfer Max hop 95
Figure 4-29 data queuing delay-- different data transfer Max hop 96
Figure 4-30 successful data transfer delay-- different data transfer Max hop 96
Figure 4-31 buffer full probability—different node move condition 98
Figure 4-32 number of messages in buffer—different node move condition 98
Figure 4-33 data transfer hops—different node move condition 99
Figure 4-34 throughput and overhead—different node move condition 99
Figure 4-35 data queuing delay—different node move condition 100
Figure 4-36 successful data transfer delay—different node move condition 100

1 Introduction

Optimized Link State Routing Protocol (OLSR) is developed for ad hoc networks. It's an optimization or improvement of the conventional link state routing protocol. Because it's a relatively new developed routing technique, the performance has not been analyzed thoroughly. In my thesis, I establish a detailed simulation model and analyze the network performance such as throughput, overhead, data transfer delay and buffer overflow probability, etc.

The thesis is organized as follows:

In Chapter 1, I start with the introduction to the wireless data networks and some standards. I split the standards into two categories: control based schemes and random access schemes. I discuss two important standards using random access schemes.

In Chapter 2, I present the major routing protocols for ad hoc networks. The previous results are also presented related to the performance analysis.

In Chapter 3, I give the detailed input and output parameters and definitions. The simulation assumption, modeling and procedure are also described in detail.

In Chapter 4, I evaluate the performance of the proposed protocol on the basis of the simulation results obtained in different conditions by varying input parameters.

In Chapter 5, I briefly give the conclusions and some suggestions for future work.

1.1 Wireless data network

1.1.1 General

Wireless network is an emerging new technology that will allow users to access information and services electronically, regardless of their geographic position. Wireless

networks can be classified in two types: infrastructure network and infrastructure-less (ad hoc) networks. Infrastructure network consists of a network with fixed and wired gateways. A mobile host communicates with a bridge in the network (called base station) within its communication radius. The mobile unit can move geographically while it is communicating. When it goes out of range of one base station, it connects with a new base station and starts communicating through it. This is called handoff. In this approach the base stations are fixed.

In contrast to infrastructure based networks, in ad hoc networks all nodes are mobile and can be connected dynamically in an arbitrary manner. All nodes of these networks behave as routers and take part in discovery and maintenance of routes to other nodes in the network. Ad hoc networks are very useful in emergency search-and-rescue operations, meetings or conventions in which persons wish to quickly share information, and data acquisition operations in inhospitable terrain.

A wireless ad-hoc network is a collection of mobile/semi-mobile nodes with no pre-established infrastructure, forming a temporary network. Each of the nodes has a wireless interface and communicate with each other over either radio or infrared. Laptop computers and personal digital assistants that communicate directly with each other are some examples of nodes in an ad-hoc network. Nodes in the ad-hoc network are often mobile, but can also consist of stationary nodes, such as access points to the Internet. Semi mobile nodes can be used to deploy relay points in areas where relay points might be needed temporarily.

Fig. 1-1 shows a simple ad-hoc network with three nodes. The outermost nodes are not within transmitter range of each other. However the middle node can be used to forward

messages between the outermost nodes. The middle node is acting as a router and the three nodes have formed an ad-hoc network.

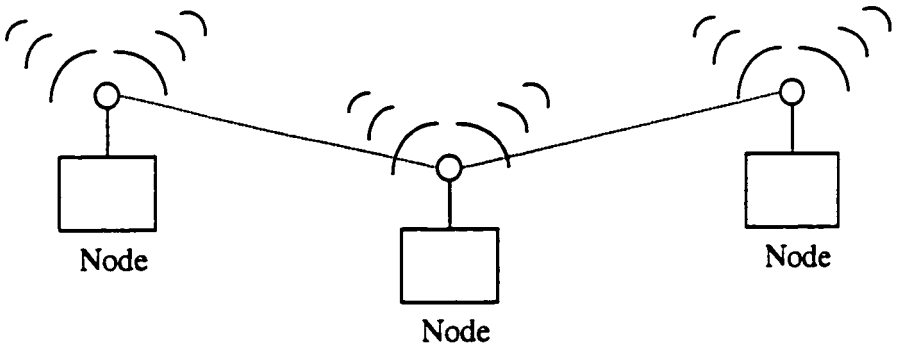


Figure 1-1 Example of a simple ad-hoc network with three participating nodes.

An ad-hoc network uses no centralized administration. This is to be sure that the network won't collapse just because one of the mobile nodes moves out of transmitter range of the others. Nodes should be able to enter/leave the network as they wish. Because of the limited transmitter range of the nodes, multiple hops may be needed to reach other nodes. Every node wishing to participate in an ad-hoc network must be willing to forward messages for other nodes. Thus every node acts both as a host and as a router. A router is an entity, which, among other things runs a routing protocol. A mobile host is simply an IP-addressable host/entity in the traditional sense (Fig. 1-2). If the node has multi-interfaces, the node can be viewed as an abstract entity consisting of a router and a set of affiliated mobile hosts (Fig. 1-3).

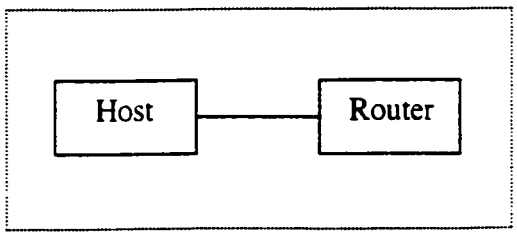


Figure 1-2 Block diagram of a mobile node acting both as host and as router.

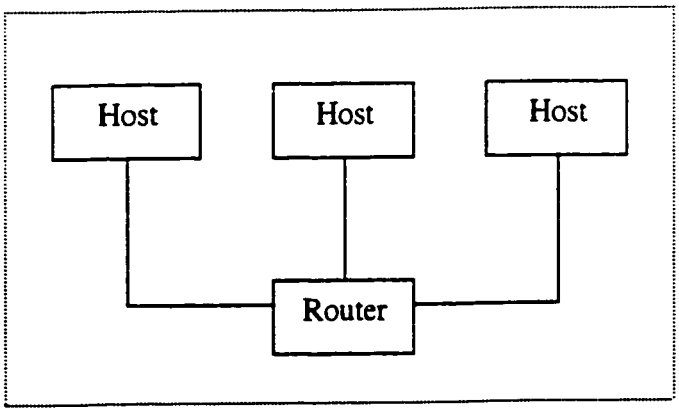


Figure 1-3 Block diagram of a multi-interface node acting both as hosts and as router.

1.1.2 Usage

The picture of what these kinds of networks will be used for is getting clearer. Suggestions vary from documents sharing at conferences to infrastructure enhancements and military applications, search and rescue, mining, camps, optimized introduction processes.

In areas where no infrastructure such as the Internet is available an ad-hoc network could be used by a group of wireless mobile hosts. This can be the case in areas where a network infrastructure may be undesirable due to reasons such as cost or convenience. Examples of such situations include disaster recovery, personnel or military troops in cases where the normal infrastructure is either unavailable or destroyed. Other examples include business associates wishing to share files in an airport terminal, or a class of students needing to interact during a lecture.

1.1.3 Characteristics

The communication environment in mobile ad hoc networks is very different from that of fixed networks. MANET allows a large range of user mobility, with high user density. Different and differing radio propagation conditions throughout the network can make the

routing task more difficult and challenging. Intermittent node connectivity is also to be considered while designing a routing protocol. As wireless links are particularly vulnerable to eavesdropping and other attacks, MANET protocols require some security mechanisms, either designed in the routing protocol itself or provided by some lower layer protocol.

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

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

1.1.4 MANET working group

IETF has a working group named MANET (Mobile Ad-hoc Networks) that is working in the field of ad-hoc networks [15].

The primary focus of the working group is to develop and evolve MANET routing specification(s) and introduce them to the Internet Standards track. The goal is to support networks scaling up to hundreds of routers. If this proves successful, future work may include development of other protocols to support additional routing functionality. The working group will also serve as a meeting place and forum for those developing and experimenting with MANET approaches.

The main task of MANET working group at IETF is to propose and specify the routing protocols for ad hoc networks. These protocols may use any underlying MAC layer such as IEEE 802.11 or EY-NPMA. MANET working group is only concerned with routing protocols at the IP layer. In this way, these protocols are different from the HIPERLAN protocol, where dynamic routing is performed at the MAC layer. In addition to unicast routing, other optional functionalities may be developed for the MANET protocols, such as multicast, broadcast, authentication, quality of service, power conservation, etc.

The working group will examine related security issues around MANET. It will consider the intended usage environments, and the threats that are (or are not) meaningful within that environment.

1.2 Wireless data network standards

1.2.1 Network standard category

Many types of data network standards exist in the wireless world. We can categorize these standards according to the channel access schemes they use. The first type of standards fall in the category, which uses a control based scheme for the channel access, for example TDMA (Time Division Multiple Access) or FDMA (Frequency Division Multiple Access). The second type of standards are those which use random access

protocols like CSMA [21], CSMA with collision avoidance [22], or CSMA with active signaling, for example EYNPMA (Elimination Yield Non-pre-emptive Priority Multiple Access) [10]. We will briefly look at these two main types of standards in the following sections.

1.2.1.1 TDMA type networks

TDMA (or FDMA) type of networks uses a centralized control scheme for channel access. This scheme requires a central entity controlling the network which has access to all resources and which is distributing these resources fairly among network nodes according to their needs. In this scheme, the active nodes have some type of 'resource reservation' either in time or in frequency, for a certain time. Some of the standards, which use TDMA scheme, are:

- Pagers
- DECT (Digital European Cordless Telecommunications)
- GSM (Global System for Mobile communications)

1.2.1.2 CSMA type networks

The second type of data network standards uses random access protocols. Wireless networks which use a random access scheme for accessing the common shared medium generally employ a CSMA (Carrier Sense Multiple Access) technique or its variants such as CSMA with collision avoidance (CSMA/CA), or CSMA with active signaling, for example EY-NPMA (Elimination Yield Non-pre-emptive Priority Multiple Access). A random access protocol does not require a central entity and therefore the control is completely distributed. All the nodes, sharing the same frequency band, contend for

channel access and the winner of the contention process transmits its packet. At the end of the packet transmission, the contention process restarts among the contending nodes.

Two important, existing standards of CSMA type networks are:

- IEEE 802.11 (900 MHz, 2.4 GHz & 5.2-5.85 GHz)
- ETSI ETS 300-652 HIPERLAN type 1 (5.2-5.35GHz)

1.2.1.2.1 IEEE Standards

The most common standards in use are those made by IEEE (Institute of Electrical and Electronics Engineering). Its main standards for wireless data networks are:

- IEEE 802.11 [19]. It permits three different technologies: FHSS (Frequency Hoping Spread Spectrum), DSSS (Direct Sequence Spread Spectrum) and BBIR (Base Band Infra Red) at the physical layer in the same network. The standard restricts the emitted power to 1W and uses the ISM (industrial, scientific, and medical) frequency band of 900 MHz, 2.4 GHz and 5.7 GHz. IEEE 802.11 has its own standardized MAC protocol, and it permits only a restricted set of topologies (having a central base station). It has a maximum throughput of 11 Mbps.
- IEEE 802.11a [17]. It is designed as an extension to the basic standard, for high-speed operation. It operates on 5 GHz frequency band and has a maximum throughput of 56 Mbps.
- IEEE 802.11b [18]. It is also designed as an extension to the basic standard, for high-speed operation. It operates on 2.4 GHz band and has a maximum throughput of 11 Mbps.

1.2.1.2.2 ETSI Standards

ETSI (European Telecommunication Standards Institute) is the main organization that formulates the telecommunication standards in Europe. Its main standards for the wireless data networks are:

- ETS 300-328 standard, which uses 2.4 GHz ISM band. It provides a maximum throughput of 1-2 Mbps, with a restriction of low emitted power. No MAC protocol is standardized.
- HIPERLAN type 1 standard [8] uses the 5.2 GHz bands (5.15-5.35 GHz) that are reserved for this application. It provides a maximum throughput of up to 23 Mbps. This radio network protocol provides a support for the multimedia transmission through its prioritized channel access protocol and performs dynamic routing at the MAC layer.
- HIPERLAN type 2 standard [9] uses the 5.47-5.725 GHz bands that are reserved for it. It uses the base station topology as contrary to HIPERLAN type 1, which uses internal routing.
- Other standards which are under consideration in ETSI are: HIPERLAN type 3 ATM wireless access} and HIPERLAN type 4 (Radio local loop).

1.2.2 IEEE 802.11 standard overview

IEEE 802.11 [1] is one of the most important standards of the market. It uses the ISM shared bands of 900 MHz, 2.4 GHz and 5.7 GHZ.

It provides a maximum throughput of 11 to 56 Mbps with a radio range of about 100 m [11]. It permits the network architectures with base station. It has a single MAC protocol,

supporting three different physical layer technologies (FHSS, DSSS and BBIR), possibly co-existing in the same network. Its QoS mechanism is based on polling the nodes for real time transmission. The MAC frame format in IEEE 802.11 [13] is as shown in Figs. 1-4,1-5.

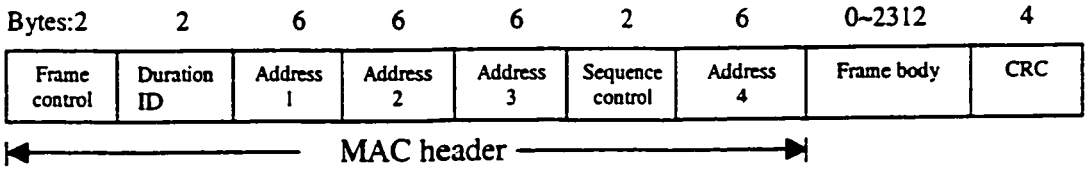


Figure 1-4 IEEE 802.11 MAC frame format

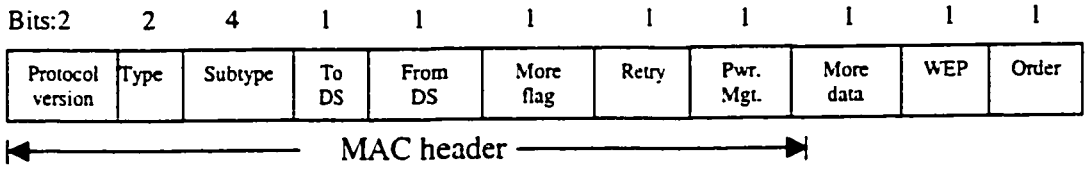


Figure 1-5 IEEE 802.11 MAC frame---frame control

The protocol version field carries the version of the 802.11 standard. Type and subtype fields determine the function of frame. To DS and From DS field specifies the direction a frame enters or exits the distributed system. More flag deal with the fragmentation of the frame. Retry is set to 1 if this frame is a retransmission. Power management indicates the encryption algorithm. Order field indicates whether the frame must be strictly ordered.

The frame body is a variable-length field consisting of the data payload and 7 octets for encryption. The 6-octet address fields are used to identify the basic service set, the destination, the source address, and the receiver and transmitter addresses. Duration field indicates the time the channel will be allocated for successful transmission of a frame. CRC field is for error detection.

The complete IEEE 802.11 architecture is shown in Fig. 1-6. The *basic service set* (BSS) is the basic building block of an IEEE 802.11 LAN and this consists of devices referred to as *stations* (STA). Basically, the set of STAs that can talk to each other can form a BSS. Multiple BSSs are interconnected through an architectural component, called *distribution system* (DS), to form an *extended service set* (ESS). An *access point* (AP) is a STA that provides access to DS by providing DS services. If the LAN just consists of single BSS, then that is called *independent BSS* (IBSS). This IBSS is often referred to *ad hoc network* for it does not require any pre-planning.

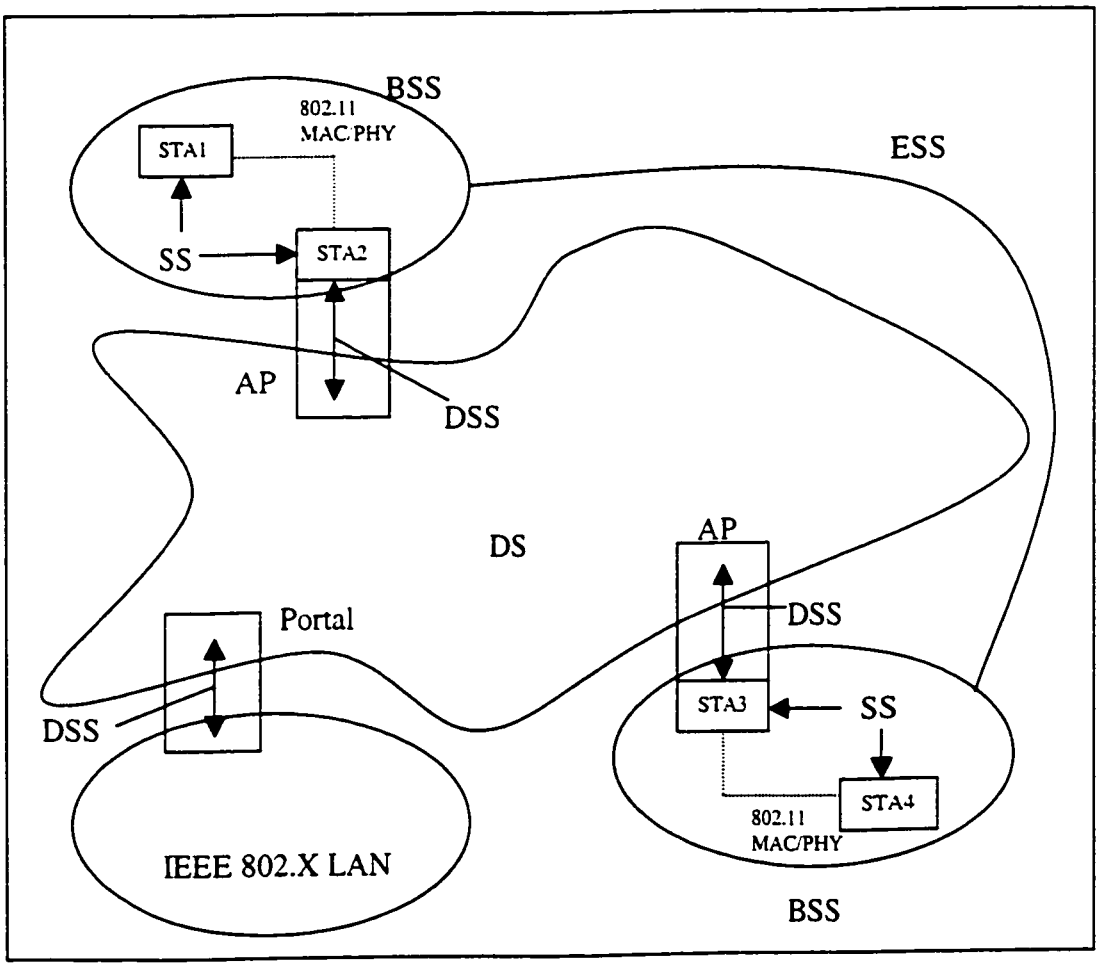


Figure 1-6 Complete IEEE 802.11 architecture

IEEE 802.11 specifies the services that are to be provided by STAs and DS. It does not specify how DS is to be implemented; it can be either distributed or central. A service provided by a station is called *station service* (SS) and a service by a DS is called *distributed system service* (DSS).

A station willing to join an ESS first chooses an AP from all available APs or can start a new BSS by assuming the role of an AP. The available APs are found out by looking for the periodic beacons they broadcast over the medium. A station upon choosing an AP will authenticate itself to AP and then associates with it. This association basically provides this STA to AP mapping information to the DS, which is utilized in routing the messages in the ESS.

The IEEE 802.11 standard places specifications on the parameters of both the physical (PHY) and medium access control (MAC) layers of the network. The PHY layer, which actually handles the transmission of data between nodes, can use either direct sequence spread spectrum, frequency hopping spread spectrum, or infrared (IR) pulse position modulation. IEEE 802.11 makes provisions for data rates of either 1 Mbps or 2 Mbps, and calls for operation in the 2.4 - 2.4835 GHz frequency band (in the case of spread-spectrum transmission), which is an unlicensed band for industrial, scientific, and medical (ISM) applications, and 300 - 428,000 GHz for IR transmission. Infrared is generally considered to be more secure to eavesdropping, because IR transmissions require absolute line-of-sight links (no transmission is possible outside any simply connected space or around corners), as opposed to radio frequency transmissions, which can penetrate walls and be intercepted by third parties unknowingly. However, infrared transmissions can be

adversely affected by sunlight, and the spread-spectrum protocol of 802.11 does provide some rudimentary security for typical data transfers.

The MAC layer is a set of protocols that is responsible for maintaining order in the use of a shared medium. The 802.11 standard specifies a carrier sense multiple access with collision avoidance (CSMA/CA) protocol. In this protocol, when a node receives a packet to be transmitted, it first listens to ensure no other node is transmitting. If the channel is clear, it then transmits the packet. Otherwise, it chooses a random "back off factor" which determines the amount of time the node must wait until it is allowed to transmit its packet. During period in which the channel is clear, the transmitting node decreases its back off counter. (When the channel is busy it does not decrement its back off counter.) When the back off counter reaches zero, the node transmits the packet. Since the probability that two nodes will choose the same back off factor is small, collisions between messages are minimized. Collision detection, as is employed in Ethernet, cannot be used for the radio frequency transmissions of IEEE 802.11. The reason for this is that when a node is transmitting it cannot hear any other node in the system which may be transmitting, since its own signal will drown out any others arriving at the node.

IEEE 802.11 employs a random access scheme for its MAC protocol, with a combination of several techniques, like BEB (Binary Exponential Back off), Inter frame spacing, Acknowledgement, RTS/CTS and NAV vector, etc [19]. RTS/CTS mechanism allows for a more reliable transmission of unicast messages by blocking the transmission channel.

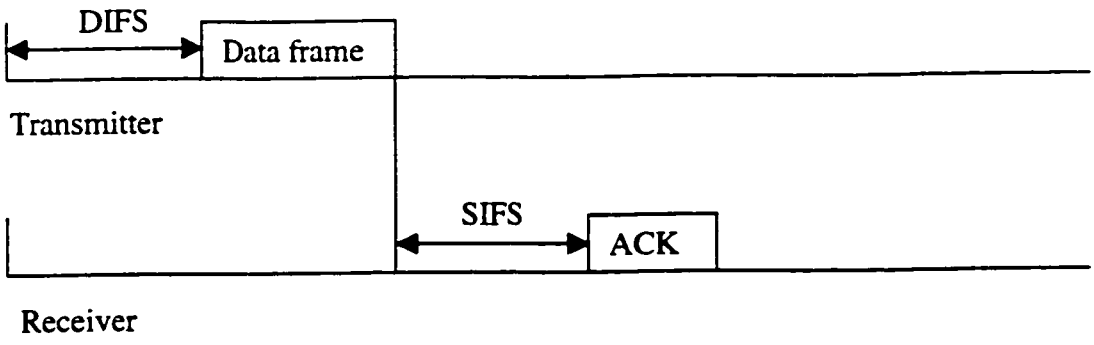


Figure 1-7 Data transmission and ACK in IEEE 802.11 without RTS/CTS

If the channel is sensed idle for an amount of time equal to or greater than the distributed inter frame space (DIFS), a station is then allowed to transmit [13]. When a intended receiving station has correctly and completely received a frame, it waits a time of short inter frame space (SIFS) and then sends an explicit acknowledgement frame back to the sender.

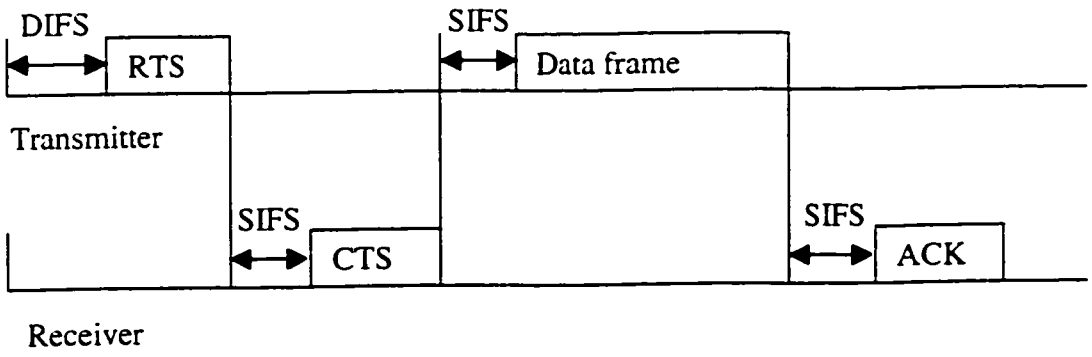


Figure 1-8 Data transmission and ACK in IEEE 802.11 with RTS/CTS

By using of Request To Send (RTS) and Clear To Send (CTS) frames, CSMA/CA can reserve access to the medium. RTS and CTS frames contains a duration field that defines the period of time that the medium is to be reserved for the transmission of the data frame and the returning ACK frame.

Priority access to the wireless medium is controlled through the use of inter frame space. Station only required to wait a SIFS before transmission have the highest priority access to the medium. Basic access requires stations to wait for DIFS idle time. The source station maintains control of the channel throughout the transmission of frame by using only an SIFS period after receiving an ACK and transmitting the next fragment.

Whenever a packet is to be transmitted, the transmitting node first sends out a short RTS packet containing information on the length of the packet. If the receiving node hears the RTS, it responds with a short CTS packet. After this exchange, the transmitting node sends its packet. When the packet is received successfully, as determined by a cyclic redundancy check (CRC), the receiving node transmits an acknowledgment (ACK) packet. This back-and-forth exchange is necessary to avoid the "hidden node" problem, illustrated in Fig. 1-9. As shown, station A and station B can both communicate directly with the access point; however, the barrier that represents lack of connectivity, prevents station A and B from communicating directly with each other. The problem is that a collision will occur when station A attempts to access the medium because it will not be able to detect that station B is already transmitting.

To guard against collisions based on hidden nodes and high utilization, the transmitting station B should send an RTS frame to the access point, requesting service for a certain amount of time. If the access point approves, it will broadcast a CTS frame announcing this time to all stations that hear the frame transmission. As a result, all stations, including station A, will not attempt to access the medium for the specified amount of time.

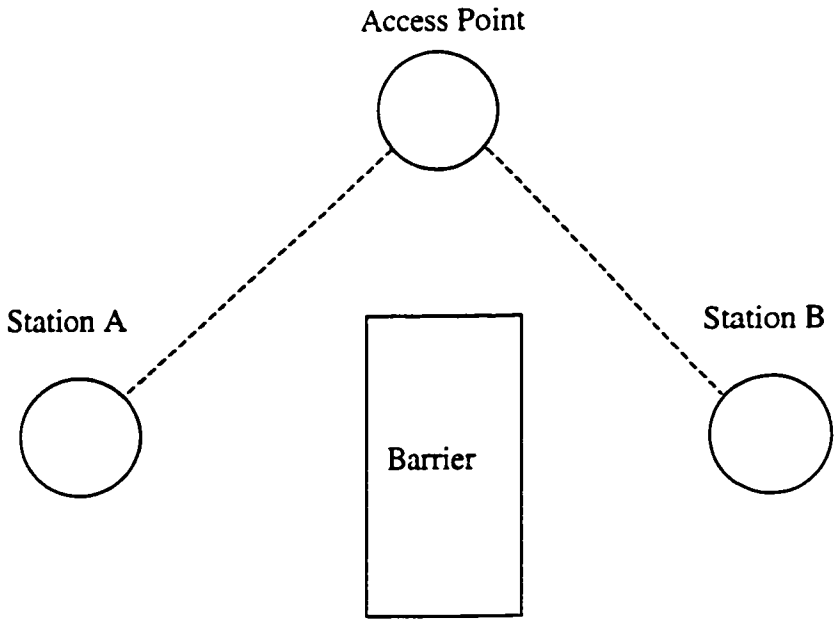


Figure 1-9 Hidden node problem

1.2.3 ETSI's HIPERLAN type 1 standard overview

HIPERLAN type 1 is an ETSI's standard for radio LANs. It uses the frequency band of 5.2 - 5.35 GHz, which is reserved for this purpose in Europe. It is a wireless-Ethernet type standard, providing a maximum throughput of up to 23 Mbps. At the physical layer, it employs the GMSK (Gaussian Minimum Shift Keying) technology with equalization. It has a completely distributed architecture with un-restricted topologies, i.e. without any base station requirement, and is specially adapted for mobile wireless networks. Through internal routing, a HIPERLAN network can cover a region larger than the radio range of individual nodes. Its mechanism of packet forwarding keeps the out of range nodes connected to the network. Mobility management is done by dynamic neighbor and routing tables with periodic update of the information in these tables. HIPERLAN has optional data encryption as well as a system of conservation of energy.

1.2.3.1 HIPERLAN's random access protocol with priorities

The HIPERLAN protocol has its own specific random access MAC protocol that provides prioritized access for multimedia transmission and hence ensures quality of service. Packet quality of service is expressed in terms of packet deadline and user priority. These priorities are dynamic and the packet gets higher priority as the deadline approaches (Fig. 1-10). Hence, shorter deadlines map into higher priorities for the packet. The random access mechanism is based on CSMA technique with addition of a non-data signal burst before the packet and an acknowledgement after the packet. The technique is called EY-NPMA (Elimination Yield Non-pre-emptive Priority Multiple Access) [8] and is illustrated in Fig. 1-11. In EY-NPMA, the contention process starts with the priority phase to handle the multimedia traffic. All the contending nodes take a small back off according to their priorities. During this back off period, a node keeps sensing the channel. If it senses the channel busy it quits the contention. The "assertion slot" is to indicate the success. After priority phase is the elimination phase, in which the contending nodes send a burst of random length to eliminate the other contending nodes. Only the nodes that send the longest burst will win and stay in the contending process. The yield period is to select a single contender. Each surviving node chooses a random number of yield slots before starting the transmission. The node having the smallest yield period is the final winner, which starts sending the actual packet right after this yield period. All the remaining contending nodes sense this transmission and therefore quit the contention.

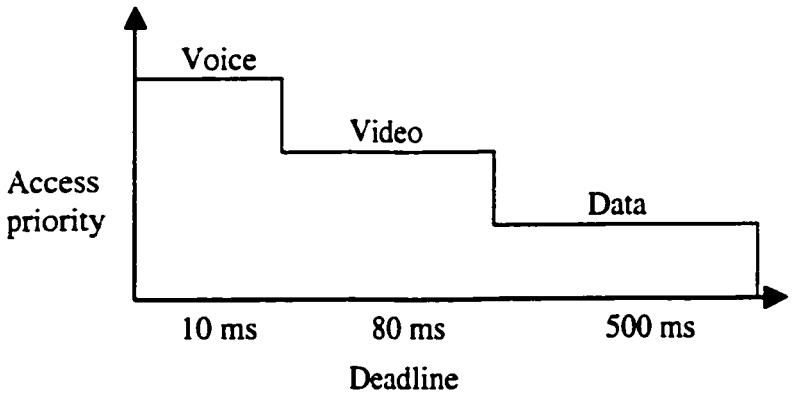


Figure 1-10 Messages with shorter deadlines have higher priority

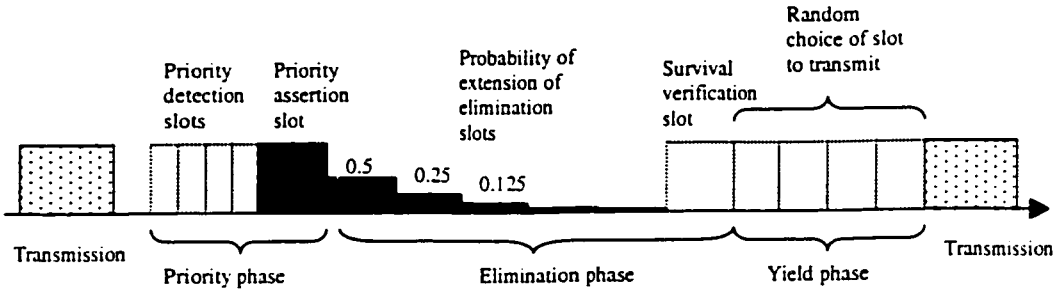


Figure 1-11 EY-NPMA in HIPERLAN

1.2.3.2 Routing in HIPERLAN

HIPERLAN type 1 is based on a hop-by-hop link state routing scheme. Each node broadcasts its link state updates in the network. This information helps a node to maintain a topology table and then to calculate its routing table on the basis of this topology information. Routing in HIPERLAN is performed using the concept of multipoint relays. Only multipoint relays are selected as intermediate nodes while calculating a route from source to destination. Each node of a network selects its multipoint relays among the one-hop neighbor nodes. Each node broadcasts this information periodically. In HIPERLAN,

the control messages are using these multipoint relays to achieve route optimization. The protocol performs hop-by-hop forwarding of messages by dynamically and distributive calculating the routes at each node. Hence, the latest information is used to determine the route for a packet if a link failure occurs while the packet is in transit (Fig. 1-12). As the protocol uses internal routing with packet forwarding by intermediate nodes, there is no decrease in reliability when network size increases: more nodes become available to link the distant nodes and hence reinforce the connectivity of the network.

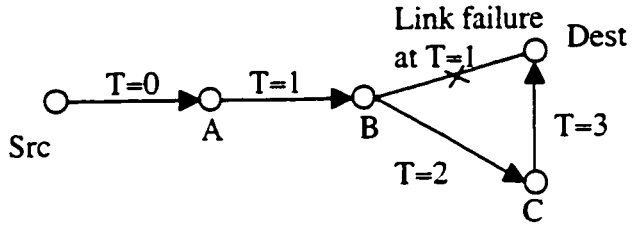


Figure 1-12 Hop by hop forwarding in HIPERLAN

2 Mobile Ad-hoc (MANET) routing protocols

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

2.1 Conventional protocols

Because of the fact that it may be necessary to hop several hops (multi-hop) before a packet reaches the destination, a routing protocol is needed. If a routing protocol is needed, why not use a conventional routing protocol like link state or distance vector [20]? Distance vector routing protocol requires that each router simply inform its neighbors of its routing table. For each network path, the receiving routers pick the neighbor advertising the lowest cost, and then add this entry into its routing table for re-advertisement. Link state routing protocol requires each router to maintain at least a partial map of the network. When a network link changes state, a notification, called a link state advertisement (LSA) is flooded throughout the network. All the routers note the change, and re-compute their routes accordingly. The main problem with link-state and distance vector is that they are designed for a static topology, which means that they would have problems to converge to a steady state in an ad-hoc network with a very frequently changing topology.

Link state and distance vector would probably work very well in an ad-hoc network with low mobility, i.e. a network where the topology is not changing very often. The problem that still remains is that link-state and distance-vector are highly dependent on periodic

control messages. As the number of network nodes can be large, the potential number of destinations is also large. This requires large and frequent exchange of data among the network nodes. This is in contradiction with the fact that all updates in a wireless interconnected ad hoc network are transmitted over the air and thus are costly in resources such as bandwidth, battery power and CPU. Because both link-state and distance vector tries to maintain routes to all reachable destinations, it is necessary to maintain these routes and this also wastes resources for the same reason as above.

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

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

2.1.1 Link State

In link-state routing [20], each node maintains a view of the complete topology with a cost for each link. To keep these costs consistent; each node periodically broadcasts the link costs of its outgoing links to all other nodes using flooding. As each node receives this information, it updates its view of the network and applies a shortest path algorithm to choose the next-hop for each destination.

Some link costs in a node view can be incorrect because of long propagation delays, partitioned networks, etc. Such inconsistent network topology views can lead to formation of routing-loops. These loops are however short-lived, because they disappear in the time it takes a message to traverse the diameter of the network.

2.1.2 Distance Vector

In distance vector [20] each node only monitors the cost of its outgoing links, but instead of broadcasting this information to all nodes, it periodically broadcasts to each of its neighbors an estimate of the shortest distance to every other node in the network. The receiving nodes then use this information to recalculate the routing tables, by using a shortest path algorithm.

Compared to link-state, distance vector is computed more efficiently, easier to implement and requires much less storage space. However, it is well known that distance vector can cause the formation of both short-lived and long-lived routing loops.

2.1.3 Source Routing

Source routing [20] means that each packet must carry the complete path that the packet should take through the network. The routing decision is therefore made at the source. The advantage with this approach is that it is very easy to avoid routing loops. The disadvantage is that each packet requires a slight overhead.

2.1.4 Flooding

Many routing protocols use broadcast to distribute control information, that is, send the control information from an origin node to all other nodes. A widely used form of broadcasting is flooding [20] and operates as follows. The origin node sends its information to its neighbors (in the wireless case, this means all nodes that are within transmitter range). The neighbors relay it to their neighbors and so on, until the packet has reached all nodes in the network. A node will only relay a packet once and to ensure this some sort of sequence number can be used. This sequence number is increased for each new packet a node sends.

2.2 Desirable properties

If the conventional routing protocols do not meet our demands, we need a new routing protocol. The question is what properties such protocols should have? These are some of the properties [25] that are desirable:

- Distributed operation

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

- Loop free

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

- Demand based operation

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

- Unidirectional link support

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

- Security

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

- Power conservation

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

- Multiple routes

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

- Quality of service support

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

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

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

2.3 MANET routing protocols

The protocols proposed in MANET change rapidly. Here we discuss 10 protocols.

- Fisheye State Routing Protocol (FSR) for Ad Hoc Networks
- Ad Hoc On Demand Distance Vector (AODV) Routing
- Dynamic Source Routing Protocol (DSR)
- Zone Routing Protocol (ZRP) for Ad Hoc Networks
- Temporally-Ordered Routing Algorithm (TORA)
- Hierarchical State Routing (HSR)
- Destination Sequenced Distance Vector (DSDV)
- IP Flooding in Ad hoc Mobile Networks
- Topology Broadcast based on Reverse-Path Forwarding (TBRPF)
- Optimized Link State Routing Protocol (OLSR)

2.4 Fisheye State Routing Protocol (FSR) for Ad Hoc Networks

Fisheye State Routing [16] is a table-driven routing protocol adapted to the wireless ad hoc environment. It provides an implicit hierarchical routing structure. Through updating link state information with different frequencies depending on the fisheye scope distance, FSR scales well to large network size and keeps overhead low without compromising route computation accuracy when the destination is near. The routing accuracy of FSR is

comparable with an ideal Link State scheme. By retaining a routing entry for each destination, FSR avoids the extra work of "finding" the destination (as in on-demand routing) and thus maintains low single packet transmission latency. As mobility increases, routes to remote destinations become less accurate. However, when a packet approaches its destination, it finds increasingly accurate routing instructions as it enters sectors with a higher refresh rate. As a result, FSR is more desirable for large mobile networks where mobility is high and the bandwidth is low. By choosing proper number of scope levels and radius size, FSR proves to be a flexible solution to the challenge of maintaining accurate routes in ad hoc networks.

The Fisheye State Routing (FSR) algorithm for ad hoc networks introduces the notion of multi-level "scope" to reduce routing update overhead in large networks. A node stores the Link State for every destination in the network. It periodically broadcasts the Link State update of a destination to its neighbors with a frequency that depends on the hop distance to that destination (i.e., the "scope" relative to that destination). State updates corresponding to far away destinations are propagated with lower frequency than those for close by destinations. From state updates, nodes construct the topology map of the entire network and compute efficient routes. The route on which the packet travels becomes progressively more accurate as the packet approaches its destination. FSR resembles Link State routing in that it propagates Link State updates. However, the updates are propagated as aggregates, periodically (with period dependent on distance) instead of being flooded individually from each source. FSR leads to major reduction in link overhead caused by routing table updates. It enhances scalability of large, mobile ad hoc networks.

The following properties of FSR highlight its advantages.

* Simplicity
* Usage of up-to-date shortest routes
* Robustness to host mobility
* Exchange Partial Routing Update with neighbors
* Reduced Routing Update Traffic

FSR is best suited for large-scale mobile ad hoc wireless networks, as the scope update scheme has larger advantages in reducing routing update packet size and achieve high date packet to routing packet ratio. Moreover, the fact that the route error is weighted by distance obviously reduces the sensitivity to network size.

FSR is also suited for high mobility ad hoc wireless networks. This is because in a mobility environment, a change on a link far away from the source does not necessarily cause a change in the routing table at the source.

Fisheye State Routing is a table-driven or proactive routing protocol. It bases on link state protocol and has the ability of immediately providing route information when needed. The fisheye scope technique allows exchanging link state messages at different intervals for nodes within different fisheye scope distance, which reduces the link state message size. Further optimization allows FSR only broadcast topology message to neighbors in order to reduce the flood overhead. With these optimizations, FSR significantly reduces the topology exchange overhead and scales well to large network size.

FSR routes each data packet according to locally computed routing table. The routing table uses most recent topology information. The fisheye scope updating message scheme will not lose routing accuracy for inner scope nodes. For outer scope nodes, information in routing entries may blur due to longer exchange interval, but the extra work of

"finding" the destination (as in on-demand routing) is not necessary. Thus low single packet transmission latency can be maintained. At a mobile environment, this inaccuracy for remote nodes will increase. However, when a packet approaches its destination, it finds increasingly accurate routing instructions as it enters sectors with a higher refresh rate.

FSR doesn't trigger any control messages when a link failure is reported. Thus it is suitable for high topology change environment. The broken link will not be included in the next fisheye scope link state message exchange. Sequence number and table refreshment enables the FSR to maintain the latest link state information and loop-free in an unreliable propagation media and highly mobile network.

2.5 Ad Hoc On Demand Distance Vector (AODV) Routing

2.5.1 Description

The Ad Hoc On-Demand Distance Vector (AODV) routing protocol enables multi-hop routing between participating mobile nodes wishing to establish and maintain an ad-hoc network [3]. AODV is based upon the distance vector algorithm. The difference is that AODV is reactive, as opposed to proactive protocols like DV, i.e. AODV only requests a route when needed and does not require nodes to maintain routes to destinations that are not actively used in communications. As long as the endpoints of a communication connection have valid routes to each other, AODV does not play any role.

Features of this protocol include loop freedom and that link breakages cause immediate notifications to be sent to the affected set of nodes, but only that set. Additionally, AODV

has support for multicast routing and avoids the Bellman Ford "counting to infinity" problem. The use of destination sequence numbers guarantees that a route is "fresh".

- Route table management

AODV needs to keep track of the following information for each route table entry:

  - Destination IP Address: IP address for the destination node.
  - Destination Sequence Number: Sequence number for this destination.
  - Hop Count: Number of hops to the destination.
  - Next Hop: The neighbor, which has been designated to forward messages to the destination for this route entry.
  - Lifetime: The time for which the route is considered valid.
  - Active neighbor list: Neighbor nodes that are actively using this route entry.
  - Request buffer: Makes sure that a request is only processed once.

- Route discovery

A node broadcasts a RREQ when it needs a route to a destination and does not have one available. This can happen if the route to the destination is unknown, or if a previously valid route expires. After broadcasting a RREQ, the node waits for a RREP. If the reply is not received within a certain time, the node may rebroadcast the RREQ or assume that there is no route to the destination (See Fig. 2-1).

Forwarding of RREQs is done when the node receiving a RREQ does not have a route to the destination. It then rebroadcast the RREQ. The node also creates a temporary reverse route to the Source IP Address in its routing table with next hop equal to the IP address field of the neighboring node that sent the broadcast RREQ. This is done to keep track of

a route back to the original node making the request, and might be used for an eventual RREP to find its way back to the requesting node. The route is temporary in the sense that it is valid for a much shorter time, than an actual route entry.

When the RREQ reaches a node that either is the destination node or a node with a valid route to the destination, a RREP is generated and unicasted back to the requesting node. While this RREP is forwarded, a route is created to the destination and when the RREP reaches the source node, there exists a route from the source to the destination.

- Route maintenance

When a node detects that a route to a neighbor no longer is valid, it will remove the routing entry and send a link failure message, a triggered route reply message to the neighbors that are actively using the route, informing them that this route no longer is valid. For this purpose AODV uses an active neighbor list to keep track of the neighbors that are using a particular route. The nodes that receive this message will repeat this procedure. The message will eventually be received by the affected sources that can chose to either stop sending data or requesting a new route by sending out a new RREQ.

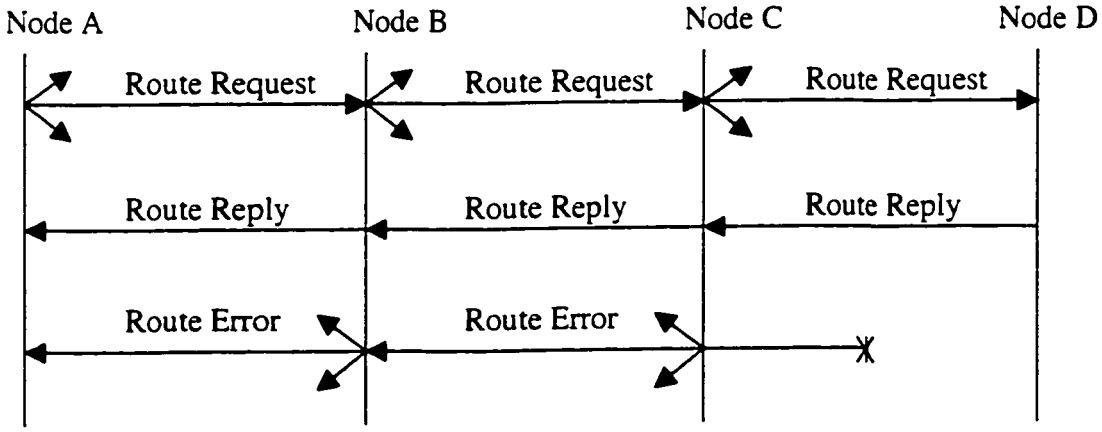


Figure 2-1 AODV Messages.

2.5.2 Properties

The advantage with AODV compared to classical routing protocols like distance vector and link-state is that AODV has greatly reduced the number of routing messages in the network. AODV achieves this by using a reactive approach. This is probably necessary in an ad-hoc network to get reasonably performance when the topology is changing often.

AODV is also routing in the more traditional sense compared to for instance source routing based proposals like DSR. The advantage with a more traditional routing protocol in an ad-hoc network is that connection from the ad-hoc network to a wired network like the Internet is most likely easier.

The sequence numbers that AODV uses represents the freshness of a route and is increased when something happens in the surrounding area. The sequence prevents loops from being formed, but can however also be the cause for new problems. What happens for instance when the sequence numbers no longer are synchronized in the network? This can happen when the network becomes partitioned, or the sequence numbers wrap around.

2.6 Dynamic Source Routing Protocol (DSR)

2.6.1 Description

Dynamic Source Routing (DSR) [14] also belongs to the class of reactive protocols and allows nodes to dynamically discover a route across multiple network hops to any destination. Source routing means that each packet in its header carries the complete ordered list of nodes through which the packet must pass. DSR uses no periodic routing messages (e.g. no router advertisements), thereby reducing network bandwidth overhead,

conserving battery power and avoiding large routing updates throughout the ad-hoc network.

Instead DSR relies on support from the MAC layer (the MAC layer should inform the routing protocol about link failures). The two basic modes of operation in DSR are route discovery and route maintenance.

- Route discovery

Route discovery is the mechanism whereby a node X wishing to send a packet to Y, obtains the source route to Y. Node X requests a route by broadcasting a Route Request (RREQ) packet. Every node receiving this RREQ searches through its route cache for a route to the requested destination. DSR stores all known routes in its route cache. If no route is found, it forwards the RREQ further and adds its own address to the recorded hop sequence. This request propagates through the network until either the destination or a node with a route to the destination is reached. When this happen a Route Reply (RREP) is unicasted back to the originator.

The route back to the originator can be retrieved in several ways. The simplest way is to reverse the hop record in the packet. Once a route is found, it is stored in the cache with a time stamp and the route maintenance phase begins.

- Route maintenance

Route maintenance is the mechanism by which a packet sender S detects if the network topology has changed so that it can no longer use its route to the destination D. This might happen because a host listed in a source route, move out of wireless transmission range or is turned off making the route unusable. A failed link is detected by monitoring acknowledgements.

When route maintenance detects a problem with a route in use, a route error packet is sent back to the source node. When this error packet is received, the hop in error is removed from this host route cache, and all routes that contain this hop are truncated at this point.

2.6.2 Properties

DSR uses the key advantage of source routing. Intermediate nodes do not need to maintain up-to-date routing information in order to route the messages they forward. There is also no need for periodic routing advertisement messages, which will lead to reduce network overhead, particularly during periods when little or no significant host movement is taking place. Battery power is also conserved on the mobile hosts, both by not sending and by not needing to receive message, a host could go down to sleep instead.

This protocol has the advantage of learning routes by scanning for information in messages that are received. A route from A to C through B means that A learns the route to C, but also that it will learn the route to B. The source route will also mean that B learns the route to A and C and that C learns the route to A and B. This form of active learning is very good and reduces overhead in the network.

However, each packet carries a slight overhead containing the source route of the packet. This overhead grows when the packet has to go through more hops to reach the destination. So the messages sent will be slightly bigger, because of the overhead.

Running the interfaces in promiscuous mode is a serious security issue. Since the address filtering of the interface is turned off and all messages are scanned for information. A potential intruder could listen to all messages and scan them for useful information such as passwords and credit card numbers. Applications have to provide the security by encrypting their data messages before transmission. The routing protocols are prime

targets for impersonation attacks and must therefore also be encrypted. One way to achieve this is to use IP-sec.

2.7 Zone Routing Protocol (ZRP)

2.7.1 Description

Zone Routing Protocol (ZRP) [28] is a hybrid of a reactive and a proactive protocol. It divides the network into several routing zones and specifies two totally detached protocols that operate inside and between the routing zones.

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

The second protocol, the Inter-zone Routing Protocol (IERP) is reactive and is used for finding routes between different routing zones. This is useful if the destination node does not lie within the routing zone.

The protocol then broadcasts (i.e. bordercasts) a Route REQuest (RREQ) to all border nodes within the routing zone, which in turn forwards the request if the destination node is not found within their routing zone. This procedure is repeated until the requested node is found and a route reply is sent back to the source indicating the route. IERP uses a Bordercast Resolution Protocol (BRP) that is included in ZRP. BRP provides

bordercasting services, which do not exist in IP. Bordercasting is the process of sending IP datagrams from one node to all its peripheral nodes. BRP keeps track of the peripheral nodes and resolves a border cast address to the individual IP-addresses of the peripheral nodes. The message that was bordercasted is then encapsulated into a BRP packet and sent to each peripheral node.

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

Consider the network in Fig. 2-2. Node S wants to send a packet to node D. Since node D is not in the routing zone of S, a route request is sent to the border nodes B and F. Each border node checks to see if D is in their routing zone. Neither B nor F finds the requested node in their routing zone; thus the request is forwarded to the respectively border nodes. F sends the request to S, B, C and H while B sends the request to S, F, E and G. Now the requested node D is found within the routing zone of both C and E thus a reply is generated and sent back towards the source node S. To prevent the requests from going back to previously queried routing zone, a Processed Request List issued. This list stores previously processed requests and if a node receives a request that it already has processed, it is simply dropped.

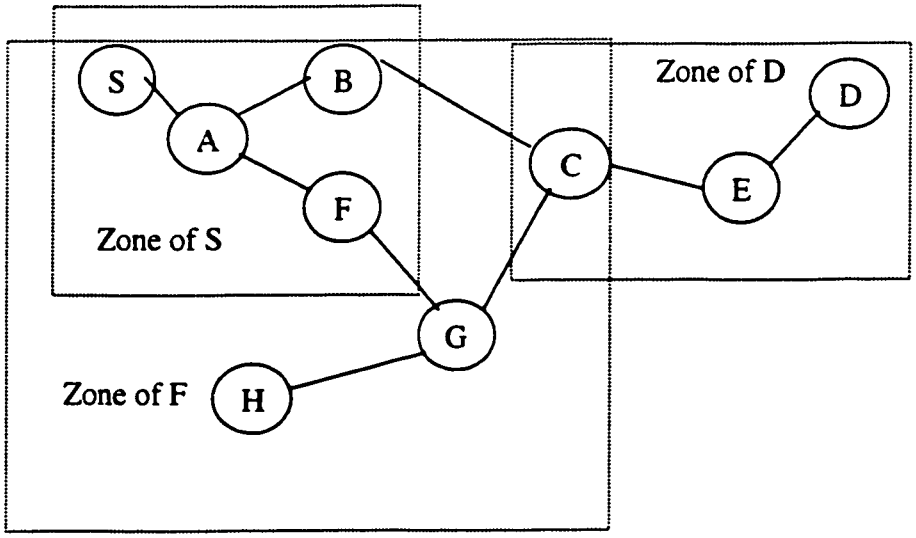


Figure 2-2 Example network of ZRP.

2.7.2 Properties

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

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

ZRP also limits propagation of information about topological changes to the neighborhood of the change only (as opposed to a fully proactive scheme, which would

basically flood the entire network when a change in topology occurred). However, a change in topology can affect several routing zones.

2.8 Temporally-Ordered Routing Algorithm (TORA)

2.8.1 Description

Temporally Ordered Routing Algorithm (TORA) [27] is a distributed routing protocol. The basic underlying algorithm is one in a family referred to as link reversal algorithms. TORA is designed to minimize reaction to topological changes. A key concept in its design is that control messages are typically localized to a very small set of nodes. It guarantees that all routes are loop-free (temporary loops may form), and typically provides multiple routes for any source/destination pair. It provides only the routing mechanism and depends on Internet MANET Encapsulation Protocol (IMEP) for other underlying functions.

TORA can be separated into three basic functions: creating routes, maintaining routes, and erasing routes. The creation of routes basically assigns directions to links in an undirected network or portion of the network, building a directed acyclic graph rooted at the destination (See Fig. 2-3).

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

height than specified in the UPD message. The node will then broadcast its own UPD packet. This will result in a number of directed links from the originator of the QRY packet to the destination. This process can result in multiple routes.

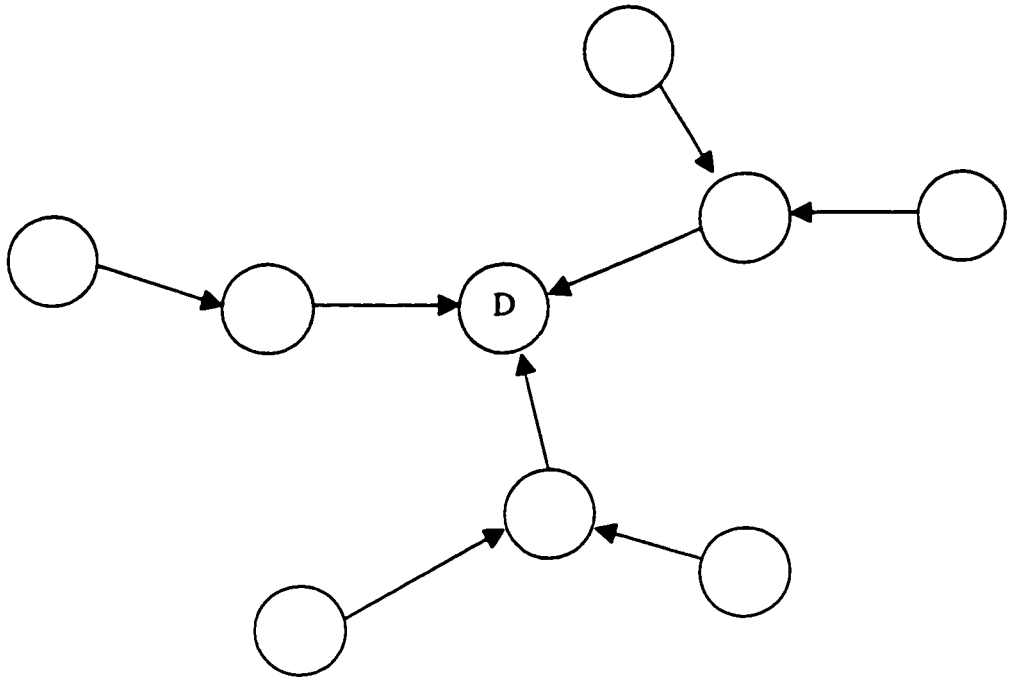


Figure 2-3 Directed acyclic graph rooted at destination.

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

2.8.2 Properties

The protocols underlying link reversal algorithm will react to link changes through a simple localized single pass of the distributed algorithm. This prevents CLR messages to

propagate too far in the network. The overhead in TORA is quite large because of the use of IMEP.

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

2.9 Hierarchical State Routing (HSR)

HSR combines dynamic, distributed multilevel hierarchical grouping (clustering) and efficient location management [5].

Clustering (dividing nodes into groups and different kinds of nodes) at the MAC and network layers help the routing of data as the number of nodes grows and subsequently the cost of processing (nodes memory and processing required).

HSR keeps a hierarchical topology, where selected cluster heads at the lowest level become member of the next higher level and so on... While this clustering is based on network topology (i.e. physical), further logical partitioning of nodes eases the location management problem in HSR. Fig. 2-4 shows four physical level clusters namely CO-1, CO-2, CO-3, and CO-4. Cluster heads are selected either manually (by the network administrator) or via the appropriate real time distributed voting mechanism, e.g. In Fig. 2-4, nodes 1,2,3,4 are assumed to be cluster heads, nodes 5, 9, 10 are internal nodes and nodes 6, 7, 8, 11 are gateway nodes.

Gateway nodes are those belonging to more than one physical cluster. At the physical cluster level all cluster heads, gateways, and internal nodes use the MAC address, but each also has a hierarchical address as will follow.

Cluster heads exchange the link state LS information with other cluster heads via the gateway nodes. For example in Fig. 2-4 cluster heads 2,3 exchange LS information via gateway 8. Cluster heads 4,3 via gateway 11 and so on. In the sequel logical level C1-1 is formed of cluster heads 1, 2, 3, 4. However, in level C1-1, only 1 and 3 are cluster heads.

Similarly at level C2-1, only 1 is a cluster head while node 3 is an ordinary node at C1-2 level.

Routing table storage at each node is reduced by the aforementioned hierarchical topology. For example, for node 5 to deliver a packet to node 10, it will forward it to the cluster head 1 which has a tunnel (route) to the cluster head 3 which finally deliver the packet to node 10. But how would the cluster head 1 know that node 10 is a member of a cluster headed by 3?

The answer is: Nodes in each cluster exchange virtual LS information about the cluster (who is the head, who are the members) and lower cluster (with less details) and the process repeats at lower clusters. Each virtual node floods this LS information down to nodes within its lower level cluster. Consequently, each physical node would have hierarchical topology information rather than the full topology of flat routing where all individual routes to each node in the networks are stored at all nodes and are exchanged at the same rate to all nodes.

Node 1 has a virtual link or tunnel, i.e. the succession of nodes (1, 6, 2, 8, 3) to node 3 which is the cluster head of the final destination. This tunnel is computed from the LS information flooded down from higher clusters heads as above). Finally node 3, delivers the packet to node 10 along the downward hierarchical path which is a mere single hop.

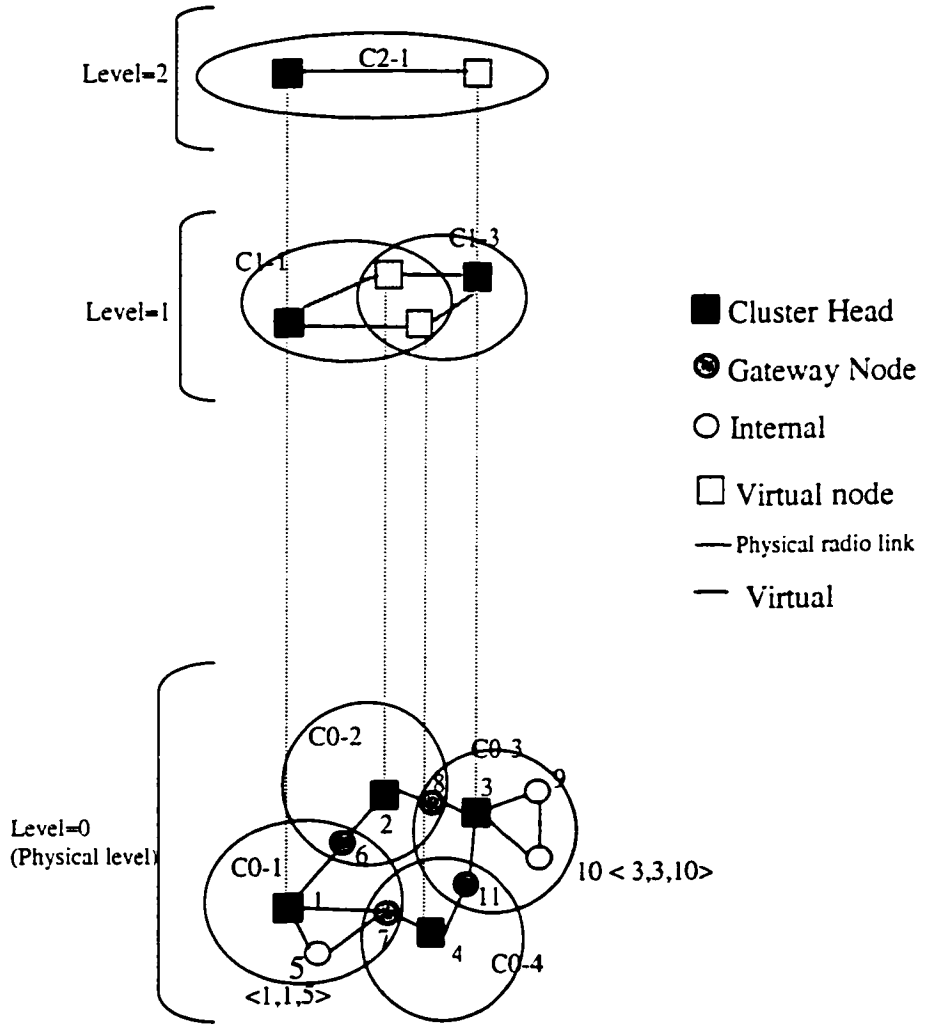


Figure 2-4 An exampie of physical/virtual clustering

2.10 Destination Sequenced Distance Vector(DSDV)

2.10.1 Description

Destination Sequenced Distance Vector (DSDV) is a hop-by-hop distance vector routing protocol that in each node has a routing table that for all reachable destinations stores the next-hop and number of hops for that destination. Like distance-vector, DSDV requires that each node periodically broadcast routing updates. The advantage with DSDV over traditional distance vector protocols is that DSDV guarantees loop-freedom [2].

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

These adjustments consist of triggered updates that will take care of topology changes in the time between broadcasts. To reduce the amount of information in these packets there are two types of update messages defined: full and incremental dump. The full dump

carries all available routing information and the incremental dump that only carries the information that has changed since the last dump.

2.10.2 Properties

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

2.11 IP Flooding in Ad hoc Mobile Networks

An ad hoc mobile network is a collection of nodes, each of which communicates over wireless channels and is capable of movement. Nodes participating in such an ad hoc network communicate on a peer-to-peer basis. Flooding is often a desired form of communication in these networks, as it can enable both the dissemination of control information and the delivery of data messages. This section describes a method for sending messages to every node in ad hoc networks [4].

A particular specification was made for a classical flooding algorithm, as it can be used to disseminate IP packets across ad hoc networks. Flooding is needed in many circumstances; in particular, it is useful for the kind of route discovery operations that are required for on-demand route acquisition in several MANET protocols.

This protocol specification works when the nodes flooding messages ensure that each such distinct packet that they send is tagged with a distinct value in the ident field of the IP header.

In IPv4, there are two kinds of broadcast address, and it seems that neither one of them is likely to present a good choice for the IP address to be used for flooding. The IPv4 address for "limited broadcast" is 255.255.255.255, and is not supposed to be forwarded. Since the nodes in an ad hoc network are asked to forward the flooded messages, the limited broadcast address is a poor choice. The other available choice, the "directed broadcast" address, would presume a choice of routing prefix for the ad hoc network and thus is not a reasonable choice.

Thus, new multicast groups for flooding to all nodes of an ad hoc network are specified. These multicast groups are specified to contain all nodes of a contiguous ad hoc network, so that messages transmitted to the multicast address associated with the group will be delivered to all nodes as desired. For IPv6, the multicast address is specified to be "site-local". The names of the multicast groups are given as "ALL_IPv4_MANET_NODES" and "ALL_IPv6_MANET_NODES".

2.12 Topology Broadcast based on Reverse-Path Forwarding (TBRPF)

TBRPF is a proactive, link-state routing protocol [24] designed for mobile ad-hoc networks, which provides hop-by-hop routing along shortest paths to each destination. Each node running TBRPF computes a source tree (providing paths to all reachable nodes) based on partial topology information stored in its topology table, using a

modification of Dijkstra's algorithm. To minimize overhead, each node reports only part of its source tree to neighbors. This is in contrast to other protocols (e.g., STAR) in which each node reports its entire source tree to neighbors. TBRPF uses a combination of periodic and differential updates to keep all neighbors informed of the reportable part of its source tree. Each node also has the option to report additional topology information (up to the full topology), to provide improved robustness in highly mobile networks.

TBRPF performs neighbor discovery using "differential" HELLO messages that report only changes in the status of neighbors. This results in HELLO messages that are much smaller than those of other link-state routing protocols such as OLSR.

Each node running TBRPF computes a source tree (providing shortest paths to all reachable nodes) based on partial topology information stored in its topology table, using a modification of Dijkstra's algorithm.

TBRPF uses a combination of periodic and differential updates to keep all neighbors informed of the reportable part of its source tree. Each node also has the option to report addition topology information (up to the full topology), to provide improved robustness in highly mobile networks.

TBRPF consists of two modules: the neighbor discovery module and the routing module (which performs topology discovery and route computation).

2.13 Optimized Link State Routing Protocol

The Optimized Link State Routing protocol (OLSR) [26] is an important proactive routing protocol existing in the IETF MANET working group proposed by INRIA (Institut National De Recherche En Informatique Et En Automatique), France. It's an optimization over the pure link state protocol. The protocol uses control messages for

neighbor sensing to discover the neighborhood and to establish knowledge of the link status between the node and all of its neighbors. This knowledge is then, through the use of Multi Point Relays (MPRs), flooded into the network, providing each node with partial topology information, necessary to compute optimal routes to all nodes in the network. Only nodes selected as MPRs retransmit topology information into the network. The use of MPRs combined with local duplicate elimination is used to minimize the number of retransmissions in the network and thereby reduce overhead.

2.13.1 Multi Point Relay

OLSR optimizes the process of flooding control messages by using Multi Point Relays (MPRs). Each node selects a set of MPRs among its neighbors. The role of the MPRs is to retransmit the selecting node's control messages. The MPR set is selected so that all two-hop neighbors can be reached through nodes in the MPR set. Only neighbors with symmetric links are considered when choosing MPRs. Computation of the MPR set is triggered by changes in the neighborhood or two-hop neighborhood. MPR selector set is the collection of nodes that have selected a particular node as MPR.

A heuristic selection algorithm is used. First, all the neighbors that provide the only path to one or more two-hop neighbors are selected. Next, one of the neighbors that can reach most of the two-hop neighbors, not yet covered by the MPR set, is selected and added to the MPR set. This step is repeated until all two-hop neighbors can be reached. The last step in the algorithm is an optimization of the MPR set: Each node in the MPR set is examined. If the MPR set without the particular node still covers the two-hop neighborhood, the node is removed from the set.

If the minimal MPR set is found, fewer messages are retransmitted in the network. It is, however, more important to cover the whole two-hop neighborhood than to have a small MPR set. This is because it is necessary to construct a partial topology graph with a subset of all links, yet with all nodes, to gain enough topology information to make routes from all nodes to all nodes.

Only MPRs retransmit a control message and only if the message comes from a node in its MPR selector set. Other nodes will process the packet, not retransmit it. Using MPRs therefore results in a significant reduction in the number of retransmissions in the network. Fig. 2-5(a) illustrates transmission of a packet in a small MANET using pure flooding, whereas Fig. 2-5(b) illustrates the same situation, but with the use of MPRs. Each arrow represents a transmission.

The load of control traffic is minimized in part because only nodes selected as MPRs transmits topology information, and in part because only MPRs retransmit control messages for other nodes. Furthermore, the topology information only consists of links to the nodes that have selected the particular node as MPR. This means that the control packet is smaller than if information about all links' states were diffused into the network.

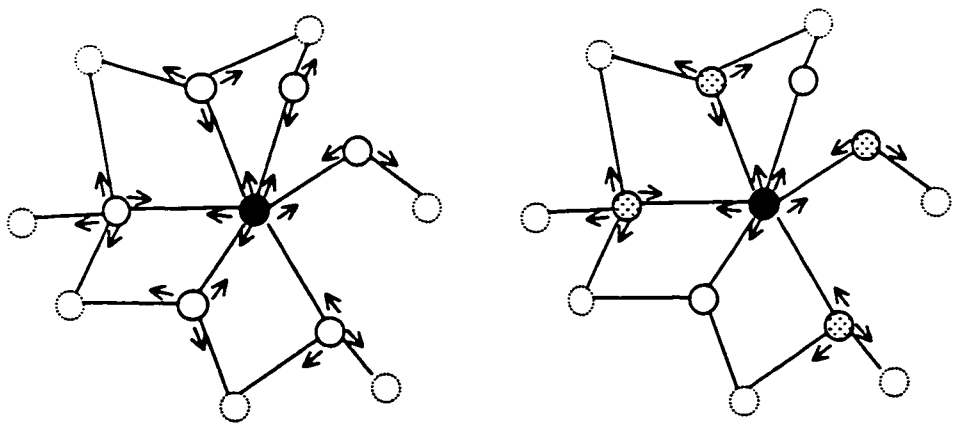


Figure 2-5 A small network with full flooding and MPR flooding.

2.13.2 Routing

Based on the information in the topology table each node calculates the routes to all other nodes using a shortest path algorithm, for example Dijkstra's algorithm, using hop-to-hop routing.

OLSR maintains the routing tables, but leaves it up to the underlying operating system to take care of packet forwarding. Thereby OLSR is not a part of the protocol stack, but only calculates routes and changes the routing tables in the operating system.

The detailed description and analysis is in the next chapter.

2.14 Some performance results

Computer simulations of different protocols have been conducted by many researchers. An important one among them is done by Iwata [12] in 1999 which gives comparison of 5

kinds of routing protocols. The network consists of 200 nodes. Data packets were generated from a Poisson kind of traffic with inter-arrival period of 2.5 seconds amounting to a data rate of 4 Kbps per source/destination pair. Each data packet consists of 10K bits, buffer size at each node is 15 packets, nodes moved at a speed of 60 km/hr, maximum radio range is 120 m and data rate on channel is 2 Mbps. On demand routing applied could be AODV schemes. Type A and B differ in routing entry time outs. The routing table entries are timed out in 3 s for type A and 6 s for type B.

For Destination Sequenced Distance Vector routing (DSDV), the overhead performance is worse than FSR, HSR (Hierarchical State Routing)...etc. Fig. 2-7 shows that the DSDV overhead performance versus speed of nodes is prohibitive. Figs. 2-6 and 2-7 show that the DSDV performance is somewhat independent of the nodes traffic conditions, this is due to the fact that all nodes are busy anyhow with the transmission of their routing tables. DSDV exhibits good delay and average number of hops (Figs. 2-8 and 2-9). However, for highly deployed networks, where the number of nodes increases as the area enlarges, the overhead ratio is much worse than FSR, HSR...etc (Fig. 2-10).

For FSR, Figs. 2-6 and 2-7 show that the routing overhead per cluster does not change much with active nodes number or their mobility (speeds). However, FSR control overhead is relatively higher than other routing techniques.

Fig. 2-8 shows that the average data packet delay does not rise much with nodes mobility, also FSR performance is the best in this regard compared to other techniques. Similar results are drawn from Fig. 2-9 which shows the average number of hops traveled by a typical data packet. Fig. 2-10 shows that the FSR overhead ratio does not increase as much as other techniques as the network size increases.

For HSR. Fig. 2-6 shows that overhead ratio in HSR is relatively constant number of communication pairs. Also it is better than that of FSR, DSDV. Fig. 2-7 shows that HSR overhead ratio is better than FSR, DSDV as the nodes increase their speed. Fig. 2-8 also shows better delay for HSR, but the average number of hops in Fig. 2-9 does not reveal marked improvement compared to other routing techniques. Finally, Fig. 2-10 shows a superior overhead ratio performance of HSR, as the number of communication nodes rises (while keeping the user density the same).

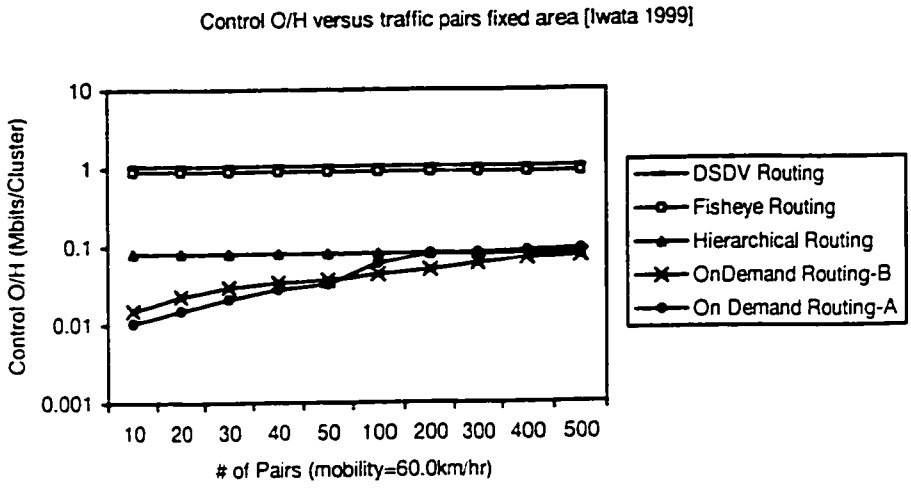


Figure 2-6 Control O/H versus traffic pairs fixed area

For on demand routing, Fig. 2-6 shows that overhead ratio increases almost linearly with the number of pairs—up to 100 pairs and increases with a lesser rate beyond 100 pairs. Fig. 2-7 shows that on demand routing overhead ratio is constant since the updates are independent of speed. Fig. 2-8 shows larger delay for on demand routing. The average number of hops in Fig. 2-9 increases with speed because of the re-computation of the route at midpoint when the original route is invalidated by node movements. Finally, Fig. 2-10 shows that on demand routing performs well in a small network and generates considerable O/H for large networks.

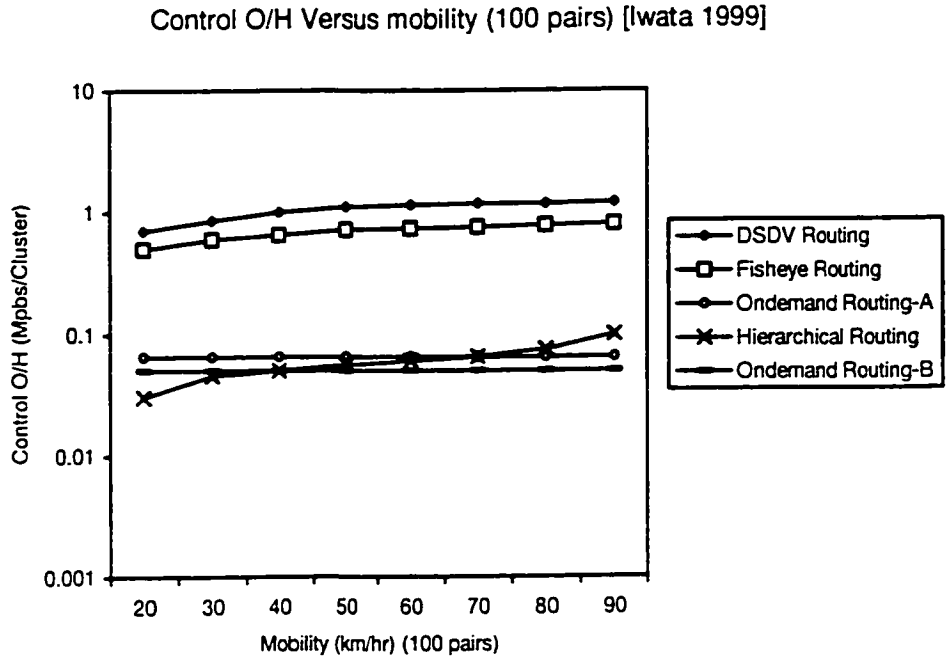


Figure 2-7 Control O/H Versus mobility (100 pairs)

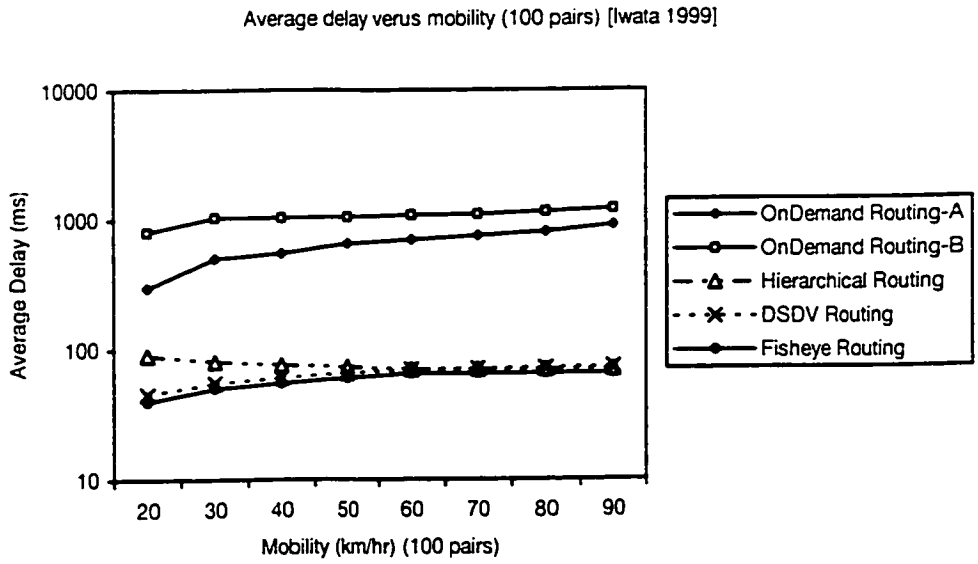


Figure 2-8 Average delay versus mobility (100 pairs)

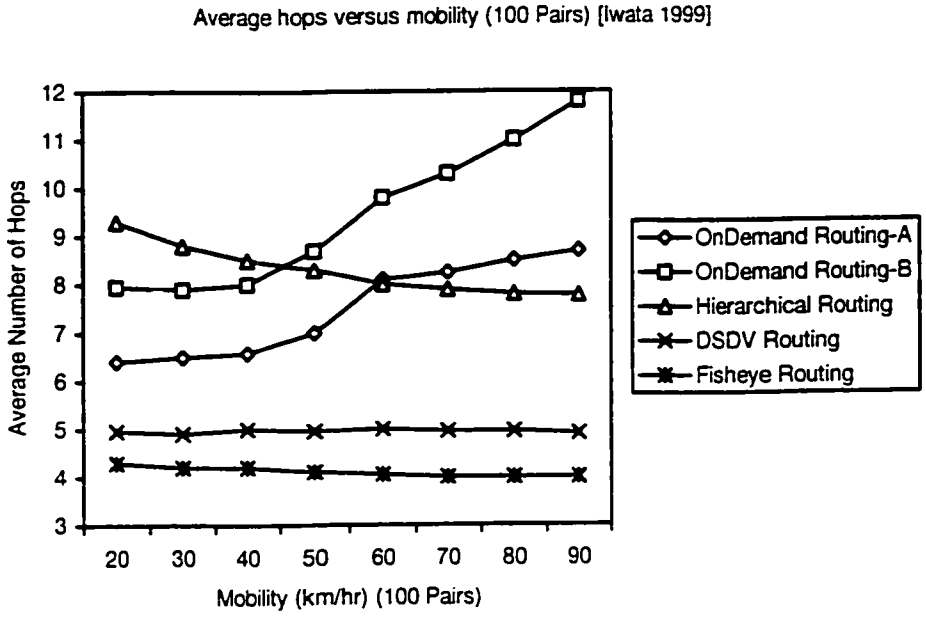


Figure 2-9 Average hops versus mobility (100 Pairs)

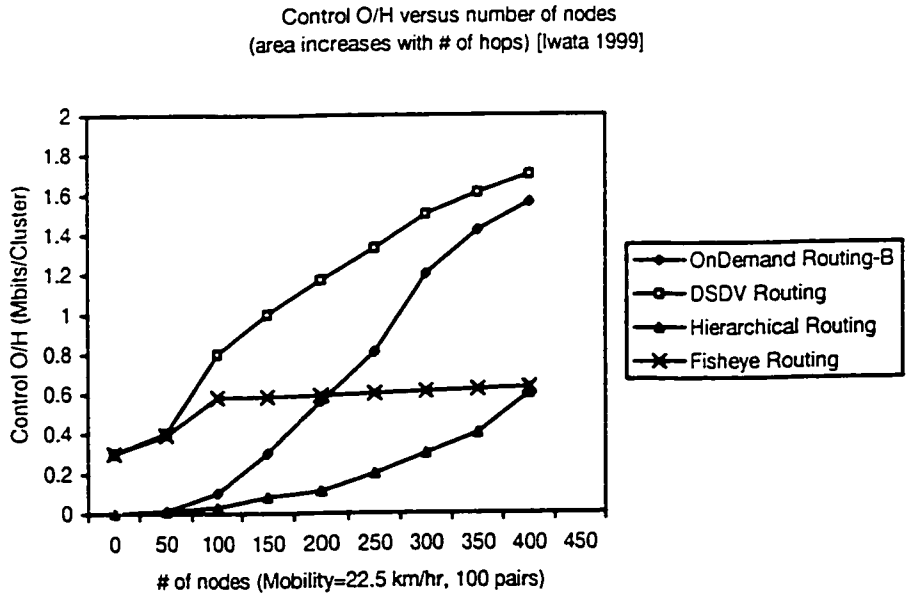


Figure 2-10 Control O/H versus number of nodes

3 OLSR simulation model

In this chapter a simulation model is established using c++ to evaluate the performance of OLSR. A detailed description is given in this chapter. Input and output parameters are also defined and assigned appropriate values.

3.1 Input-Output Parameter and Definition

3.1.1 Input Parameter and Definition

Besides some constant values, there are a lot of variables, by proper setting of which we can achieve better performance in our simulation. The followings are the input parameters:

- Area: All nodes are generated initially within a certain area of 1000*1000 m^2.
- Transmission rate: All nodes can transmit messages in the rate of 1M bps.
- Transmission range: Each node can transmit messages in the range of 250 m. Only the nodes within distance of 250m can hear (receive) the transmitted messages.
- Speed: Each mobile node moves at a speed from 0 to 108 km/h randomly selected every 8s.
- Direction: Each mobile node moves at a direction randomly selected from East, South, West, and North every 8s.
- Packet size: Each packet (message) contains 20000bits of information.
- Traffic load: We define traffic load parameter Lamda, different Lamda corresponding to different loads. It ranges from 0 to 0.125.Lamda is 0.125 meaning 4 messages can be generated for one node in a certain iteration.

- Buffer: Each node has two buffers, input buffer and output buffer. Each buffer can hold 30 messages.
- Number of nodes: 50 nodes will be generated within an area of $1000*1000m^2$.
- Mobile nodes: 20 nodes are mobile with random speed within a certain range and the remaining 30 nodes are stationary.
- Iteration number: Program runs in iterations, in each iteration one node can send or receive only one message. In order to have a stable network performance, the total number of iterations should be large enough. Here we set it 10000 after several tests.
- Sending control message period: Each node sends control messages periodically. We define sending control message period as the duration time to send Hello messages. The period of sending Topology Control (TC) message is 3 times longer compared to sending Hello message period.
- Data time out: Each data packet can queue in the output buffer for a certain time and will be dropped after. This time is called data time out.
- Node move probability: We assume every node can move with node move probability. For example the total average number of moving node per iteration is 50*0.4=20. But different nodes are involved during different iterations.
- Max message hop count: Each message has Max hop count to determine how long the message can go. If it has traveled this Max hops, then it is discarded from the net.

3.1.2 Output Parameters and Definitions

3.1.2.1 Average and variance of data transfer delay

Data transfer delay is defined as the total time that one data message takes from its generation to arriving its destination. It includes queuing delay (the time a message waits in the buffer) and transmission delay. The mean of data transfer delay is calculated by dividing the sum of all data transfer delays of all messages by the total number of successfully transmitted messages. The sample variance of data transfer delay is calculated by dividing the square sum of the difference between the average transfer delay and individual transfer delay by the total number of successfully transmitted messages. The formulas for the average of data transfer delay (ADTD) and variance of data transfer delay (VDTD) are as follows:

$$ADTD = \frac{\sum_{i=1}^{Ns} DTD_i}{Ns}$$

$$VDTD = \frac{\sum_{i=1}^{Ns} (ADTD - DTD_i)^2}{Ns}$$

Where Ns is total number of successfully transmitted messages in the whole program iterations of all nodes (Sum of total successful messages). DTD_i represents the transfer delay of each successfully transmitted message. It is equal to the difference between the end iteration and begin iteration contained in the message. Begin iteration and end iteration represents the generation time and final arrival time of each message respectively.

3.1.2.2 Average and variance of queuing delay

Queuing delay is defined as the time duration during which one message waits for transmission. Mean of queuing delay equals the sum of queuing delays of all messages in all program iterations of all nodes divided by the total number of messages. The variance of queuing delay equals the sum of the squares of the difference between average queuing delay and individual queuing delay in all program iterations and over all nodes divided by the total number of messages. The formulas for the Average Queuing Delay (AQD) and Variance of Queuing Delay (VQD) are listed as follows:

$$AQD = \frac{\sum_{i=1}^{Nm} QD_i}{Nm}$$

$$VQD = \frac{\sum_{i=1}^{Nm} (AQD - QD_i)^2}{Nm}$$

Where Nm is the total number of messages, QD_i represents the queuing delay of the i^{th} message.

3.1.2.3 Average and variance of hop counter

After data messages are generated, the node looks up its routing table to find a route to destination. If the destination is not a neighbor, data messages first go to the next hop in the route and continue going on to the next hop until they reach the destination node. In other words, messages need to travel several hops to reach the destination. For simulation purposes, we set the hop counter to record the number of hops that the message needs to reach the destination. Mean of the hop counter is defined as the ratio of sum of all hop-counters of all successfully transmitted messages divided by the total number of successfully transmitted messages. Correspondingly, Variance of the hop counter is

defined as the sum of the square difference between the average hop counter and the individual hop counter divided by the total number of successfully transmitted messages by all nodes in the whole simulation time. The formulas for Average Hop Counter (AHC) and Variance of Hop Counter (VHC) are:

$$AHC = \frac{\sum_{i=1}^{Ns} HC_i}{Ns}$$

$$VHC = \frac{\sum_{i=1}^{Ns} (AHC - HC_i)^2}{Ns}$$

Where HC_i is the hop counter of each successfully transmitted message, Ns is the total number of successfully transmitted messages.

3.1.2.4 Average and variance of buffer overflow probability

Buffer overflow probability is 1 when buffer is full, otherwise is 0. Mean of buffer overflow probability equals the sum of individual buffer overflow probabilities over all program iterations and over all buffers divided by the total number of buffers and total number of iterations. The variance of buffer overflow probability equals the sum of the square difference between average buffer overflow probability and individual buffer overflow probability over all program iterations and over all nodes divided by the total number of iterations and by number of buffers. The formulas for Average buffer overflow probability (ABF) and Variance of buffer overflow probability (VBF) are listed as follows:

$$ABF = \frac{\sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{Nb} BF_{ijk}}{n * m * Nb}$$

$$VBF = \frac{\sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{Nb}(ABF - BF_{ijk})^2}{n*m*Nb}$$

Where m is the maximum program iterations and n is the total number of nodes, BF_{ijk} represents the buffer overflow probability of the k^{th} buffer in the j^{th} node and the i^{th} iteration. Nb is the number of buffers in each node. It is 2 in our simulation. In our project, n is equal to 50 and m ranges from (0~10000).

3.1.2.5 Average and variance of number of messages in the buffer

Mean of messages in buffers equals the sum of messages in buffers over all program iterations and over all nodes divided by the total number of buffers and by number of iterations. The variance of number of messages in the buffer equals the sum of the square of the difference between average messages in buffers and individual number of messages in a certain buffer in all program iterations and among all buffers divided by the total number of iterations and number of buffers. The formulas for Average of messages in buffers (APB) and Variance of messages in buffers (VPB) are listed as follows:

$$APB = \frac{\sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{Nb}PB_{ijk}}{n*m*Nb}$$

$$VPB = \frac{\sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{Nb}(APB - PB_{ijk})^2}{n*m*Nb}$$

Where m,n,Nb are same as before, PB_{ijk} represents the messages in the k^{th} buffers of the j^{th} node in the i^{th} iteration.

3.1.2.6 Throughput

Throughput is defined as total number of successfully delivered data messages divided by total number of data messages generated over all nodes and all iterations. The formula for Throughput is listed as follows:

$$Throughput = \frac{\sum_{i=1}^{m}\sum_{j=1}^{n} SD_{ij}}{\sum_{i=1}^{m}\sum_{j=1}^{n} DA_{ij}}$$

Where m, n are same as before, SD_{ij} represents the successfully transmitted data messages of the j^{th} node in the i^{th} iteration. DA_{ij} represents all data messages transmitted by the j^{th} node in the i^{th} iteration.

3.1.2.7 Generation overhead

Generation overhead is defined as total number of control messages generated in all iterations divided by total number of messages generated by all nodes in all iterations. The formula for the Generation overhead is listed as follows:

$$\text{Generation overhead} = \frac{\sum_{i=1}^{m}\sum_{j=1}^{n} CM_{ij}}{\sum_{i=1}^{m}\sum_{j=1}^{n} AM_{ij}}$$

Where m, n are same as before, CM_{ij} represents the number of control messages of the j^{th} node in the i^{th} iteration. AM_{ij} represents all messages generated by the j^{th} node in the i^{th} iteration.

3.1.2.8 Transmission overhead

Transmission overhead is defined as total number of transmissions for all control messages divided by the sum of total number of transmissions for all control messages

and successful data messages. The formulas for Transmission overhead is listed as follows:

$$\text{Transmission overhead} = \frac{\sum_{i=1}^{m}\sum_{j=1}^{n} CM_{ij}}{\sum_{i=1}^{m}\sum_{j=1}^{n} CM_{ij} + \sum_{i=1}^{m}\sum_{j=1}^{n} SD_{ij}}$$

Where m, n are same as before, CM_{ij} represents the number of control messages of the j^{th} node in the i^{th} iteration. SD_{ij} represents the successfully transmitted data messages of the j^{th} node in the i^{th} iteration.

3.2 Simulation procedure

3.2.1 Simulation assumptions

Our simulation is based on the following assumptions:

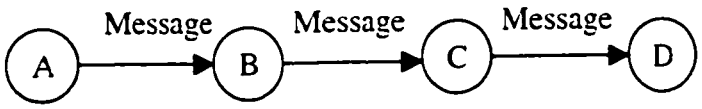
- The network of nodes represents a random graph model, in which nodes are placed randomly in a given region.
- All the nodes are identical, but they function independently of each other.
- If the node is mobile, each node independently decides its movement: its speed and the direction.
- The nodes access the transmission channel using CSMA/CA technique
- Nodes with distance larger than twice the transmission range can access channel without collision.
- One packet contains one message.
- Data and control messages have the same priority.
- Node can't receive and transmit messages at the same iteration. It can only receive OR transmit message during an iteration.

- Control messages have the same transmission time as data messages.
- All the node links are bi-directional.
- Nodes change their movements every 8 seconds.
- No support from link layer.
- Acknowledgement is not required.

3.2.2 Simulation data structure

3.2.2.1 Message data structure

- Originator Address: This address is the address of the node that originally generates the message.
- Sender address: This address is address of the node that is sending the message during a certain iteration. It is different from originator address. The latter is the message original source.
- Destination address: This is the destination of the message.
- Next address: This is the next hop that message will go to.



- A: Originator Address
- B: Sender Address
- C: Next Address
- D: Destination address

Figure 3-1 Example of different addresses

- Packet type: there are 3 types of messages in the simulation:

- ❖ Hello message: This is used to communicate with neighbors and the packet type is set 1.
- ❖ TC (Topology Control) message: In order to build the Intra-forwarding database needed for routing the messages, each relay node broadcasts specific service messages called Topology Control (TC) messages. Packet type is 2.
- ❖ Data message: Packet type is 3.

- MPR link: Hello message contains the link status information. All addresses of the nodes that have been selected as MPR are declared in the Hello message.
- Symmetric link: All addresses of nodes that have a symmetric link with the node are declared as the link status information in Hello message.
- MPR_S table: Each MPR declares a table that contains the MPR-selector address in TC message.
- Data: Randomly generated.
- Time to live: This determines the life of message. If it becomes zero, the message will no longer be transmitted in the network and will be discarded.
- Hop counter: This counts the number of hops that the message travels.
- Birth time: This records the iteration during which a certain message is born.
- Reach time: This records the iteration during which the message reaches its destination.
- Empty flag: If a message is discarded, we give the message an empty flag to show that it is discarded.

3.2.2.2 Node data structure

- Node number: Node number is the node index beginning from zero.
- Address: This is the node address.
- Position: Node position in the given region.
- Speed: Node moving speed
- Direction: node moving direction.
- Output buffer: This stores the messages to be sent.
- Input buffer: This stores the received messages.
- Send Hello iteration: Every node has a number for sending Hello messages. Once the iteration reaches its number for Hello, it will send one Hello message. Hello messages are sent every 100-iterations, e.g. 2s.
- Sending TC iteration: When iteration reaches its number for TC, the node will send one TC message. TC messages will be sent every 300-iterations, e.g. 6s.
- Neighbor table: It contains all the neighbor node addresses
- Neighbor table sequence number record: Because of node mobility, the neighbor table will change from time to time. It is important to keep a record to know the latest number of neighbor table.
- 2-hop neighbor table: It contains all the 2 –hop neighbor node addresses.
- MPR table: It contains all addresses of the MPR that have been selected by this node.
- MPR_S table: It contains all addresses of the MPR_S that select this node as their MPR.
- Topology table: It contains the topology information of network.

- Topology table sequence number record: This records the latest number of topology table.
- Routing table: It contains the routing information.

3.2.2.3 Table data structure

We use link list as data structure to establish all tables. Every table has two pointers, Begin pointer and End pointer to point to the beginning and end of table. Every entry in the table has a Next pointer to point to the next entry.

3.2.2.3.1 Neighbor table

In the neighbor table, each node records the information about its one-hop neighbors. The information is recorded in the Neighbor table as a neighbor entry. The neighbor table may have the following format to record these entries:

Seq_num

1.	N_addr	N_degree	N_time	*next
2.	N_addr	N_degree	N_time	*next
3.	,,	,,	,,	,,

Each entry in the table consists of N_addr, N_degree, N_time, and *next which specifies that the node with address N_addr is a one-hop neighbor to this local node, the number of links to the 2-hop neighbors through this node is N_degree. * next is a pointer pointing to the next entry. Each neighbor entry has an associated holding time N_time, upon expiry of which it is no longer valid and hence removed.

The neighbor table also contains a sequence number value Seq_num. Every time a node updates its neighbor table, this Seq_num is incremented to a higher value.

3.2.2.3.2 2-hop neighbor table

In the 2-hop neighbor table, each node records the information about its 2-hop neighbors. The information is recorded in the 2-hop neighbor table as a 2-hop neighbor entry. The 2-hop neighbor table may have the following format to record these entries:

1.	N2_addr	N2_time	*next
2.	N2_addr	N2_time	*next
3.	„	„	„

Each entry in the table consists of N2_addr, N2_time, and *next which specifies that the node with address N2_addr is a 2-hop neighbor to this local node. * next is a pointer pointing to the next entry. Each 2-hop neighbor entry has an associated holding time N2_time, upon expiry of which it is no longer valid and hence removed.

3.2.2.3.3 MPR table

Each node selects its MPR and put MPR information in the MPR table. The information is recorded in the MPR table as a MPR entry. The MPR table may have the following format to record these entries:

1.	MPR_addr	time	*next
2.	MPR_addr	time	*next
3.	„	„	„

Each entry in the table consists of MPR_addr, time, and *next which specifies that the node with address MPR_addr is selected as MPR by this node. * next is a pointer pointing to the next entry. Each MPR entry has an associated holding time, upon expiry of which it is no longer valid and hence removed.

3.2.2.3.4 MPR Selector table

With information obtained from the HELLO messages, each node also constructs its MPR Selector table, in which it puts the addresses of its one-hop neighbor nodes, which have selected it as a multipoint relay. The MPR Selector table may have the following format to record the entries:

MSSN

1.	address	time	*next
2.	address	time	*next
3.	,,	,,	,,

A sequence number MSSN is associated to this table which specifies that the multipoint relay selector set of the local node keeping this MPR Selector table is most recently modified with the sequence number MSSN. The node modified its MPR Selector set according to the information it receives in the HELLO messages, and increments this sequence number on each modification.

3.2.2.3.5 Topology table

Each node of the network maintains a topology table, in which it records the information about the topology of the network obtained from the TC messages. Based on this information, the routing table is calculated. A node records information about the multipoint relays of other nodes in the network in its topology table as a topology entry, which may have the following format:

Seq_num

1.	T_ dest	T_last	T_seq	T_time	*next
2.	T_dest	T_last	T_seq	T_time	*next

3. „ „ „ „ „

Each entry in the table consists of T_dest, T_last, T_seq, T_time, and *next which specifies that the node T_dest has selected the node T_last as a multipoint relay and that T_last has announced this information of its multipoint relay selector set with the sequence number T_seq. Therefore, the node T_dest can be reached in the last hop through the node T_last. Each topology entry has an associated holding time T_time, upon expiry of which it is no longer valid and hence removed.

The topology table also contains a sequence number value Seq_num. Every time a node updates its topology table, this Seq_num is incremented to a higher value.

The entries in the topology table are recorded with the topology information that is exchanged among the network nodes through TC messages.

3.2.2.3.6 Routing table

The routing table is based on the information contained in the neighbor table and the topology table. Therefore, if any of these tables is changed, the routing table is re-calculated 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 R_next R_dist *next
2. R_dest R_next R_dist *next
3. „ „ „ „

Each entry in the table consists of R_dest, R_next, R_dist, and *next which specifies that the node identified by R_dest is estimated to be R_dist hops away from the local node, and that the one hop neighbor node with address R_next in the next hop node in the route to R_dest. Then new entries are recorded in the table for each destination in the network

for which the route is known. All the destinations for which the route is broken or partially known are not entered in the table.

3.2.3 Simulation model description

3.2.3.1 Network generator

The network generator will generate a random network for the simulation. It can generate any number of nodes within an arbitrarily determined area with random graph. In this simulation, we select to generate a random graph network with 50 nodes within 1000*1000 m^2 area. 30 nodes are stationary, while other 20 nodes are mobile. Every 8 s, e.g. 400 iterations, mobile nodes will change their direction and speed.

We use channel transmission rate 1Mbps and packet size 20000 bits. So the time to transmit one packet is 0.02 s. we call this one iteration and the program runs 10000 iterations in order to get a stable network performance.

3.2.3.2 Message generating

3.2.3.2.1 Hello generating

Each node generates one Hello message every 2 s. Each node contains a parameter called send Hello iteration. Whenever the current iteration modulo100 equals the parameter, node generates one Hello message.

3.2.3.2.2 TC generating

It is almost the same as Hello message except the period is 6 s instead 2 s.

Generating Data: random number generator will generate a number between 0 and 1. If the number is less than the load factor Lamda, then one data message is generated, otherwise, no message generated.

All the generated messages will be put into output buffer waiting for transmission. And input buffer will detect and receive message from channel. The messages in input buffer will be processed one by one according to their message type. Messages in both buffers will be processed or transmitted according to FIFO.

3.2.3.3 Neighbor sensing

Each node periodically broadcasts HELLO messages, containing the information about its neighbors and their link status. These Hello messages are transmitted in broadcast mode to all one-hop neighbors, but they are not relayed to further nodes. A HELLO message contains: The list of addresses of the neighbors with which there exists a valid bi-directional link.

If a node finds its own address in a HELLO message, it considers the link to the sender node as bi-directional (or MPR, if the sender node is also selected as an MPR).

These HELLO messages permit each node to discover its neighborhood up to two hops. On the basis of this information, each node selects its multipoint relays, which are used to flood the control messages. These selected multipoint relays are indicated in the HELLO messages with the link status MPR. Upon receiving these HELLO messages, each node can construct its MPR Selector table with the nodes, which have selected it as a multipoint relay.

3.2.3.4 Multipoint relay selection

Each node of the network independently selects its own set of multipoint relays. The MPR set is calculated to contain a subset of the one-hop neighbors, which covers all the two, hop neighbors. By this we mean that the union of the neighbor sets of all MPRs contains the entire two-hop neighbor set. In order to build the list of the two hop nodes

from a given node, it suffices to track the list of bi-directional link nodes found in the HELLO messages received by this node (this two-hop neighbor information is recorded in the neighbor table). The MPR set needs not to be optimal, however it should be small enough to achieve the benefit of the multipoint relays. Multipoint relays of a given node are declared in the subsequent HELLOs transmitted by this node, so that the information reaches the multipoint relays themselves. The multipoint relay set is re-calculated when:

- A change in the neighborhood is detected or
- A change in the two-hop neighbor set is detected.

Remark: The multipoint relays technique is self-optimized. It is not essential that the multipoint relay set be minimal or optimal. But it is essential that it covers the two hop nodes.

3.2.3.5 Multipoint relay information declaration

In order to build the Intra-forwarding database needed for routing the messages, each relay node broadcasts specific service messages called Topology Control (TC) messages. TC messages are forwarded like usual broadcast messages to all nodes in the network. This technique is related to the link state technique, taking advantage of multipoint relays. Multipoint relays enable a better scalability of Intra-forwarding.

A TC message is sent by each node in the network at regular intervals to declare its MPR Selector set, i.e., the message contains the list of neighbors, which have selected the sender node as a multipoint relay. The sequence number (MSSN) associated to this multipoint relay selector set is also attached to the list. The information diffused in the network by these TC messages will help each node to calculate its routing table. A node

which has an empty MPR Selector set, i.e., nobody has selected it as a multipoint relay, will not generate any TC message.

Upon receipt of a TC message, the following procedure is executed to record the information in the topology table (See also section 3.2.2.3.5 Topology table for reference):

1. If there exists some entry in the topology table whose T_last corresponds to the originator address of the TC message and whose T_seq is greater in value than the MSSN in the received message, then no further processing of this TC message is done and it is silently discarded (case: packet received out of order).

2. If there exists some entry in the topology table whose T last corresponds to the originator address of the TC message and whose T_seq is smaller in value than the MSSN in the received message, then that topology entry is removed.

3. For each of the MPR Selector address received in the TC message:

   - If there exists some entry in the topology table whose T_dest corresponds to the MPR Selector address and the T_last corresponds to the originator address of the TC message, then the holding time T time of that topology entry is refreshed.

   - Otherwise, a new topology entry is recorded in the topology table whereas:

     - T_dest is set to the MPR Selector address,

     - T_last is set to the originator address of the TC message,

     - T_seq is set to the value of MSSN received in the TC message,

     - T_time is set to the holding time of a topology entry.

3.2.3.6 Routing table calculation

Each node maintains a routing table which allows it to route messages to the other destinations in the network. The routing table is re-calculated locally each time the neighbor table or the topology table or both are changed. The update of this routing table does not generate or trigger any messages to be transmitted, neither in the network nor in the one-hop neighborhood.

The following procedure is executed to calculate (or re-calculate) the routing table:

1. All entries in the routing table are removed.
2. The new entries are recorded in the table starting with the one-hop neighbors (h = 1) as the destination nodes. For each neighbor entry in the neighbor table, whose link status is not unidirectional, a new route entry is recorded in the routing table where R_dest and R_next are both set to the address of the neighbor and R_dist is set to 1.
3. Then the new route entries for the destination nodes h + 1 hops away are recorded in the routing table. The following procedure is executed for each value of h, starting with h = 1 and incrementing it by 1 each time. The execution will stop if no new entry is recorded in iteration.
    a. For each topology entry in the topology table, if its T_dest does not correspond to R_dest of any route entry in the routing table AND its T last corresponds to R_dest of a route entry whose R dist is equal to h, then a new route entry is recorded in the routing table where :
    b. R_dest is set to T_dest;
    c. R_next is set to R_next of the route entry whose R_dest is equal to T_last;
    d. R_dist is set to h + 1.

4 OLSR simulation results and evaluation

4.1 Network performance under different sending control message period

Control messages are Hello and TC messages. In the simulation of this protocol, the period of sending TC message is 3 times the period of sending hello message. Fig. 4-1 shows the "buffer full" probability as the sending hello message period increases.

We have 50 nodes in the network, each with 2 buffers. The total number of buffers is 100. The "buffer full" probability decreases as sending control message period increases. This is because less control messages are sent as sending period increases. So the number of messages in the buffer will be less, and the buffers are less filled. We use 100*0.02=2 s as sending hello message period in the rest of simulation. So sending TC message period is 300*0.02=6 s. In this case, and in the steady state (Number of iterations is 180 in Fig.4-1) the buffer full probability is 0.04 meaning 4 out of 100 buffers will be full. From Fig.4-2 we can see that the average number of messages in buffer is 1.5. Fig. 4-2 also shows large value of variance of number of messages in a typical buffer. This is so, because the numbers of messages in the full buffers are far above this average number 1.5. Fig.4-1 and 4-2 give the general picture about all the nodes buffers. 4 out of 100 total buffers are full, and others are almost empty, also the average number of messages in buffer is very low, which is only 1.5.

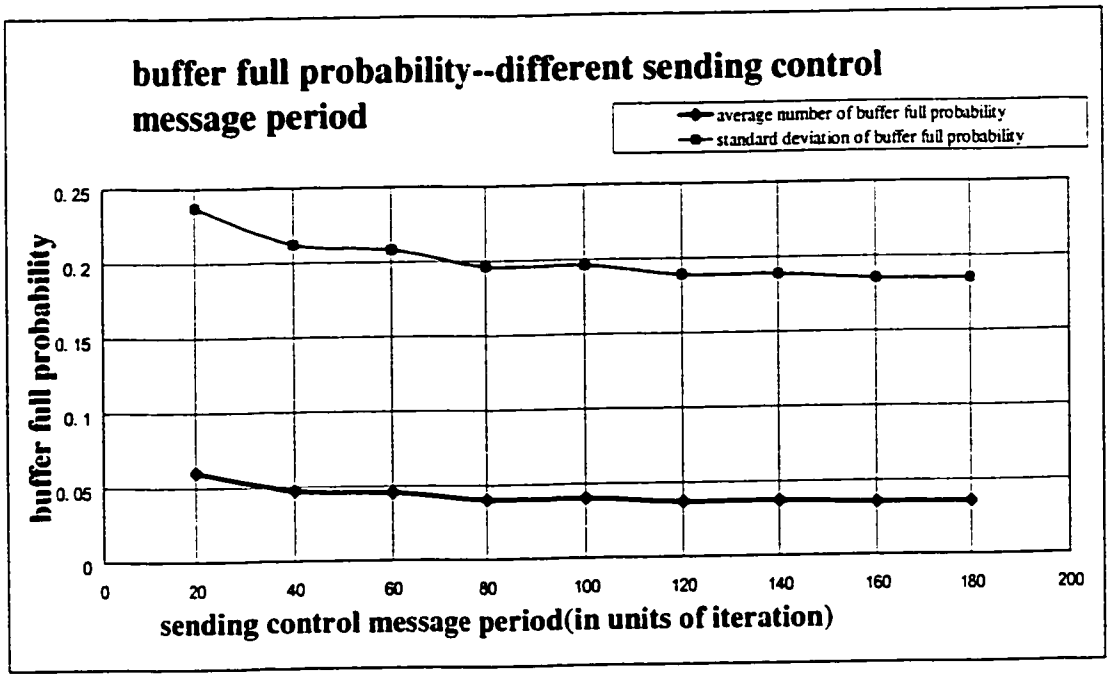


Figure 4-1 buffer full probability—different sending control message period

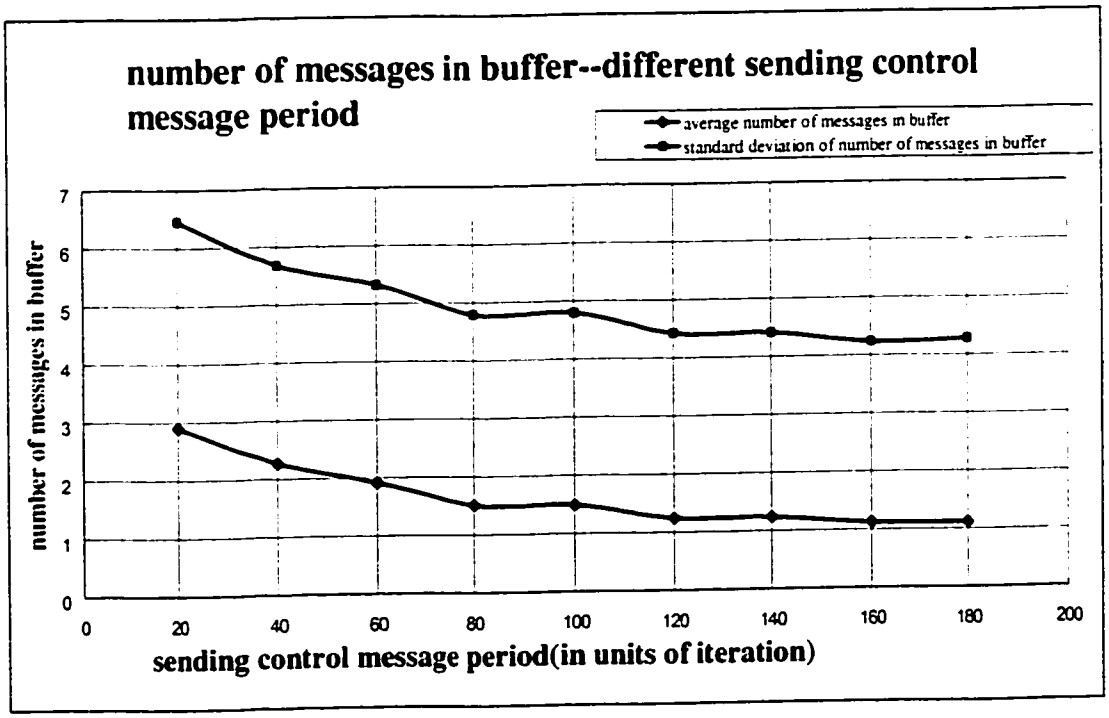


Figure 4-2 messages in buffer—different sending control message period

Test conditions:
Total node No. 50;moving nodes 20;Max simulation iteration 10000;Max node speed 108km/h; Max data transfer Hops=10; data Timeout 0.6s; load parameter Lamda=0.125.

OLSR protocol uses MPR to minimize flooding of control messages. The loads in these MPRs are much higher than in normal nodes, so some MPRs may become connection bottlenecks in the network topology graph. So these MPRs should be better equipped to get good performance in the whole network by using the OLSR protocol.

The average messages in buffer decreases as sending control message period increases, because less control messages will be sent.

As shown in Fig.4-3, data transfer hops increases as sending control message period increases. Because less control messages will be sent when sending period increases. If less control messages are sent, topology and routing information will be less clear and less accurate. So data messages have to travel more hops along non-optimal routes to the destination.

When the sending period is less than 60 iterations, the average number of hops varies largely and deviation of number of hops is very large, so routes are not stable and sometimes non-optimal. When sending period is more than 100, the average number of hops is large. So it's better to choose sending period between 60 and 100 for the specific load and other parameters given in Fig.4-3.

In Fig. 4-4, the throughput increases as sending control period increases, because increasing of sending period will generate less control messages for topology and routing update which leads to less congestion and message losses in the network. So more date messages can successfully reach their destinations and throughput increases.

The decreasing of control message sending period leads to increasing the overhead. But when the sending period is more than 100, overhead decreases slowly. From Fig. 4-3 we already know that it is better to choose sending period between 60 and 100.

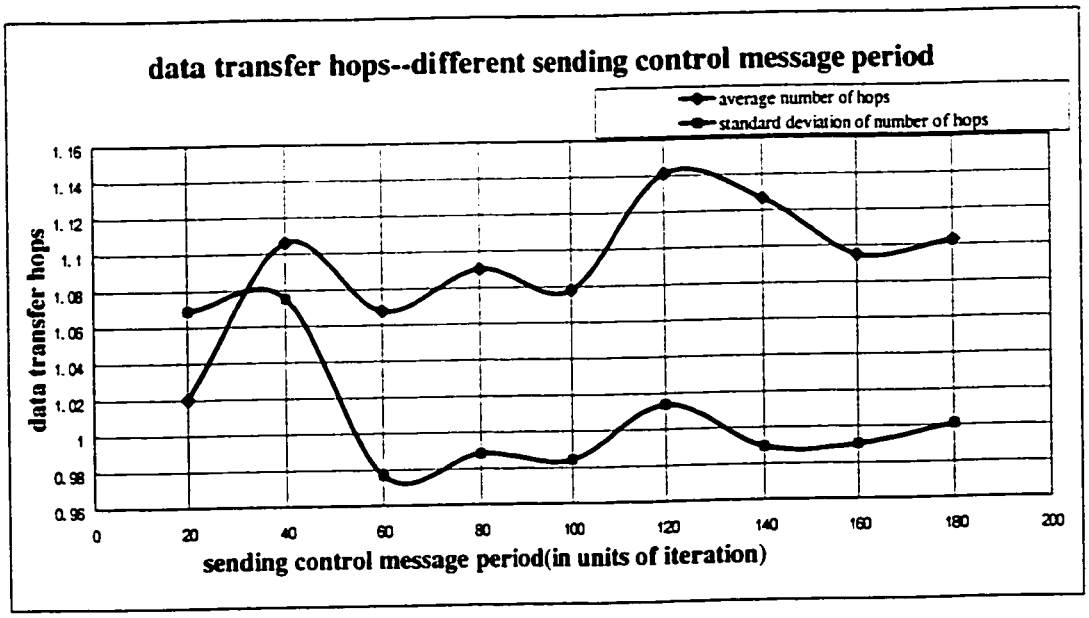


Figure 4-3 data transfer hops—different sending control message period

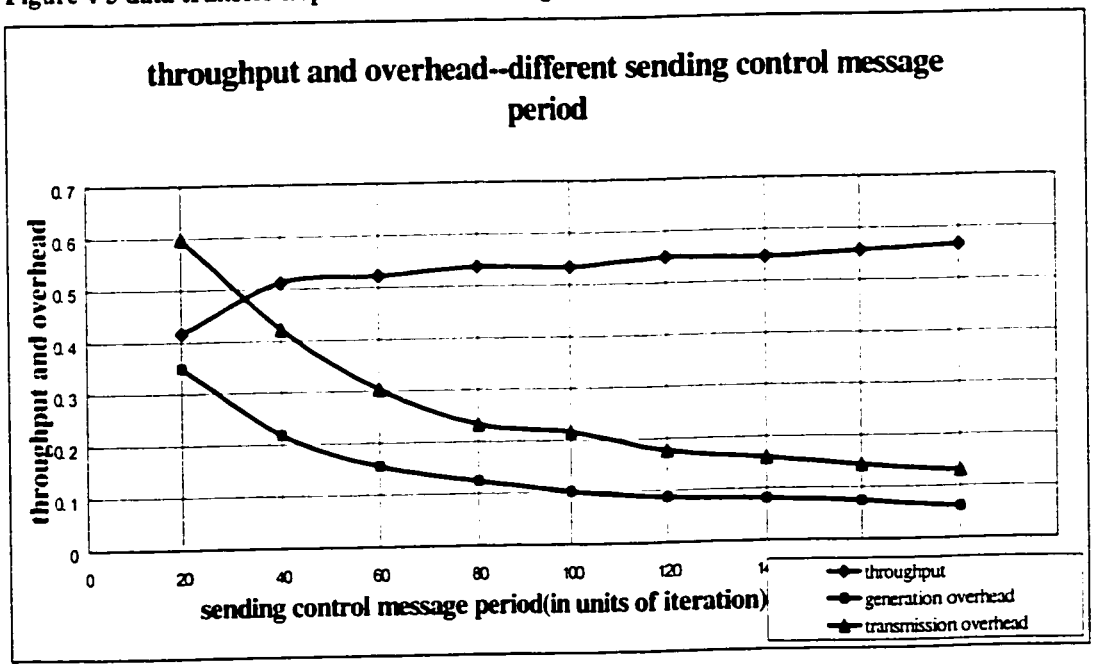


Figure 4-4 throughput and overhead—different sending control message period

Test conditions:
Total node No. 50;moving nodes 20;
Max simulation iteration 10000;Max node speed 108km/h; Max data transfer Hops=10; data Timeout 0.6s; load parameter Lamda=0.125.

So finally we can choose 100 as the sending control message period to get the best performance for the set of traffic value and other parameters indicated.

Fig. 4-5 shows that if the sending period increases the queuing delay also increases. When the sending period increases, control messages will be sent less frequently, and less control messages causes data messages to travel more hops as the network lacks topology and routing information. More messages will wait in the buffers, so the queuing delay increases. When sending period is 100, queuing delay is 6*0.02=0.12 s.

In Fig. 4-6, congestion and number of control messages decrease while sending period increases. So transfer delay decreases. When sending period is 100, transfer delay is 5 * 0.02 = 0.1 s.

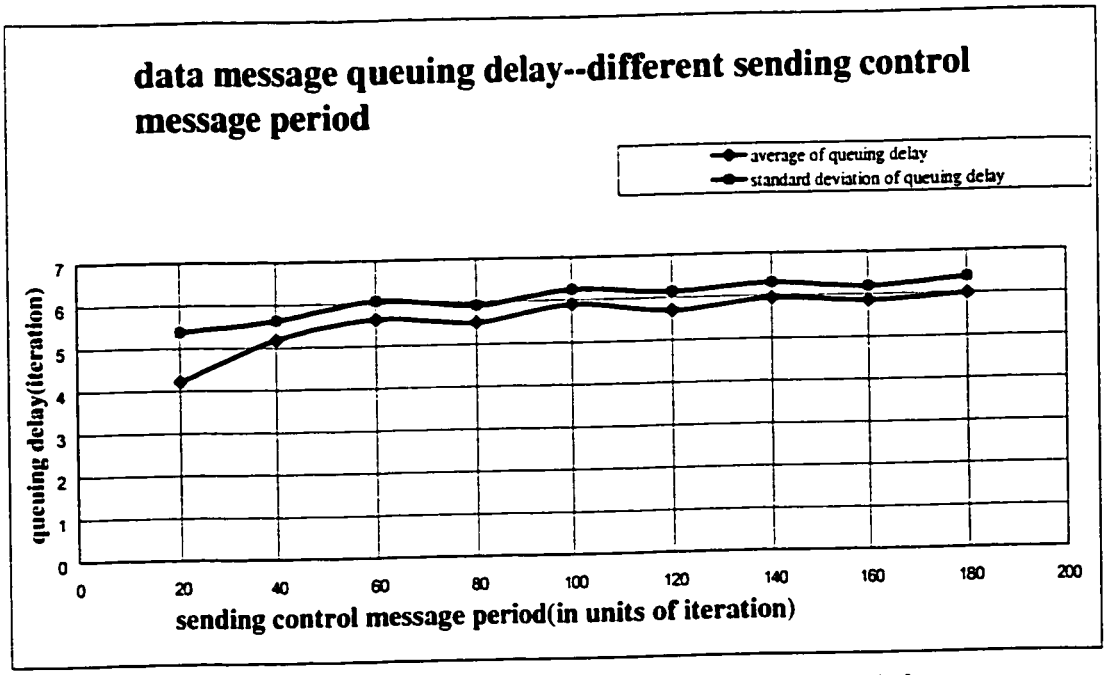


Figure 4-5 data message queue delay—different sending control message period

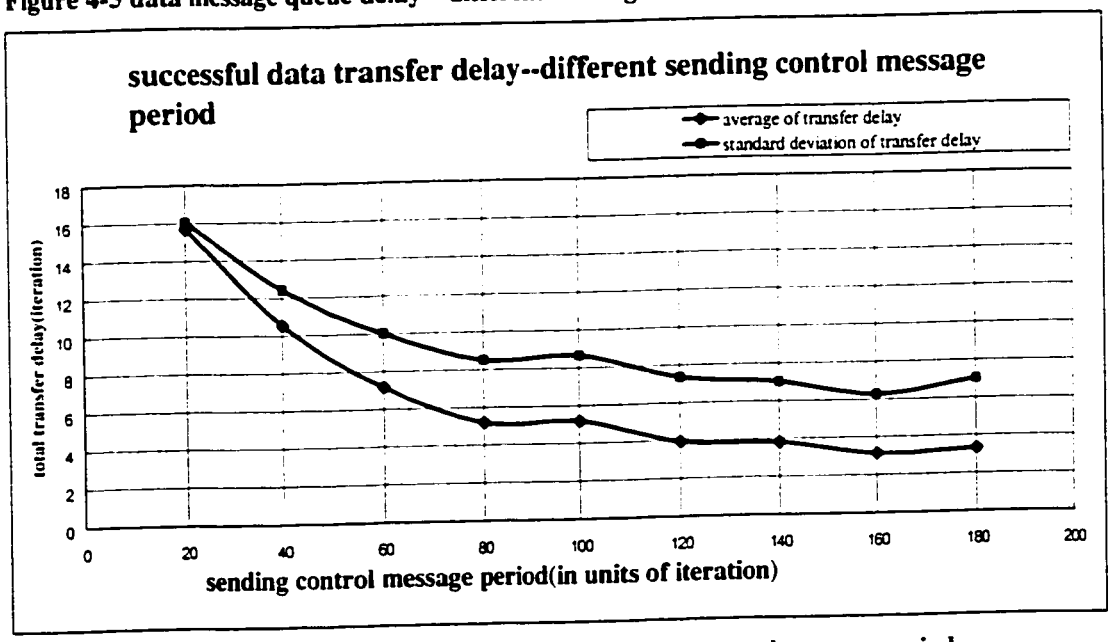


Figure 4-6 successful data transfer delay—different sending control message period

Test conditions:
Total node No. 50;moving nodes 20;
Max simulation iteration 10000;Max node speed 108km/h; Max data transfer Hops=10;
data Timeout 0.6s; load parameter Lamda=0.125.

4.2 Network performance under different data time out

Fig. 4-7 shows that the buffer full probability increases as data time out increases. When data timeout is less than 200, the number of full buffers is relatively small, about 4 or 5, but if data timeout is more than 200, the number of such buffers is relatively large, about 7 to 9. The reason is quite simple. Data timeout is the maximum time that data can exist in the output buffer. So there will be more messages in the buffer i.e., buffers will be more probably full as data timeout gets larger.

The average number of messages in buffers increases slowly as data timeout increases as shown in Fig. 4-8. The number is very small (around 2). Because 5% of the buffers are full and other buffers are almost empty.

In Fig. 4-9, we can see that data transfer number of hops decrease by increasing data timeout when data timeout is less than 200. Because more buffers get to be full when timeout increases, so data messages have to wait longer, and some messages will be dropped. The longer the route for the data message is, the worst it gets. So more and more messages with longer routes are dropped. This leads to the decreasing of average number of data transfer hops.

When data timeout is more than 200, the number of transfer hops doesn't vary much. Because more nodes have full buffers, time out does not have noticeable effects.

In Fig. 4-10, throughput decreases while increasing data timeout, because less successful data can reach destinations while more buffers get full by increasing timeout. But throughput doesn't vary much when timeout is more than 200 as more buffers are full. The generation overhead is almost a constant, because it's non-related to the data time out.

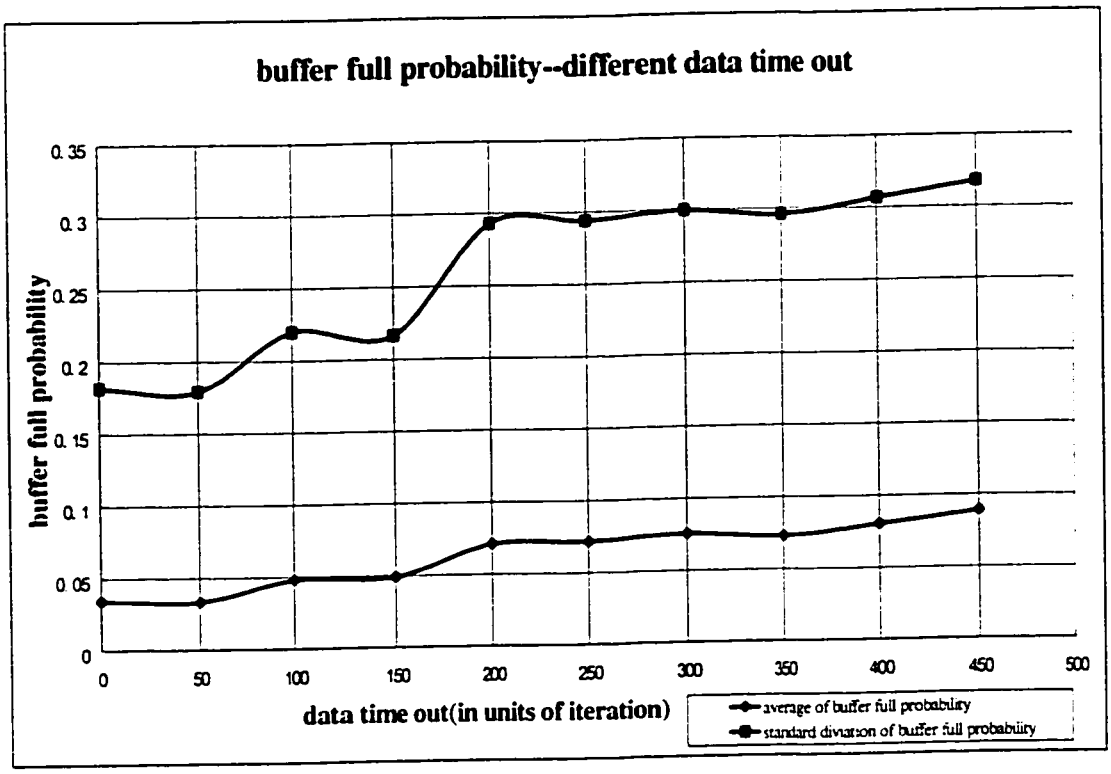


Figure 4-7 buffer full probability—different data time out

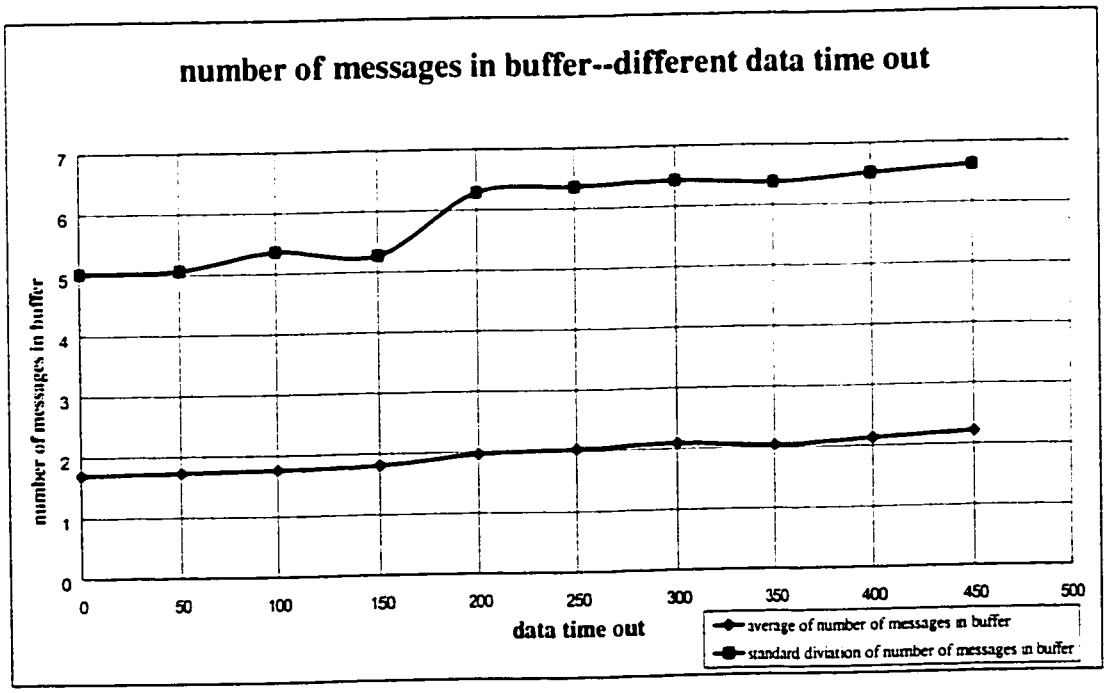


Figure 4-8 messages in buffer—different data time out

Test conditions:
Total node No. 50;moving nodes 20;
Max simulation iteration 10000;Max node speed 108km/h; Max data transfer Hops=10;
load parameter Lamda=0.125; send hello Period 2s; send TC Period 6s.

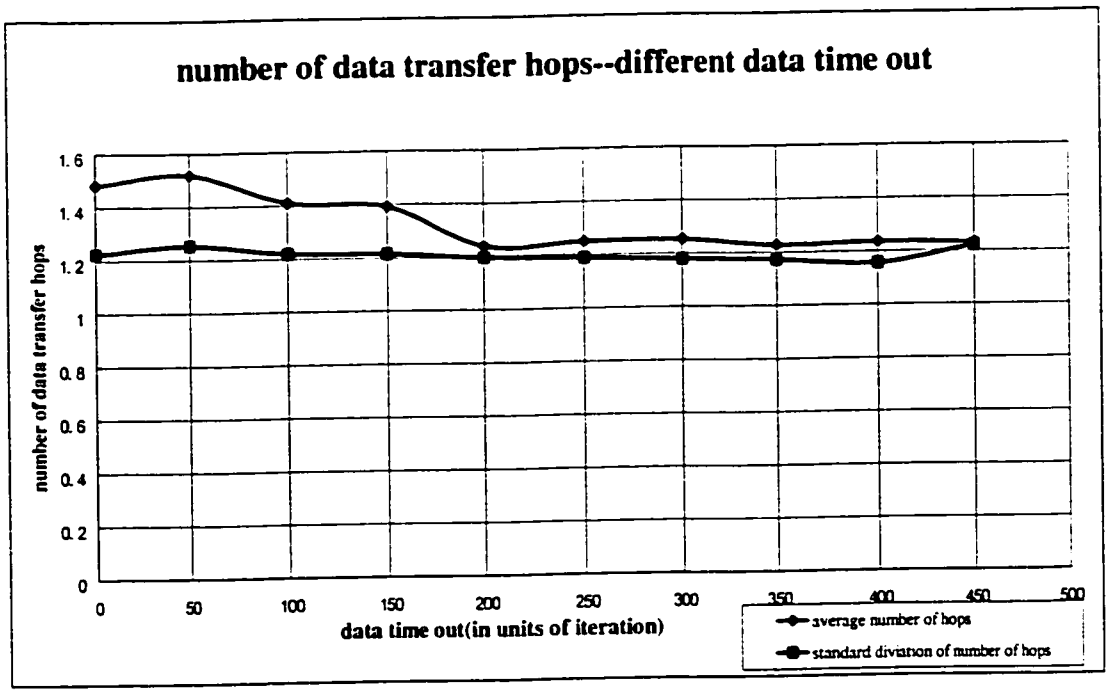


Figure 4-9 data transfer hops—different data time out

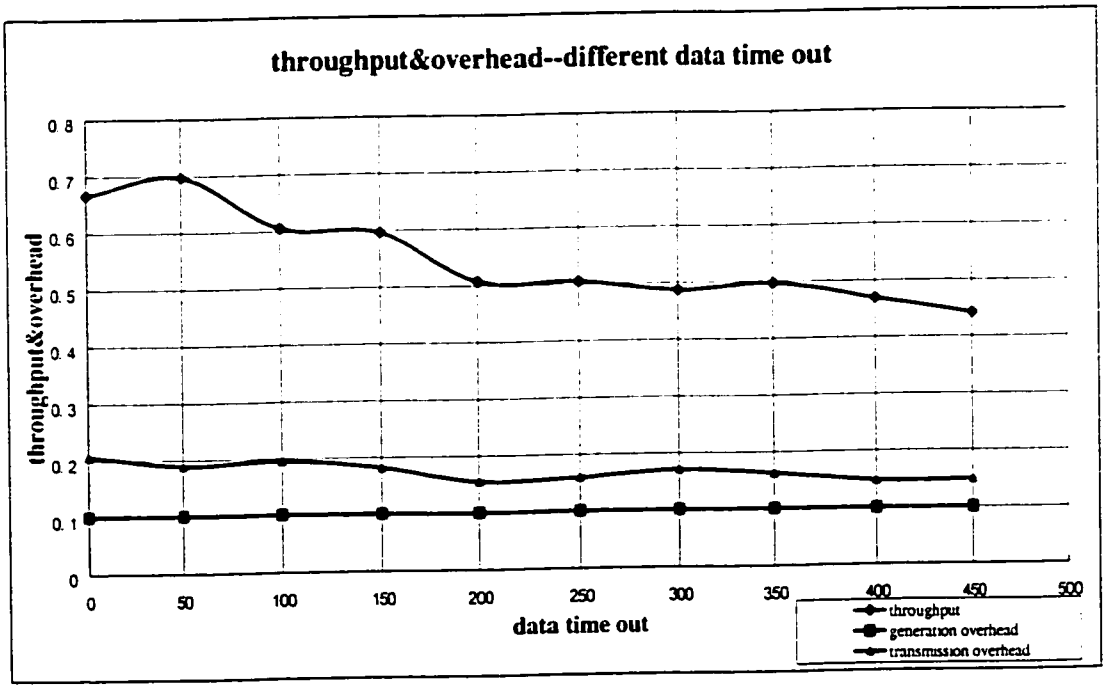


Figure 4-10 throughput and overhead—different data time out

Test conditions:
Total node No. 50;moving nodes 20;
Max simulation iteration 10000;Max node speed 108km/h; Max data transfer Hops=10; load parameter Lamda=0.125; send hello Period 2s; send TC Period 6s.

Less control messages will be transmitted while more buffers become full. So the transfer overhead decreases with data time out increasing.

In Fig. 4-11, when data time out is less than 100, queuing delay increases rapidly with time out and when time out is more than 100, it doesn't vary much. This is because time out is the main reason of data queuing in output buffers. When buffers become full, timeout doesn't have much influence. Because buffer status is then either full or almost empty.

In Fig. 4-12, when data timeout is less than 200, data transfer delay decreases rapidly and it doesn't vary much when time out is larger than 200. This is the same as number of transfer hops.

It's interesting to see that queuing delay increases and transfer delay decreases when data time out is less than 100. This is because number of transfer hops is the more relevant factor with respect to transfer delays.

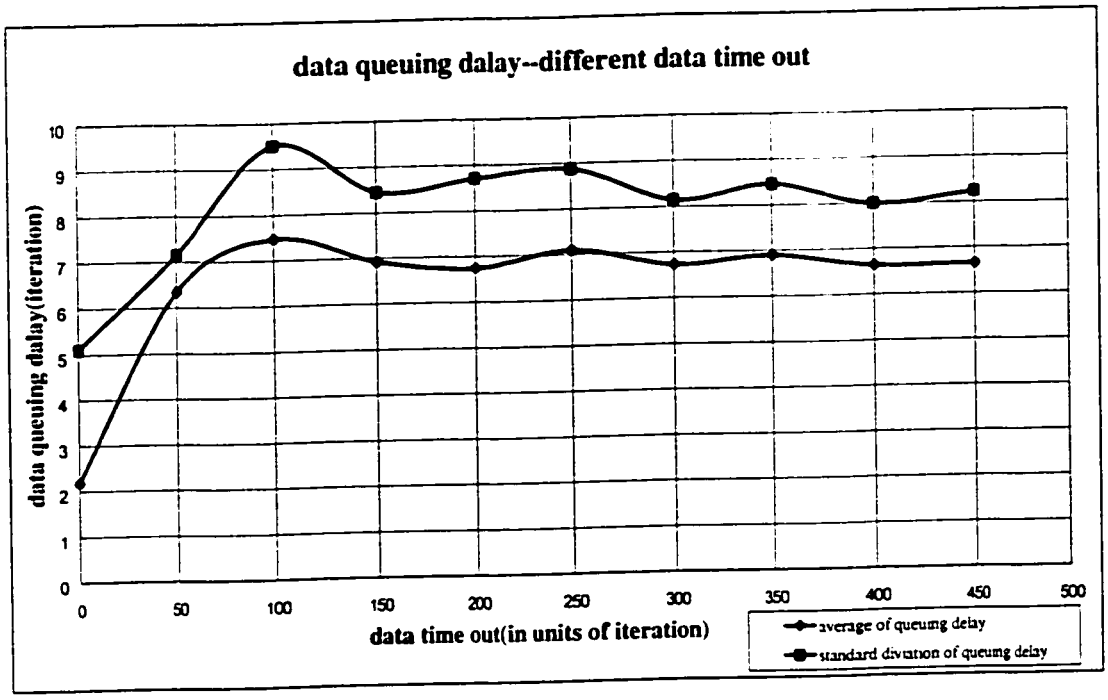


Figure 4-11 data queue delay—different data time out

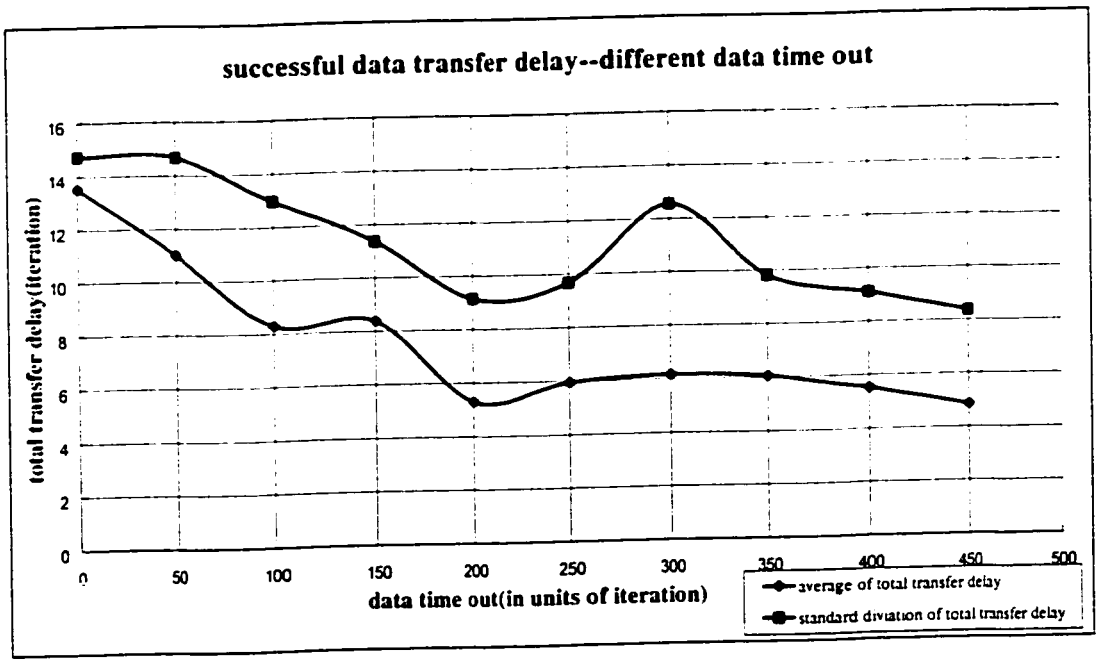


Figure 4-12 successful data transfer delay—different data time out

Test conditions:
Total node No. 50;moving nodes 20;
Max simulation iteration 10000;Max node speed 108km/h; Max data transfer Hops=10;
load parameter Lamda=0.125; send hello Period 2s; send TC Period 6s.

4.3 Network performance under different entry hold time

Entry hold time is referred to as neighbor table entry hold time, and topology table entry hold time is 3 times longer than this entry hold time. Other table entries are calculated on the basis of these two tables. So after entry-hold time for neighbor table is determined, all entry hold time of other tables are automatically determined.

When entry-hold time is less than 280, the number of full buffers is larger and when hold time is more than 280, the number of full buffers tends to be constant and stable. The average number of messages in the buffers doesn't vary much. These are shown in Fig. 4-13 and 4-14.

The reason of choosing entry hold time as 300 is explained in Fig. 4-15. When entry-hold time is less than 300, data transfer hops decrease because table entries can exist longer in tables so data messages have a higher probability of finding a route. When entry hold time is more than 300, number of hops increase as entry hold time increases, because some entries are no longer valid in the tables. These invalid routes force data messages to travel longer along non-optimal routes. Because OLSR is hop-by-hop routing, which means data messages may travel some unnecessary hops when entries hold time is longer and some routes are invalid.

In Fig. 4-16, when entry hold time is less than 280, throughput fluctuates and is relatively low. When entry hold time is more than 260 but less than 350, throughput is stable and when entry hold time is larger than 350, throughput increases slowly but this is gained on the expense of losing the performance of number of transfer hops. The generation overhead is relatively a constant and transfer overhead varies very little.

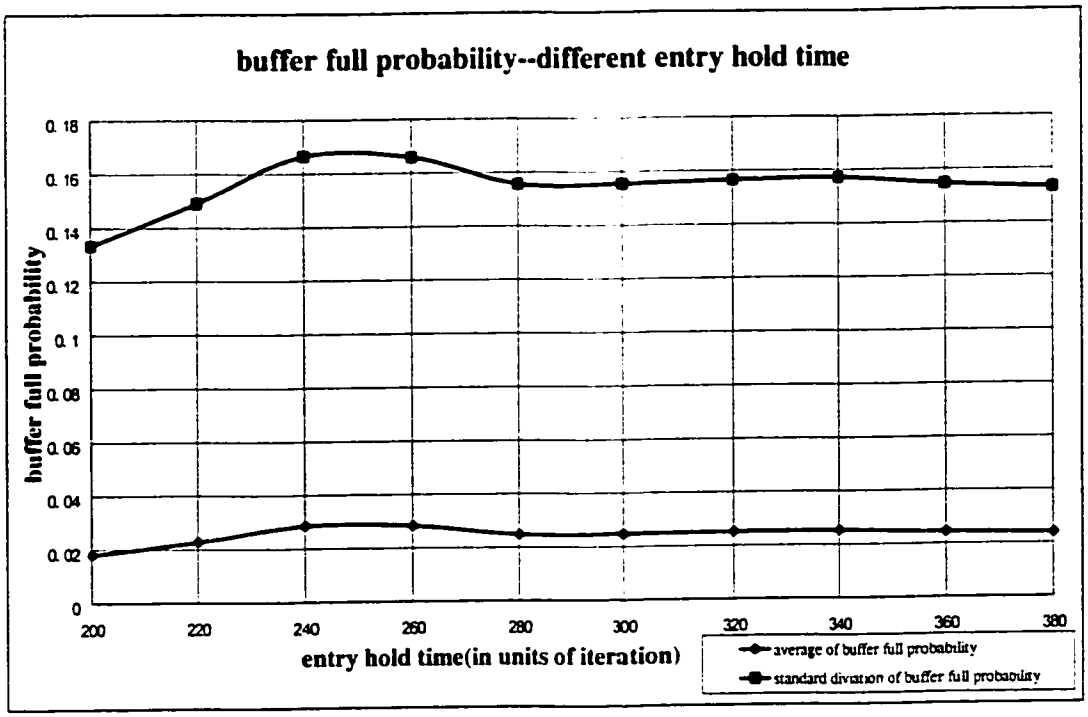


Figure 4-13 buffer full probability—different entry hold time

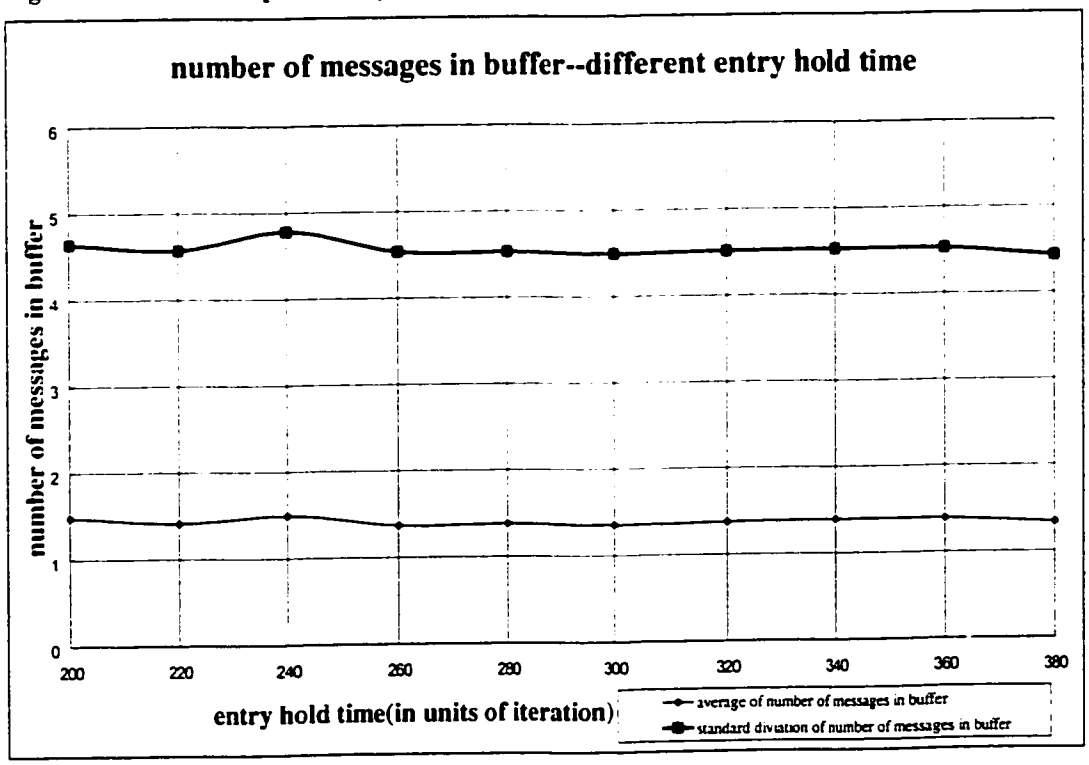


Figure 4-14 messages in buffer—different entry hold time

Test conditions:
Total node No. 50;moving nodes 20;
Max simulation iteration 10000;Max node speed 108km/h; Max data transfer Hops=10;
data Timeout 0.6s; load parameter Lamda=0.125; send hello Period 2s; send TC Period 6s.

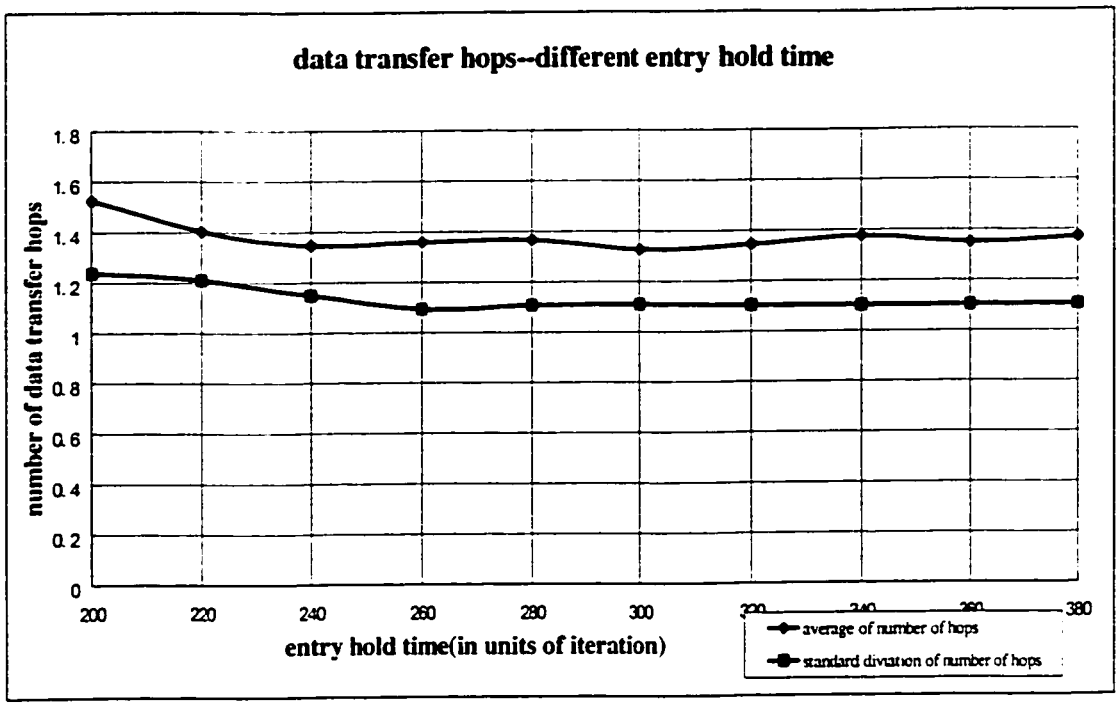


Figure 4-15 data transfer hops—different entry hold time

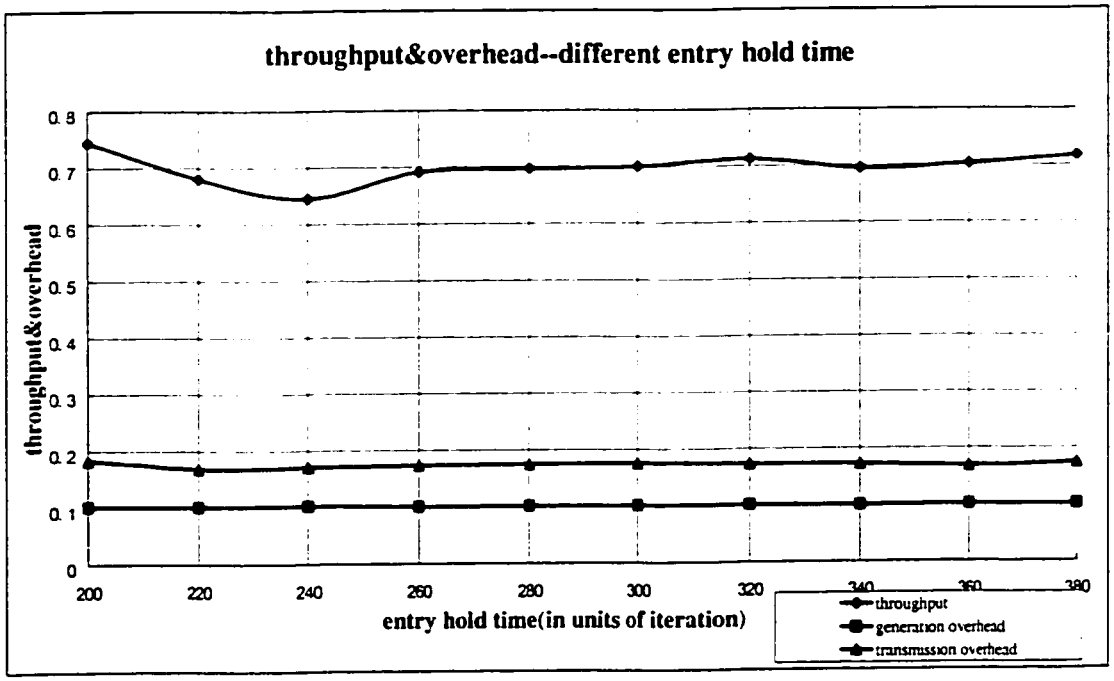


Figure 4-16 throughput and overhead—different entry hold time

Test conditions:
Total node No. 50;moving nodes 20;
Max simulation iteration 10000;Max node speed 108km/h; Max data transfer Hops=10; data Timeout 0.6s; load parameter Lamda=0.125; send hello Period 2s; send TC Period 6s.

In Fig. 4-17, data queuing delay decreases as entry hold time increases. When entry hold time increases, it's easier to find a route although some routes may be invalid. But data queue in the buffer only when it cannot find a route. So queuing delay will decrease.

In Fig. 4-18, data transfer delay decreases when entry hold time is less than 260 and increases slowly when entry hold time is more than 260. This is almost the same as number of transfer hops. Because the transfer hops is the main factor affecting transfer delays.

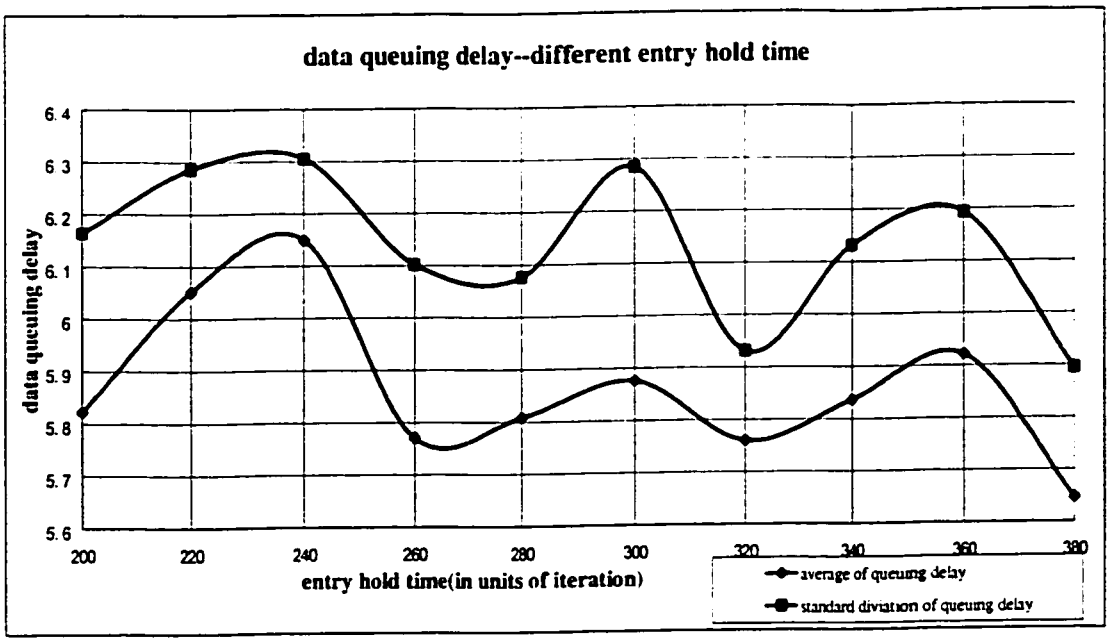


Figure 4-17 data queue delay—different entry hold time

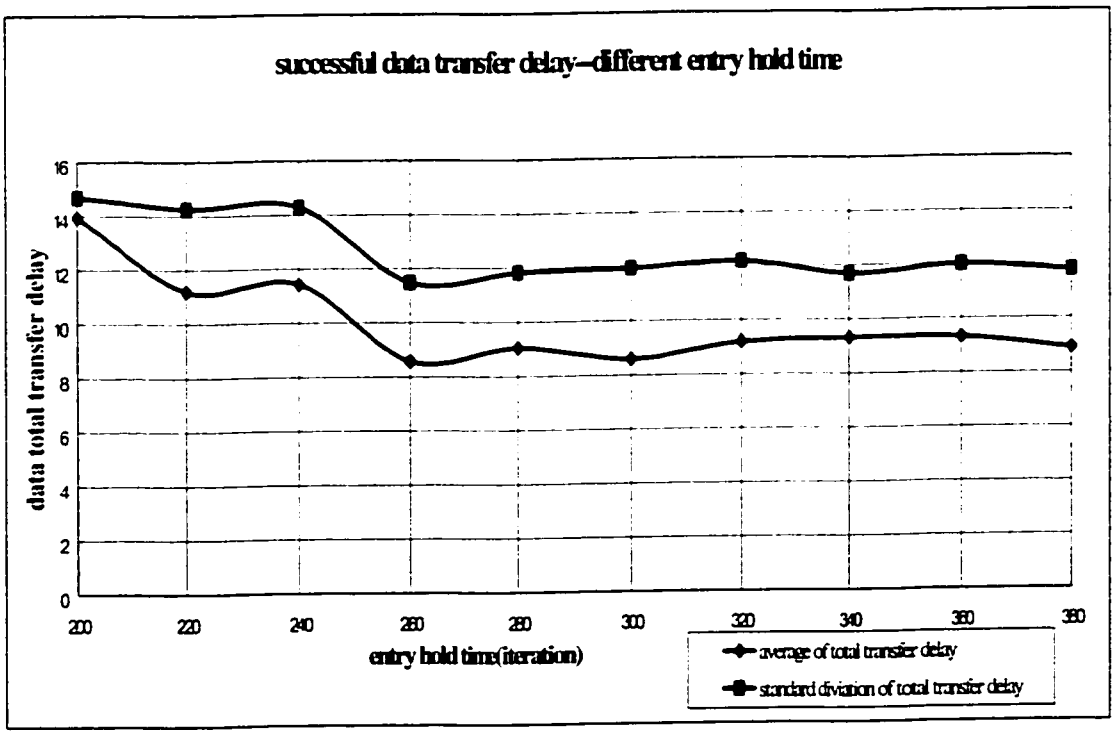


Figure 4-18 successful data transfer delay—different entry hold time

Test conditions:
Total node No. 50;moving nodes 20;
Max simulation iteration 10000;Max node speed 108km/h; Max data transfer Hops=10;
data Timeout 0.6s; load parameter Lamda=0.125; send hello Period 2s; send TC Period 6s.

4.4 Network performance under different load condition

When the parameter Lamda, which represents load condition, increases, the buffer full probability and average number of messages in buffer also increase as shown in Fig. 4-20.This is so, because more data will be generated and stalled in buffers with increasing Lamda.

In Fig. 4-21, number of data transfer hops decreases with Lamda increasing. The reason is that more data are generated and congestion develops with increasing Lamda. Messages can reach nearer destination easily but messages with further destinations are more likely to be lost. The average number of transfer hops then decreases when averaging among all the successfully transmitted data.

In Fig. 4-22, the throughput and overheads both decrease with increasing Lamda. This is so because more data and heavier congestion lead to decreasing throughput and overheads. The congestion also leads to total data transfer delay increasing as shown in Fig. 4-24.

It's interesting to observe the linear decrease of queuing delay as Lamda increases. 4–5 out of 100 buffers are full and messages queue for longer times. Other buffers are almost empty, so messages need very short queuing time. When Lamda is small, less data is generated and relatively more data is transmitted through full buffers. So queuing delay is larger when averaging among all data. When Lamda gets larger, more data are generated and relatively less data are transmitted through full buffers because of lost messages. So queuing delay decreases as shown in Fig. 4-23.

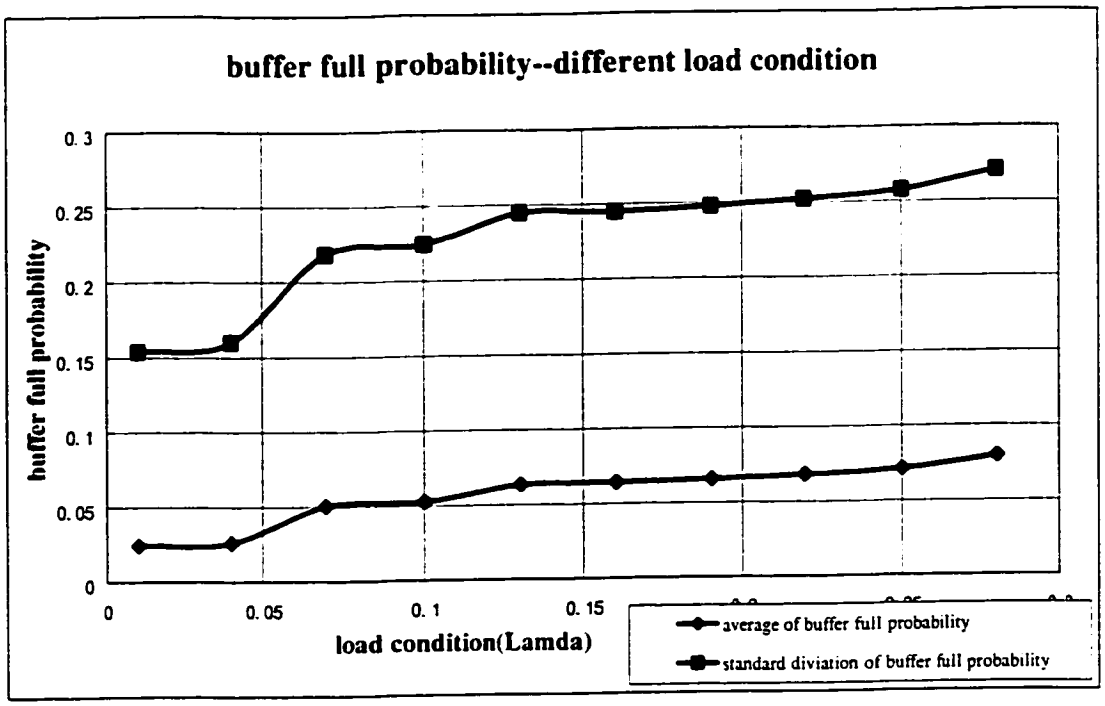


Figure 4-19 buffer full probability—different load condition

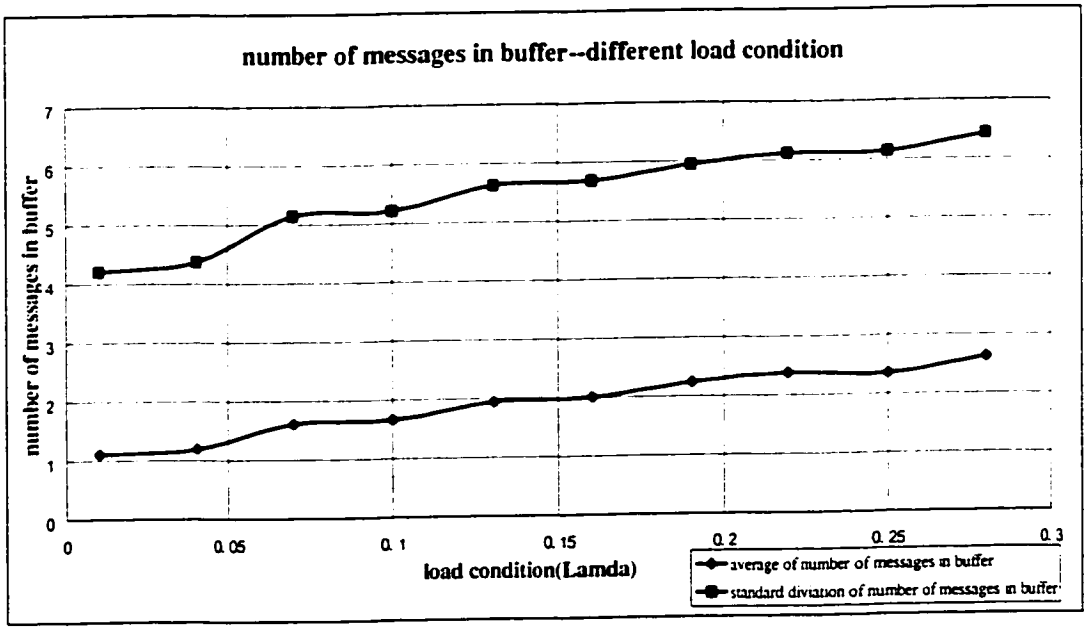


Figure 4-20 messages in buffer—different load condition

Test conditions:
Total node No. 50;moving nodes 20;
Max simulation iteration 10000;Max node speed 108km/h; Max data transfer Hops=10; data Timeout 0.6s; send hello Period 2s; send TC Period 6s.

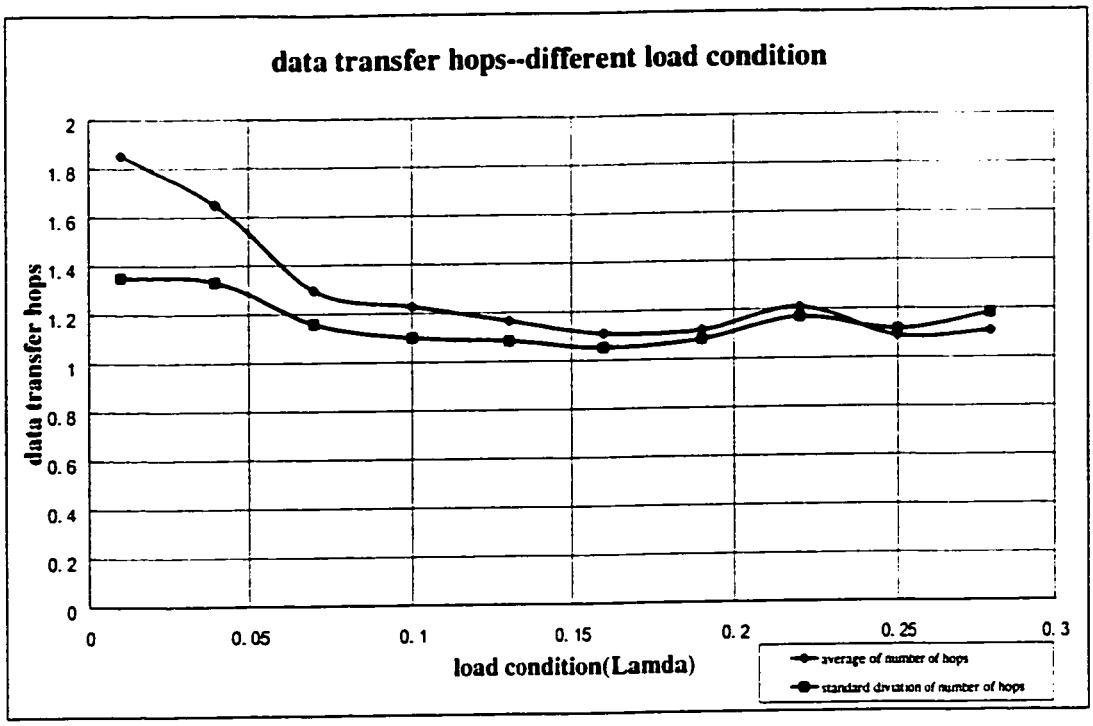


Figure 4-21 data transfer hops—different load condition

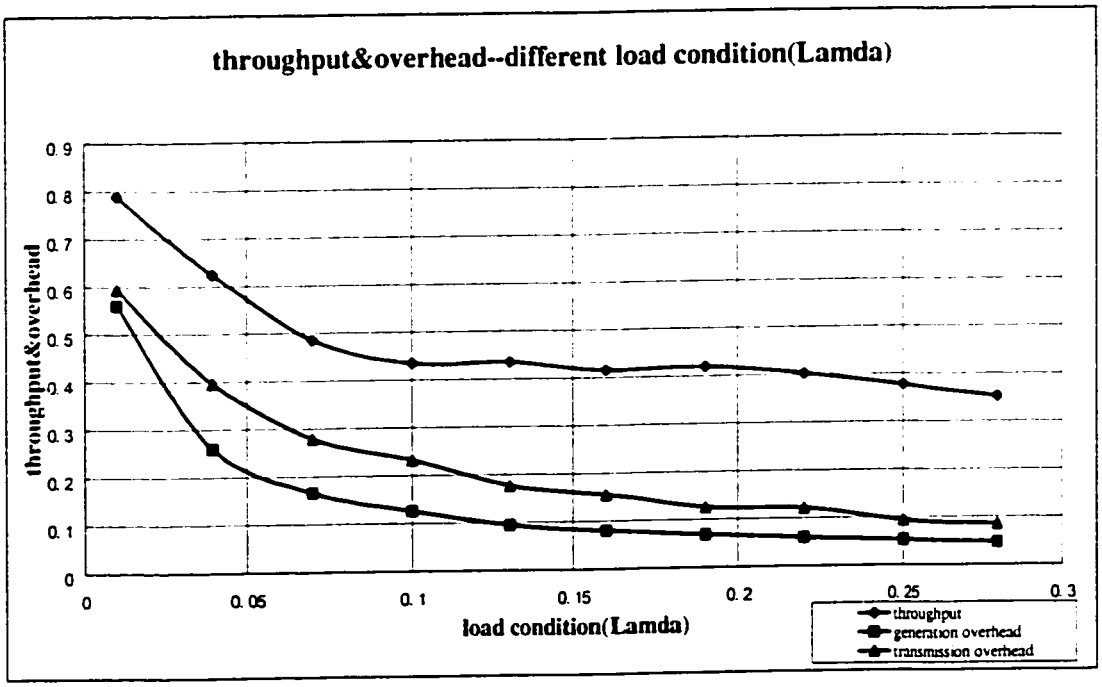


Figure 4-22 throughput and overhead—different load condition

Test conditions:
Total node No. 50;moving nodes 20;
Max simulation iteration 10000;Max node speed 108km/h; Max data transfer Hops=10; data Timeout 0.6s; send hello Period 2s; send TC Period 6s.

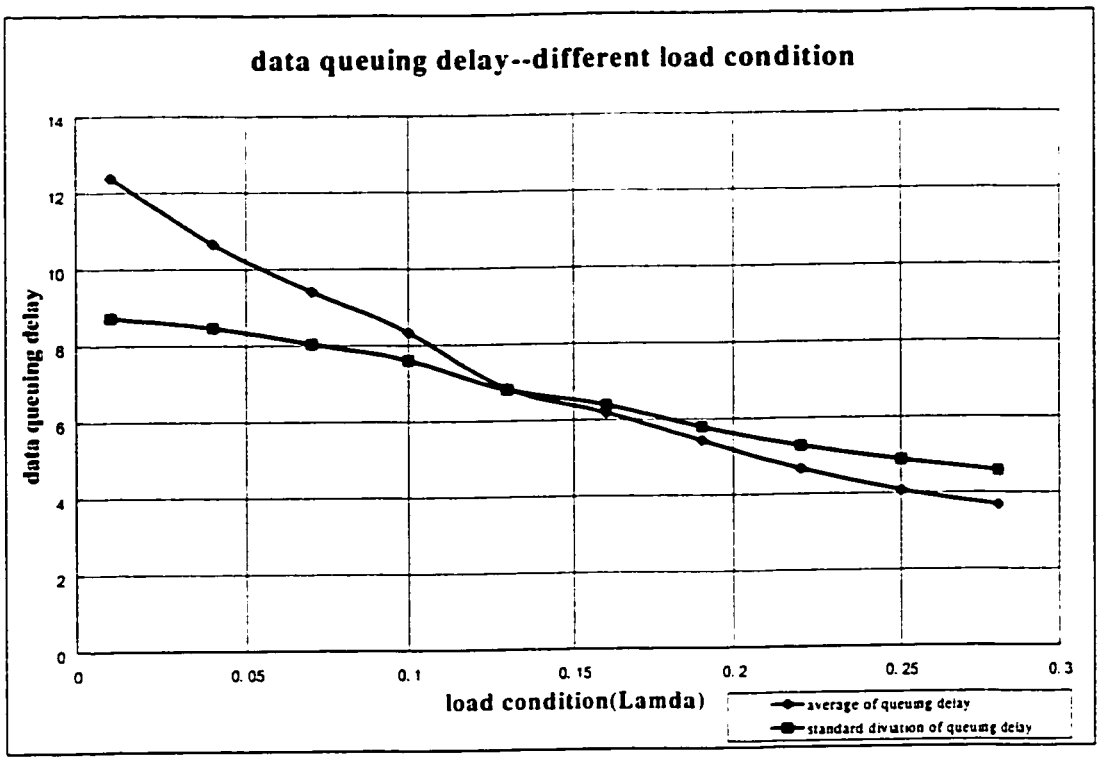


Figure 4-23 data queuing delay—different load condition

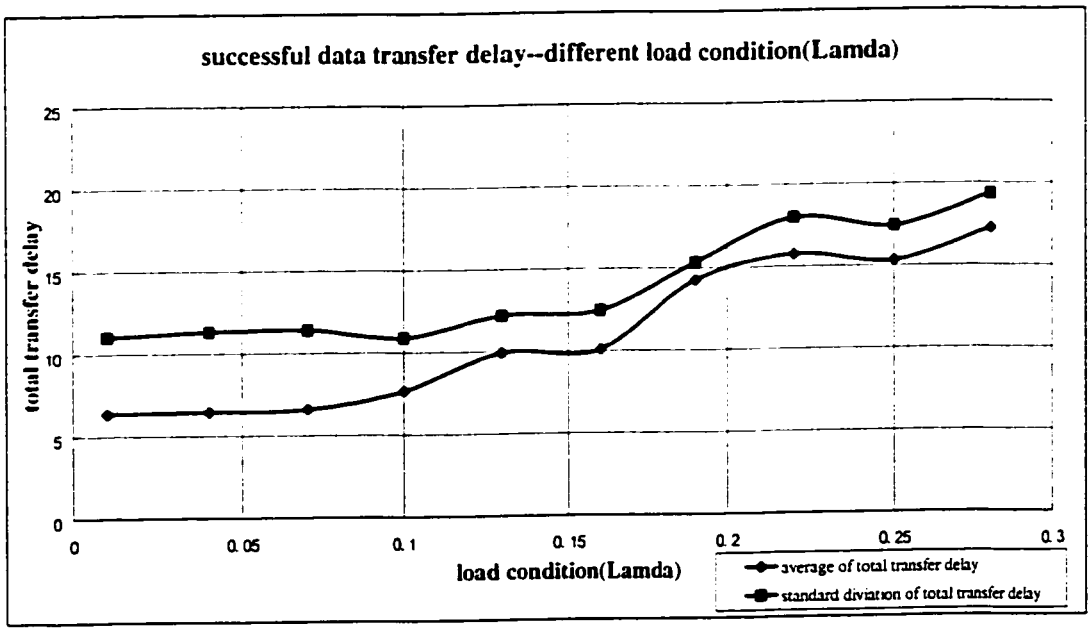


Figure 4-24 successful data transfer delay—different load condition

Test conditions:
Total node No. 50;moving nodes 20;
Max simulation iteration 10000;Max node speed 108km/h; Max data transfer Hops=10; data Timeout 0.6s; send hello Period 2s; send TC Period 6s.

4.5 Network performance under different data transfer number of hops

Buffer full probability increases with increasing the data transfer maximum hop. The reason is that more messages can travel longer in the net when increasing maximum hop. This leads to the increasing of not only buffer full probability but also the number of messages in the buffers. The average number of messages in buffers only increases very slowly. As are shown in Figs. 4-25 and 4-26.

In Figs. 4-27 and 4-28, when the maximum hop count is less than 5, more data can reach their destinations along routes of more hops. So the throughput and transfer hops increase with increasing maximum hop. When maximum hop is between 5 and 9, the performance of throughput and transfer hops is stable, as routes are not too long in the network of 50 nodes. The value of maximum hop makes no difference within a certain range. But if maximum hop value is too large (>9), the performance will be worse. This is because some data messages with long routes cause heavy congestion and affect successful transmission of other data messages.

The generation overhead is constant, it's not related to Max hop count. Less data are lost due to increasing Max hop. The number of all successful transmitted messages will increase. So the transmission overhead decreases slightly.

Max hop count has not much influence on data queuing delay, as shown in Fig. 4-29. In Fig. 4-30 we can again divide it into 3 areas. When Max hop is less than 3, success data transfer delay increases and maintains stability between 3 and 9, and decreases when larger than 9.

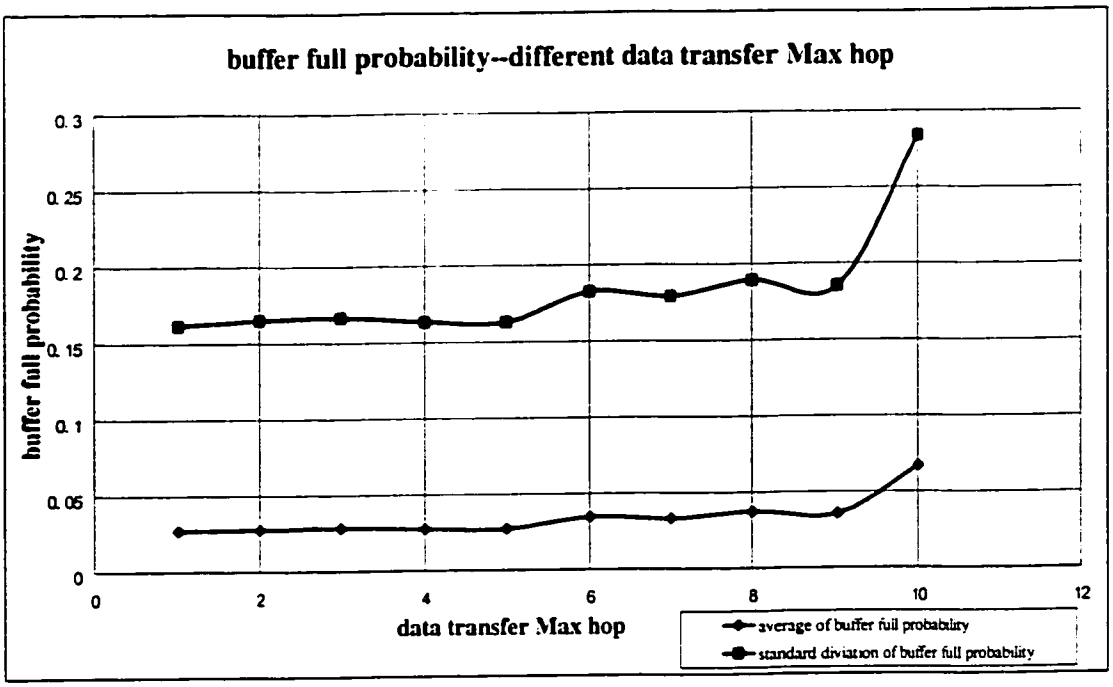


Figure 4-25 buffer full probability—different data transfer Max hop

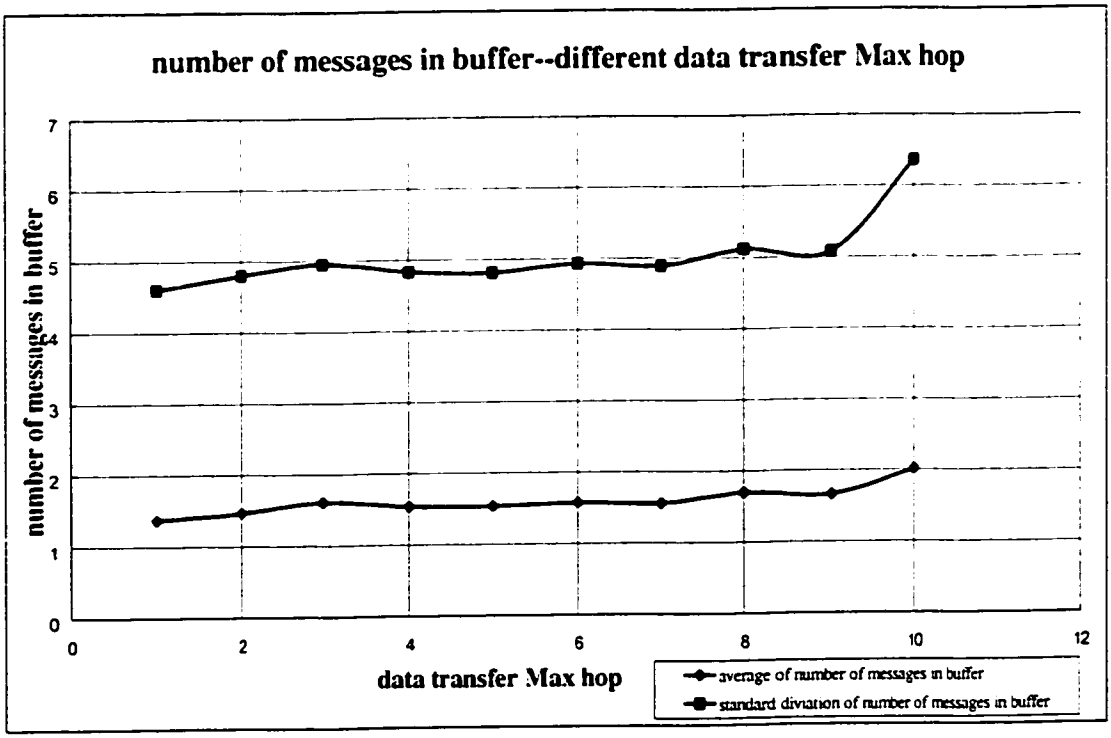


Figure 4-26 messages in buffer—different data transfer Max hop

Test conditions:
Total node No. 50;moving nodes 20;
Max simulation iteration 10000;Max node speed 108km/h; data Timeout 0.6s;
Load parameter Lamda=0.125; send hello Period 2s; send TC Period 6s.

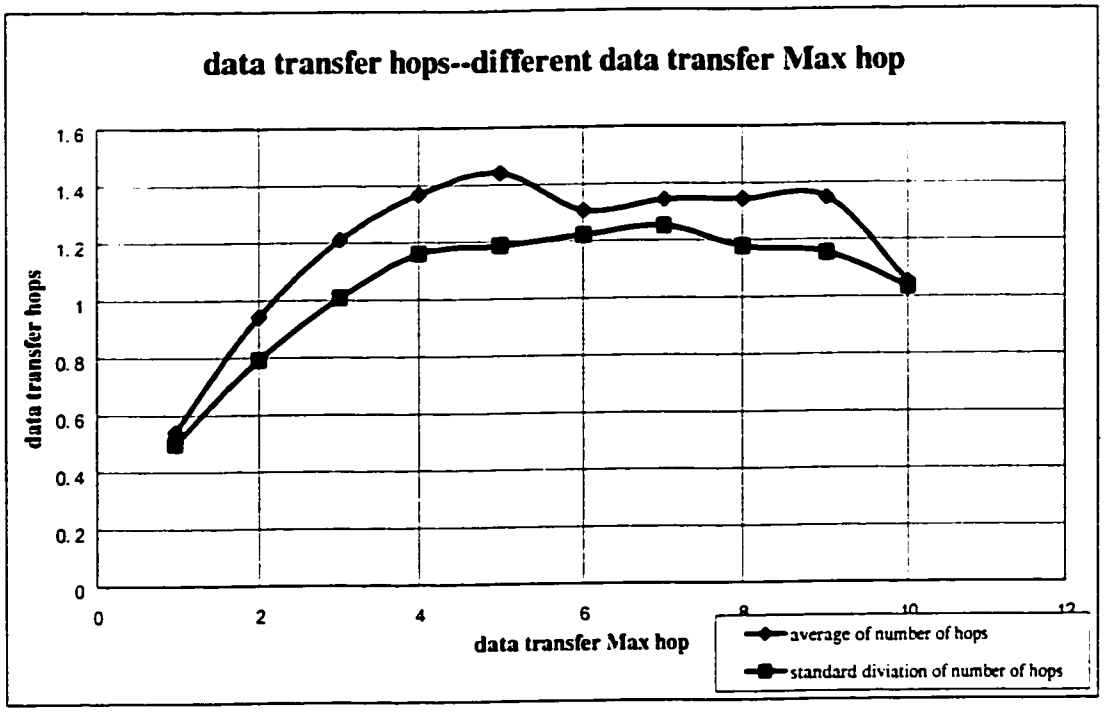


Figure 4-27 data transfer hops—different data transfer Max hop

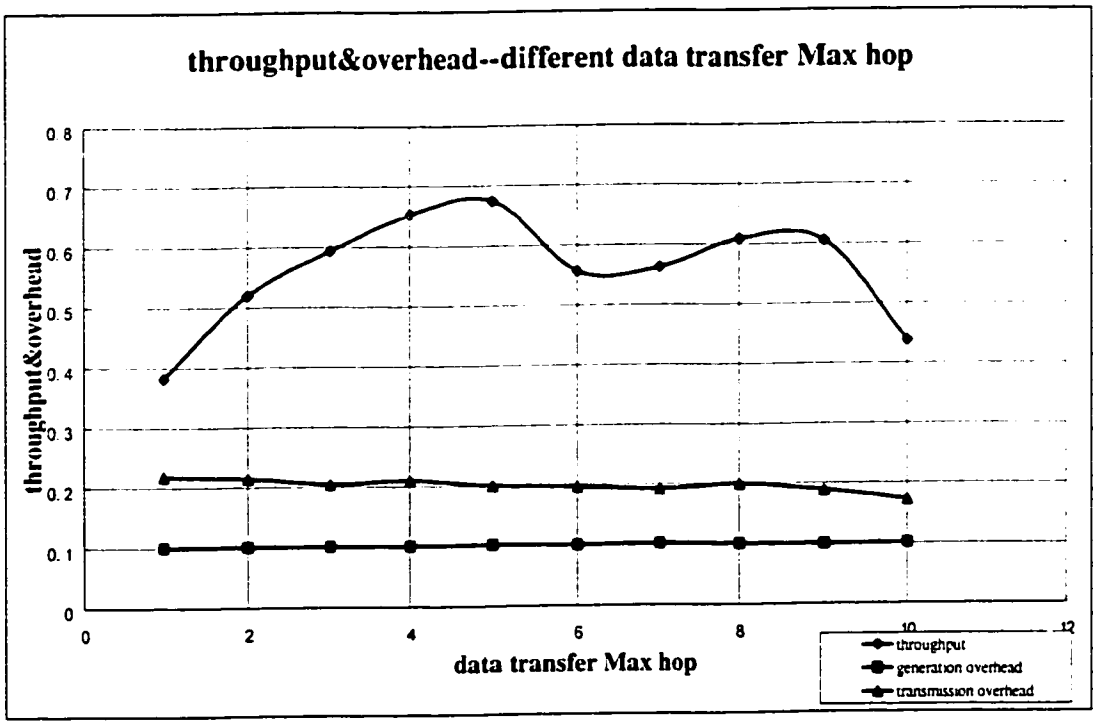


Figure 4-28 throughput and overhead-- different data transfer Max hop

Test conditions:
Total node No. 50;moving nodes 20;
Max simulation iteration 10000;Max node speed 108km/h; data Timeout 0.6s;
Load parameter Lamda=0.125; send hello Period 2s; send TC Period 6s.

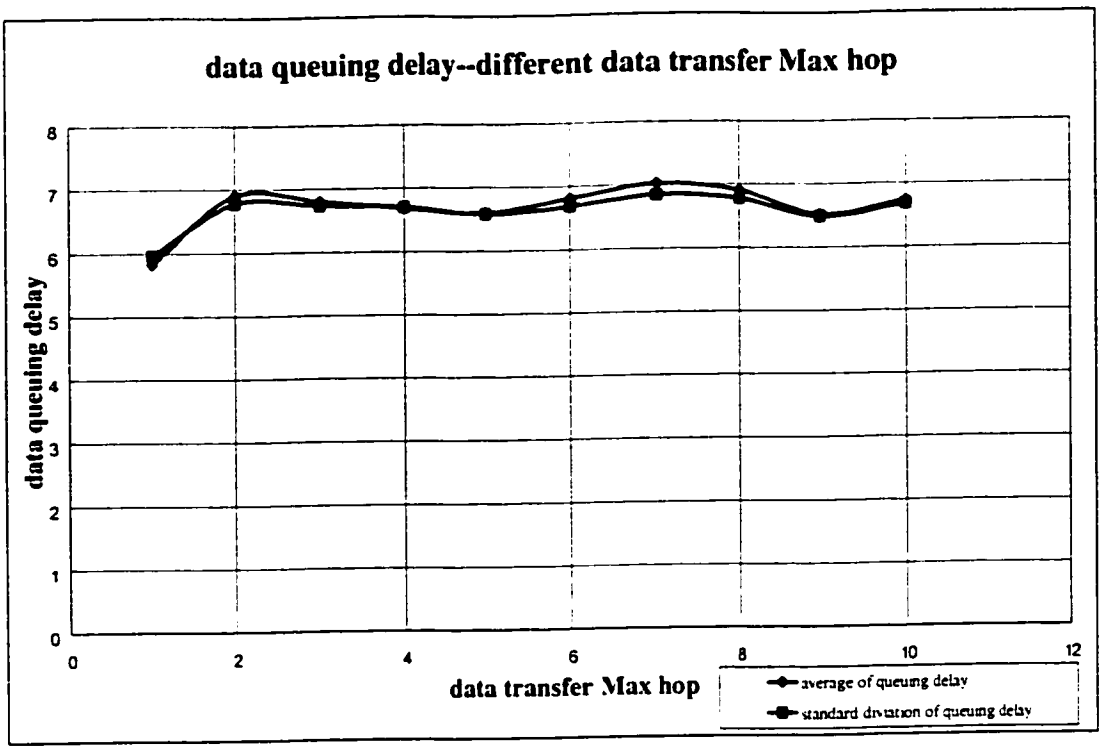


Figure 4-29 data queuing delay-- different data transfer Max hop

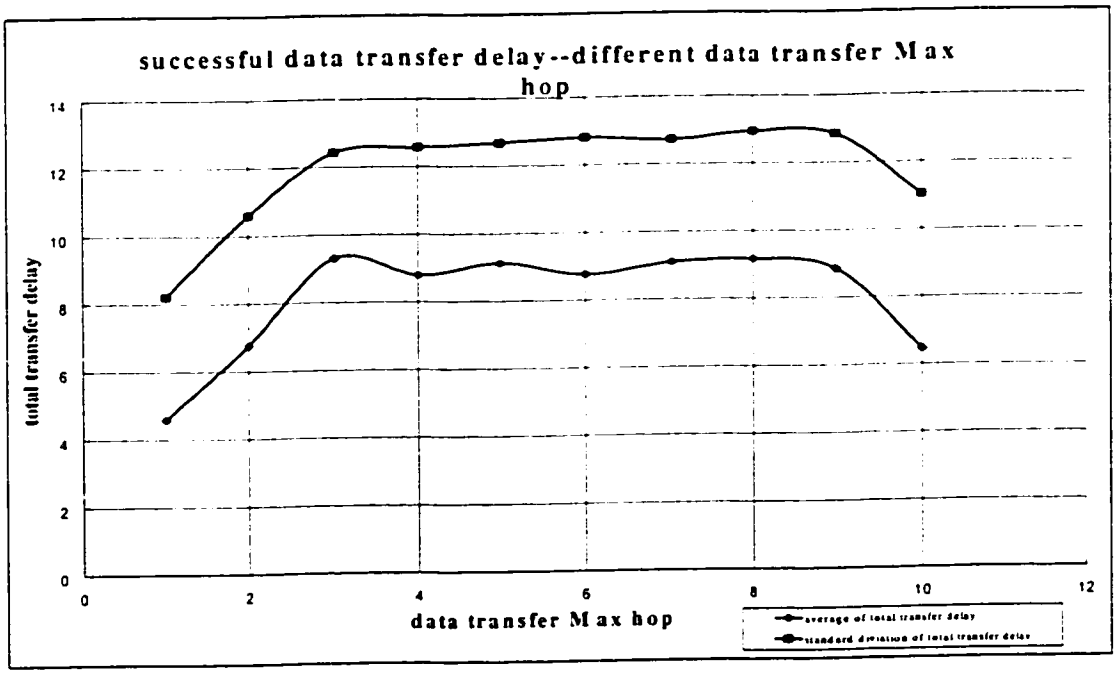


Figure 4-30 successful data transfer delay-- different data transfer Max hop

Test conditions:
Total node No. 50;moving nodes 20;
Max simulation iteration 10000;Max node speed 108km/h; data Timeout 0.6s;
Load parameter Lamda=0.125; send hello Period 2s; send TC Period 6s.

4.6 Network performance under different node move probability

In Fig. 4-31, more node connections will be lost due to more node mobility. This causes more data queuing in the buffers to find a route. So the buffer full probability and average number of messages in buffers will increase with more nodes moving. More connection loss makes messages travel more hops. So data transfer hops increase as shown in Fig. 4-33. Throughput decreases because of congestion and loss, and generation overhead is almost a constant, non-related to the number of moving nodes. Transmission overhead varies slightly, because no extra control messages need to be sent in response to link failures and additions. As are shown in Fig. 4-34.

In Figs. 4-35 and 4-36, the data queuing delay and total transfer delay increase with more nodes moving. This is because with more nodes moving, more connections are lost and more messages are queued in the buffers.

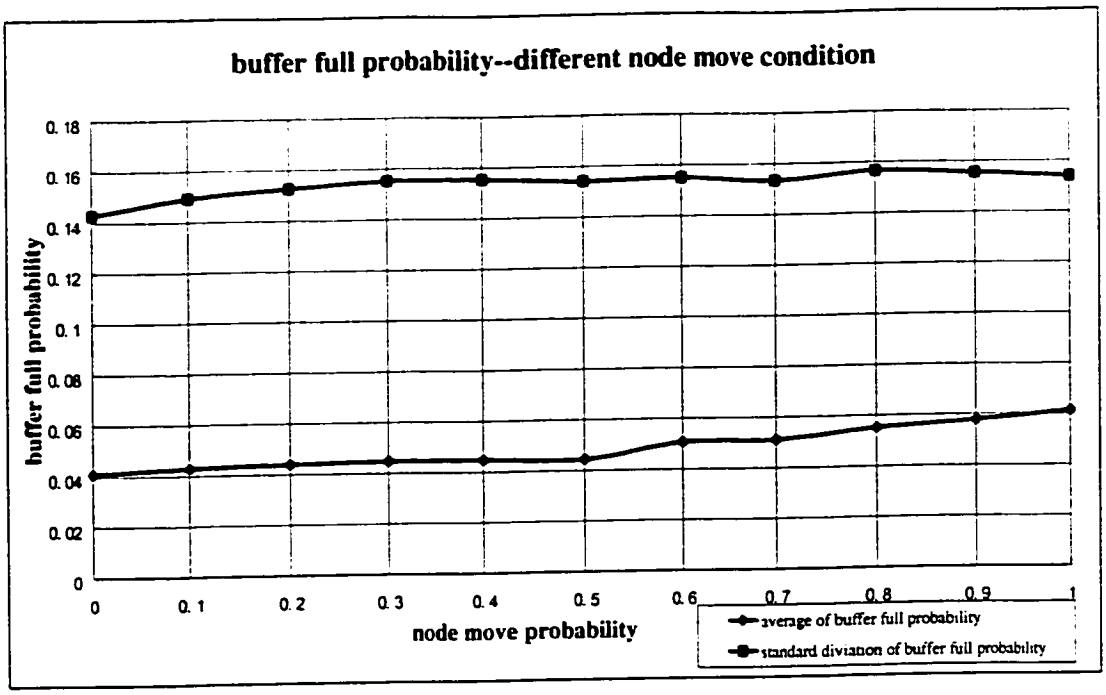


Figure 4-31 buffer full probability—different node move condition

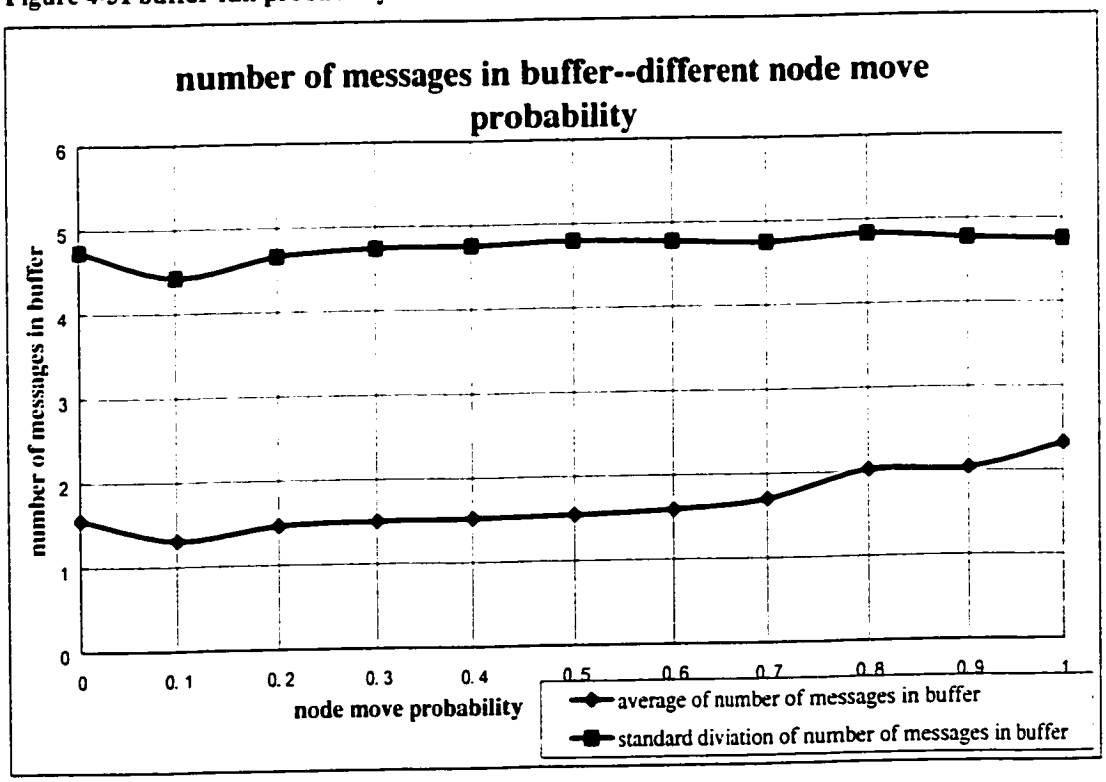


Figure 4-32 number of messages in buffer—different node move condition

Test conditions:
Total node No. 50;
Max simulation iteration 10000;Max node speed 108km/h; Max data transfer Hops=10; data Timeout 0.6s; load parameter Lamda=0.125; send hello Period 2s; send TC Period 6s.

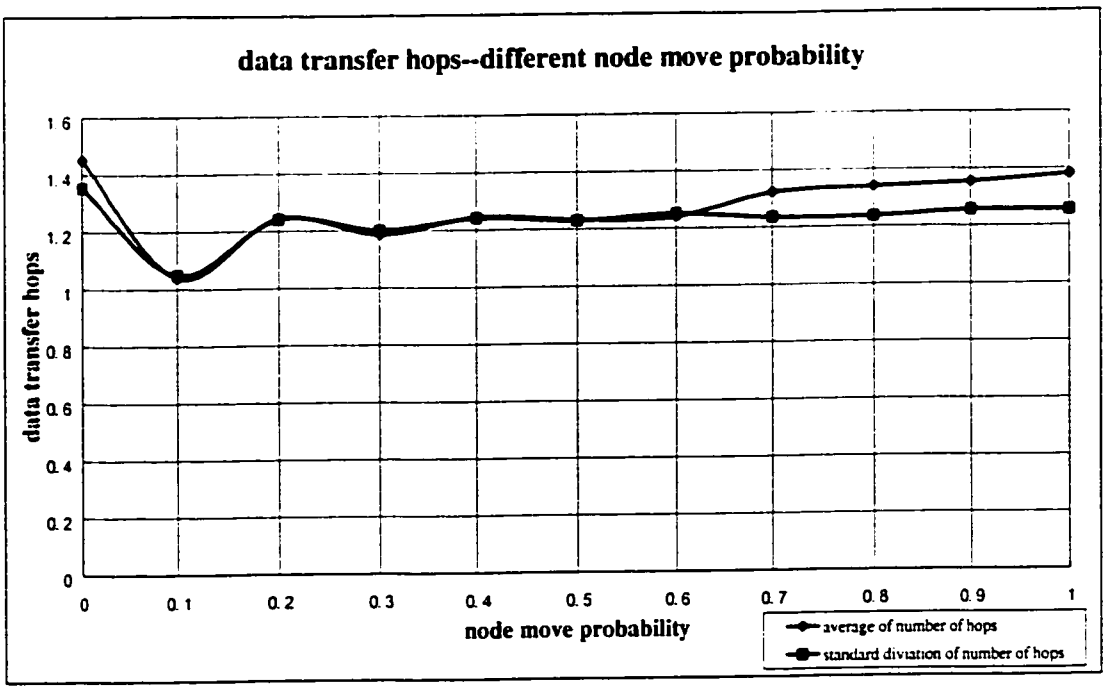


Figure 4-33 data transfer hops—different node move condition

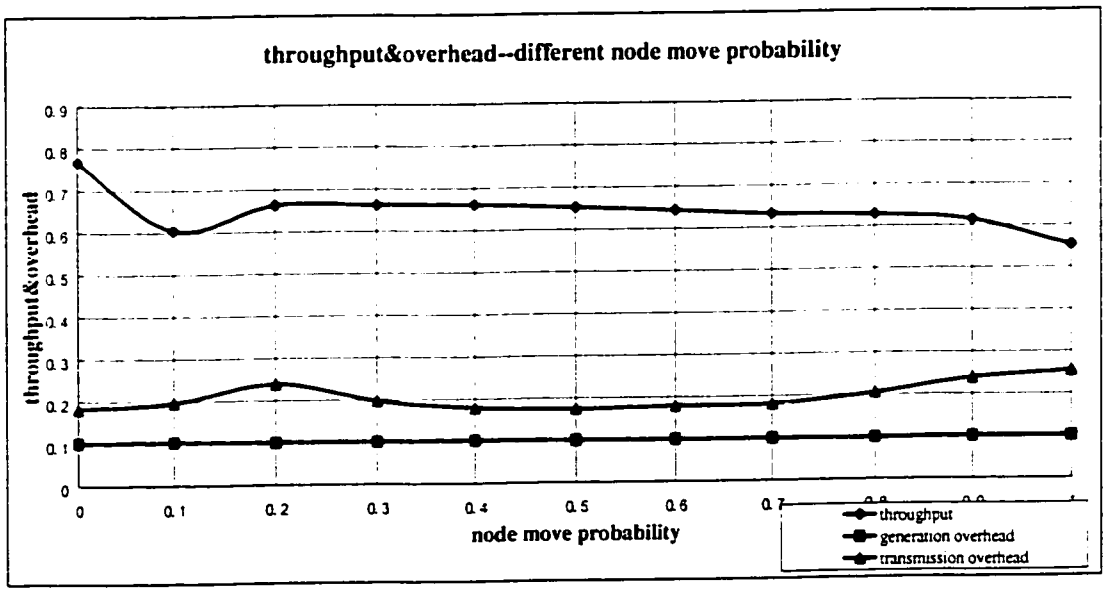


Figure 4-34 throughput and overhead—different node move condition

Test conditions:
Total node No. 50;
Max simulation iteration 10000;Max node speed 108km/h; Max data transfer Hops=10; data Timeout 0.6s; load parameter Lamda=0.125; send hello Period 2s; send TC Period 6s.

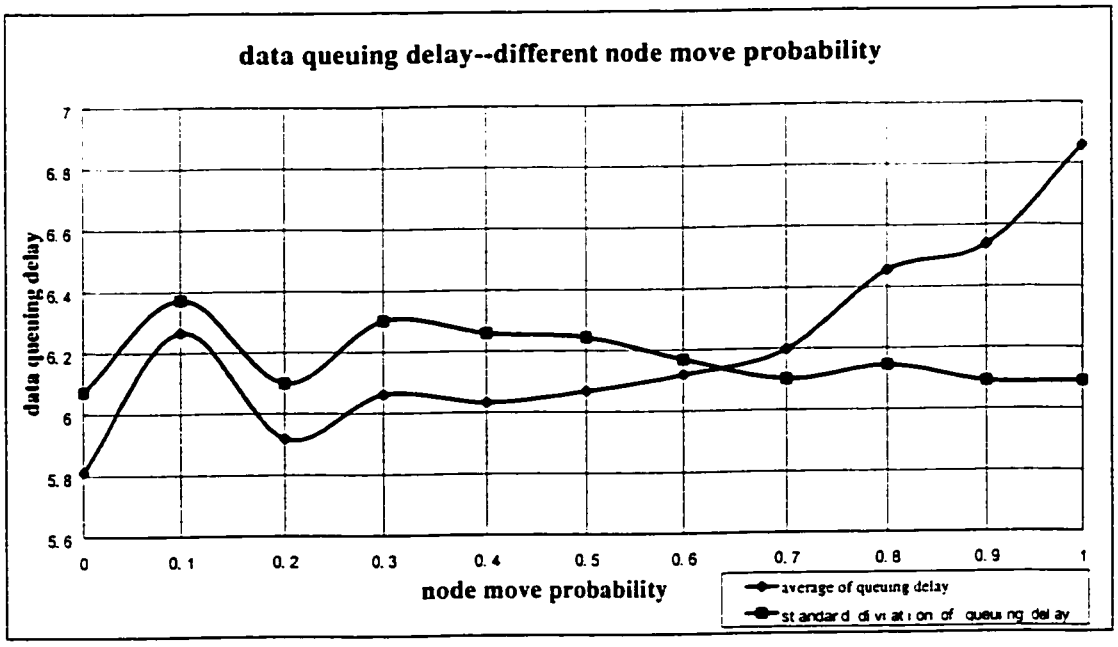


Figure 4-35 data queuing delay—different node move condition

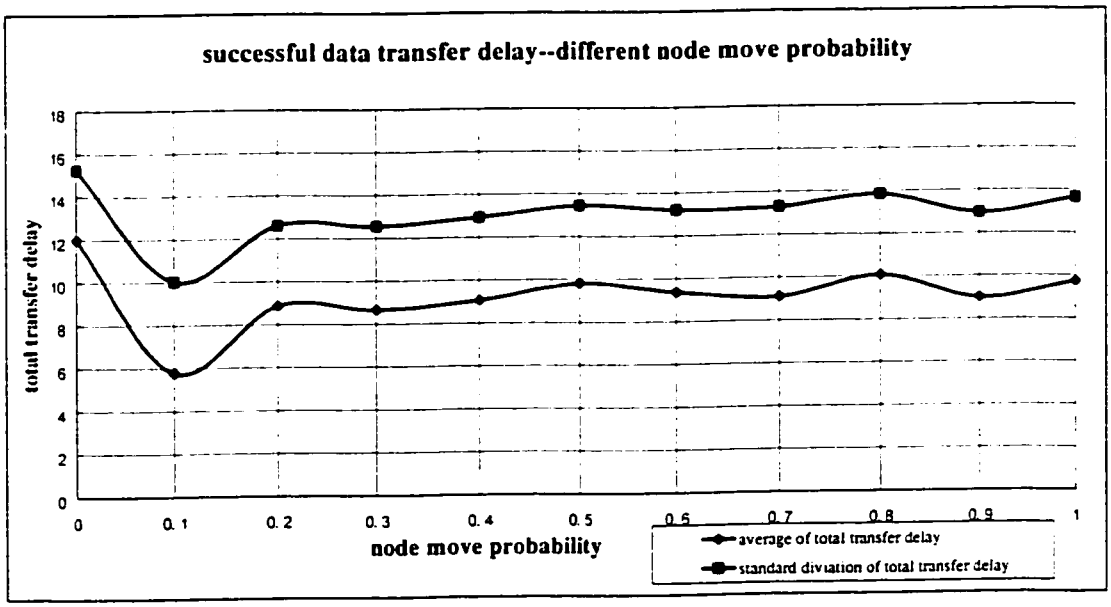


Figure 4-36 successful data transfer delay—different node move condition

Test conditions:
Total node No. 50;
Max simulation iteration 10000;Max node speed 108km/h; Max data transfer Hops=10; data Timeout 0.6s; load parameter Lamda=0.125; send hello Period 2s; send TC Period 6s.

4.7 Performance comparison of OLSR with other routing protocols

In this thesis, a detailed performance analysis of OLSR is presented. The performance comparison is not our main topic. Since in the application view, it's hard to prefer one protocol to others even if the most performance criteria of this protocol are better than the others because different protocols have different applications. For example, if we take comparison of the OLSR and AODV, one is proactive and the other is reactive. They have their own advantages and suitable applications. Which one is to be used depends on the situation. No one can replace the other. The other reason is that the testing conditions vary much from different research results and no normalization has been adopted by all researchers in this area.

Although comparison is not our main topic, we still give an example comparison of the performance of OLSR with other routing techniques to get an overall idea of how is the performance of OLSR.

Figs. 2-6 to 2-10 in section 2.14 show Iwata's simulation results of 5 different routing techniques. In this section, Iwata's results will be used to compare with the performance of OLSR.

Due to the difference of simulation model, simulation conditions of each model should be elaborated before any fair comparisons could be made. Also some parameters conversions are necessary. Even though, the comparison is approximate.

The testing conditions of Iwata's and ours have been explained clearly in chapter 2 and 3. We only mention the main differences that could be important factors affecting the

performance results. The first criteria is the number of nodes. There are 200 nodes in Iwata's simulation and 50 nodes in our simulation. We decrease the performance of OLSR by 4 times. The other one is the packet size. In OLSR, all kinds of packets have the same size of length of 20000 bits and the iteration time is 0.02s. In Iwata's model, packets have different size, 10K bits for data packets, 2K bits and 500 bits for different control packets. Smaller control packet size leads to shorter iteration time, thus benefits of shorter delay time and higher network efficiency. The drawback of different packet size is longer processing time and longer latency. The third, in OLSR, the transmission rate of channel is assumed to be 1Mbps while in Iwata's model it is 2Mbps, this may lead to at least 50% performance increasing. The fourth, in OLSR, the buffer size of each node is 30 and in Iwata's model is 15. This may lead to 50% performance degradation.

Two example comparisons will be shown. The first one is the comparison of total transfer delay by Figs. 4-6 and 2-8. In Fig 4-6, when sending control message period is set to 100 (which is the normal value), the total transfer delay is 5 iterations (5* 0.02=0.1 s). If considering the number of nodes and other criteria, it can be 400 ms. Compared to other techniques in Fig. 2-8, it's in the middle and superior to on-demand routing techniques.

The second comparison is the number of average data transfer hops by Figs. 4-3 and 2-9. In Fig 4-3, when sending control message period is set to 100 (which is the normal value), the number of average hops is 1.08. If considering the number of nodes and other criteria, it can be 4.32. Compared to other techniques in Fig. 2-9, it's almost the same as FSR and superior to other routing techniques.

From the above comparison, a conclusion can be made that OLSR has a very good overall performance among the five different techniques.

5 Conclusions and future work

In this thesis, a detailed performance analysis of OLSR is presented. Since most papers emphasize the performance comparison of different protocols. But in mobile ad- hoc networks, it's hard to prefer one protocol to others even if the most performance criteria of this protocol are better than the others because different protocols have different applications. For example, if we take comparison of the OLSR and AODV, one is proactive and the other is reactive. They have their own advantages and suitable applications. Which one is to be used depends on the situation. No one can replace the other. So we concentrate on the in depth performance analysis of OLSR.

Some interesting conclusions are summarized below:

1. Through sufficient simulation, we find that some nodes are bottlenecks of the performance of the whole network. They need to be better equipped to improve the overall performance. These nodes are usually 4 or 5 percentage of the total number of nodes.
2. In order to get better performance, the sending Hello period is better set at 2 seconds and corresponding sending TC period is set to 6 seconds.
3. Neighbour table entry hold time is better set at 300*0.02 = 6 seconds and corresponding topology table entry hold time is set to 18 seconds.
4. Max data transfer hop is better set at a medium value. It's 5~ 9 for a 50-node network.

For future work, we suggest to analyze the detailed performance of other protocols existing in MANET using the same simulation assumption and analysis method as ours.

The first important one to choose is AODV, which is the representative of the reactive type and mostly in use.

The other suggestion is analyzing the performance more efficiently by adding some other performance parameters such as processing time. In order to calculate the processing time, we suggest to establish a small mobile ad-hoc network and use real–time implementation techniques.

Reference

[1] B. Crow, I. Widjaja, J. Kim and P.Sakai, "IEEE 802.11 wireless local area networks," IEEE communications magazine, Sept. 1997.

[2] C.E.Perkins and P.Bhagwat, " Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers", in Proc. ACM SIGCOMM, vol. 24, No. 4, pp.234-244, Oct. 1994.

[3] Charle E. Perkins, Elizabeth M. Royer, Samir R.Das, "Ad hoc on-demand distance vector routing", Proc. IEEE WMCSA'99, vol.3, New Orleans, LA, pp. 90-100.

[4] Charle E. Perkins, Elizabeth M. Royer, Samir R.Das, "IP Flooding in Ad Hoc Mobile Networks", Internet-Draft, draft-ietf-manet-BCAST-00.txt, Nov. 2001.

[5] Chiang CC., Wu H-K, Liu W., and Gerla M. 1997 " Routing in clustered multi-hop, mobile, wireless networks", Proc. IEEE Singapore Int. Conf. Networks, pp.197-211.

[6] D. Bersekas and R.Gallager, "Data Networks", 2nd ed. Englewood, NJ: Prentice-hall, 1992.

[7] Edward C. Prem, "Wireless local area networks", ftp://ftp.netlav.ohio-state.edu/pub/jain/wirelesslans/index.htm

[8] ETSI STC-RES10 Committee. Radio Equipment and Systems: High Performance Radio Local Area Network Type 1, Functional specifications, June 1996.

[9] ETSI STC-RES10 Committee. Radio Equipment and Systems: High Performance Radio Local Area Network Type 2, Data Link Control Layer 2000.

[10] H.-Y. Lach. EY-NPMA. Technical Report 94/CAM/1, ETSI RES 10 working group paper, Sept, 1994.

[11] IEEE 802.11, Working Group for Wireless Local Area Networks. "Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications", IEEE Approved Draft Standard, p802.11, D6.1, May 1997.

[12] Iwata A. Chiang C.C., Pei G., Gerla M., Chan T.W., 1999 "Scalable routing strategies for Ad-Hoc wireless Networks", IEEE journal on selected areas in Comm., Vol. 17, No. 8, pp.158-162.

[13] Jim Geier, " Wireless LANs", Macmillan Network Architecture & Development Series,1999.

[14] Johnson,D.B., Maltz, D.A., " Dynamic Source Routing in Ad-Hoc wireless Networks," in Mobile Computing, chapter 5, pp.153-181, Kluwer, 1996.

[15] Joseph Macker and Scott Corson. Mobile Adhoc Networking and the IETF. ACM Mobile Computing and Communications Review, 1998.

[16] Kleinrock L. and Stevens K. 1971 " Fisheye: Lenslike Computer Display Transformation", Computer Sci. Dept, University of California, Los Angeles, CA, Tech. Rep.

[17] LAN MAN Standards Committee of the IEEE Computer Society. Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications, High-speed Physical Layer in the 5 GHz band, Sept, 1999. IEEE 802.11a.

[18] LAN MAN Standards Committee of the IEEE Computer Society. Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications, High-speed Physical Layer in the 2.4 GHz band, Sept, 1999. IEEE 802.11b.

[19] LAN MAN Standards Committee of the IEEE Computer Society. Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications. June 1997. IEEE 802.11.

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

[21] Leonard Kleinrock and Fouad A. Tobagi. Packet Switching in Radio Channels: Part I – Carrier Sense Multiple-Access Modes and Their Throughput-Delay Characteristics. IEEE transactions on Communications, volume COM-23, Dec 1975.

[22] L. Kleinrock and Scholl. Packet switching in radio channels: New Conflit Free Multiple Access Schemes. IEEE transactions on Communications, page 1015-1029, 1975.

[23] Park, V.D., Corson, " A Highly Adaptive Distributed Routing Algorithm for Mobile Networks," IEEE INFOCOM'97, Kobe, Japan, 1997.

[24] Richard G. Ogier, Fred L. Templin, "Topology Broadcast Based on Reverse-Path Forwarding (TBRPF)" Internet-Draft,draft-ietf-manet-tbrpf-05.txt,March 2002.

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

[26] Thomas Clausen, Philippe Jacquet and Anis Laouiti, "Optimized Link State Routing Protocol", Internet-Draft, draft-ietf-manet-zone-olsr-06.txt, Sept 2001.

[27] V. park, S. corson, "Temporally-Ordered Routing Algorithm version 1", Internet Draft, draft-ietf-manet-tora-spec-03.txt, June 2001.

[28] Zygmunt J.Haas and Marc R. Pearlman,"The Zone Routing Protocol for Ad Hoc Networks", Internet-Draft, draft-ietf-manet-zone-zrp-04.txt, July 2002.