

University of Alberta

EFFECTIVE ROUTING IN WIRELESS MOBILE AD HOC NETWORKS

by

Kui Wu ©

A thesis submitted to the Faculty of Graduate Studies and Research in partial fulfillment of the requirements for the degree of **Doctor of Philosophy**.

Department of Computing Science

Edmonton, Alberta  
Fall 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.

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 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.

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.

0-612-81285-5

**Canada**

University of Alberta

Library Release Form

Name of Author: Kui Wu

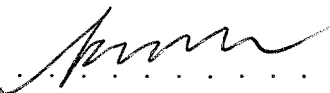
Title of Thesis: Effective Routing in Wireless Mobile Ad Hoc Networks

Degree: Doctor of Philosophy

Year this Degree Granted: 2002

Permission is hereby granted to the University of Alberta Library to reproduce single copies of this thesis and to lend or sell such copies for private, scholarly or scientific research purposes only.

The author reserves all other publication and other rights in association with the copyright in the thesis, and except as hereinbefore provided, neither the thesis nor any substantial portion thereof may be printed or otherwise reproduced in any material form whatever without the author's prior written permission.


  
.....  
Kui Wu  
329K Michener Park  
Edmonton, AB  
Canada, T6H 4M5

Date: *Aug 15, 2002*  
.....

University of Alberta

Faculty of Graduate Studies and Research

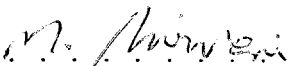
The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies and Research for acceptance, a thesis entitled **Effective Routing in Wireless Mobile Ad Hoc Networks** submitted by Kui Wu in partial fulfillment of the requirements for the degree of **Doctor of Philosophy**.

...  .....

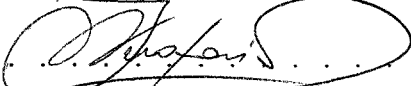
Dr. Janelle Harms

...  .....

Dr. Rick Bunt

...  .....

Dr. Mazi Shirvani

...  .....

Dr. Ioanis Nikolaïdis

...  .....

Dr. Ehab S. Elmallah

Date: . Aug. 14, 2002.

To my parents,  
who are in heaven

# Abstract

Mobile Ad hoc NETWORKS (MANETs) have received much attention and are gaining popularity because of their flexible application environments and ubiquitous information access capabilities. However, the deployment of MANETs presents many challenges. As the essential step, finding paths for multi-hop message forwarding, or routing, becomes the crucial and fundamental service support for MANETs. Routing, nevertheless, is not a trivial task in this architecture due to frequent topology changes, scarce bandwidth and power resources, and the unreliability of radio transmission. This dissertation focuses mainly on how to handle these difficulties effectively.

Location tracking is required by some applications and also provides facilities for routing implementation. We propose a LOCATION Trace Aided Routing (LOTAR) protocol that utilises the mobile users' location information to improve routing performance. In summary, we use location information to reduce the control overhead in the route discovery phase, to search quickly for a feasible path in the case of link breakage, and to hand off a flow to a stable path if the active one breaks based on prediction. Keeping "always-on" end-to-end connectivity once a flow is established is the distinctive advantage of LOTAR.

It is well-known that multipath routing can increase end-to-end throughput and provide load balancing in wired networks. However, its advantage is not obvious in MANETs because the traffic flows along the multiple paths may interfere with each other. In addition, without accurate knowledge of topology, finding multiple disjoint paths is difficult. We present the challenges of deploying multipath methods in mobile ad hoc networks and propose two on-demand methods to search for multiple node-

disjoint paths effectively. Simulation studies present the advantages as well as some important lessons for utilising multipath routing in MANETs.

Since mobile hosts are usually battery-powered, distributing the routing tasks fairly has obvious advantages. We propose a load-sensitive on-demand routing approach that utilises the network load information as the main route selection criterion. Compared with Dynamic Source Routing (DSR), our protocol provides better performance in terms of packet delivery ratio and average end-to-end delay. At low mobility, these benefits are gained without an increase in the control overhead.

To deploy a MANET for a particular realistic system, the information about end-users' behaviours, or profiles, can be used to simplify the implementation and enhance network performance. We point out a possible direction for the commercial evolution of MANETs and present a scheme to simplify routing strategies in MANETs with the help of end-users' mobility profiles. We also study how the routing strategy improves network performance in a realistic city transportation system.

# Acknowledgements

My great debt is to Dr. Janelle Harms, who has been a resourceful advisor and one of the kindest people I have ever met. Without her timely help and constructive advice, this dissertation should have never been finished.

My deepest thanks also to Dr. Ehab S. Elmallah and Dr. Ioannis Nikolaidis, who have given help and encouragement during my staying at the University of Alberta. I would like to thank Dr. Rick Bunt and Dr. Mazi Shirvani for their willingness to be in my committee.

I am grateful to the students in the communication networks research group at the University of Alberta. These wonderful people are always there to help me in a myriad of ways. It is impossible to express my gratitude in words.

Finally, millions of thanks to my sisters, Jialing Wu and Shulan Wu, for backing me up during the hard times, to my wife, Ju Yang, for her amazing patience and continuing support, and to my son, Theodore, for the joy he has brought to my life.



# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Background and Motivation . . . . .	1
1.2	Contributions . . . . .	4
1.3	Organisation of the Dissertation . . . . .	6
<b>2</b>	<b>Prior Work on Routing Protocols for MANETs</b>	<b>7</b>
2.1	Categories of Existing Routing Protocols for MANETs . . . . .	7
2.2	Proactive Routing Protocols . . . . .	8
2.2.1	Destination-Sequenced Distance-Vector Routing (DSDV) . . . . .	9
2.2.2	Global State Routing (GSR) . . . . .	10
2.3	Reactive Routing Protocols . . . . .	11
2.3.1	Dynamic Source Routing (DSR) . . . . .	11
2.3.2	Ad hoc On-Demand Distance Vector (AODV) Routing . . . . .	12
2.3.3	Location-Aided Routing (LAR) . . . . .	13
2.4	Hybrid Routing Protocols . . . . .	15
2.5	Proactive vs. Reactive vs. Hybrid Routing . . . . .	16
<b>3</b>	<b>Evaluation Methodology</b>	<b>17</b>
3.1	Simulation Model . . . . .	17
3.2	Methodology . . . . .	20
3.3	Performance Metrics . . . . .	21
3.4	Validation . . . . .	21
3.5	Verification and Calibration . . . . .	22
<b>4</b>	<b>Location Trace Aided Routing</b>	<b>24</b>
4.1	Related Work . . . . .	24
4.2	LOcation Trace Aided Routing Protocol (LOTAR) . . . . .	27
4.2.1	Introduction . . . . .	27
4.2.2	Basic Data Structures . . . . .	28
4.2.3	Route Discovery . . . . .	29
4.2.4	Route Reconstruction . . . . .	31
4.2.5	Local Flow Handoff . . . . .	33
4.2.6	Global Flow Handoff . . . . .	36
4.2.7	Correctness of LOTAR . . . . .	36
4.3	Link Lifetime Prediction Methods . . . . .	37
4.3.1	Link Lifetime Prediction in the Random Drunken Mobility Model . . . . .	38
4.3.2	Link Lifetime Prediction in the Random Waypoint Mobility Model . . . . .	38
4.4	The Criteria for Setting Parameters . . . . .	39
4.5	Simulation Settings . . . . .	40
4.6	Simulation Results . . . . .	41
4.6.1	Mobility Features . . . . .	41

4.6.2	Performance in Different Mobility Models . . . . .	42
4.6.3	The Influence of the Location Update Mechanism . . . . .	48
4.6.4	Evaluation of Flow Handoff- Ideal Performance . . . . .	58
4.7	Conclusions . . . . .	61
<b>5</b>	<b>On-Demand Multipath Routing</b>	<b>63</b>
5.1	Introduction . . . . .	63
5.2	Related Work . . . . .	65
5.3	Multiple Path Selection Criteria . . . . .	66
5.4	On-Demand Multipath Calculation . . . . .	70
5.4.1	Multipath Discovery in DSR . . . . .	70
5.4.2	The Diversity Injection Method . . . . .	71
5.4.3	The Selective Broadcast Method . . . . .	71
5.4.4	The Heuristic Redirection Method . . . . .	73
5.4.5	The Ability to Find Multiple Node-Disjoint Paths . . . . .	76
5.5	Multipath Routing . . . . .	78
5.6	Simulation Model . . . . .	78
5.7	Simulation Results . . . . .	79
5.7.1	Performance Metrics . . . . .	79
5.7.2	Comparison of Multipath Routing Performance with Unipath Routing Performance . . . . .	80
5.7.3	Comparison of Multipath Routing Performance With Various Correlation Factors . . . . .	87
5.8	Conclusions . . . . .	88
<b>6</b>	<b>Load-Sensitive Routing</b>	<b>90</b>
6.1	Introduction . . . . .	90
6.2	Load-Sensitive Routing (LSR) . . . . .	91
6.2.1	Background . . . . .	91
6.2.2	Overview of LSR . . . . .	94
6.3	Simulation Model . . . . .	97
6.4	The Influence of Path Comparison Functions . . . . .	98
6.5	Simulation Results . . . . .	102
6.5.1	Packet Delivery Ratios . . . . .	102
6.5.2	Average End-to-End Delay . . . . .	105
6.5.3	Control Overhead . . . . .	105
6.6	Conclusions . . . . .	109
<b>7</b>	<b>Profile-Based Routing</b>	<b>110</b>
7.1	Introduction . . . . .	110
7.2	Prior Work . . . . .	111
7.3	Node Architecture . . . . .	113
7.4	An Example: a City Transportation Wireless Communication System	117
7.4.1	Profile-Based Routing . . . . .	117
7.4.2	Simulation Model . . . . .	118
7.4.3	Simulation Results . . . . .	121
7.5	Conclusions . . . . .	126
<b>8</b>	<b>Conclusions and Future Research</b>	<b>127</b>
8.1	Conclusions . . . . .	127
8.2	Future Research . . . . .	128
8.2.1	Supporting Link Asymmetry . . . . .	128
8.2.2	Multicasting . . . . .	129
8.2.3	QoS Support . . . . .	129
8.2.4	Middleware . . . . .	130

8.2.5 Security . . . . .	131
<b>Bibliography</b>	<b>132</b>

# List of Figures

1.1	An example of a MANET . . . . .	3
2.1	A categorisation of routing protocols in MANETs . . . . .	8
2.2	An example of AODV . . . . .	13
2.3	The “expected zone” and “request zone” . . . . .	14
3.1	The hidden/exposed terminal problems . . . . .	19
4.1	The closest direction principle and the closest distance principle . . . . .	25
4.2	An example of the route discovery mechanism in LOTAR . . . . .	31
4.3	An example of route reconstruction . . . . .	32
4.4	An example: local flow handoff case 1 . . . . .	34
4.5	An example: local flow handoff case 2 . . . . .	36
4.6	Link change frequency in different mobility models . . . . .	42
4.7	Packet delivery ratio in the Random Drunken model . . . . .	44
4.8	Packet delivery ratio in the Random Waypoint model . . . . .	45
4.9	Average end-to-end delay in the Random Drunken model . . . . .	45
4.10	Average end-to-end delay in the Random Waypoint model . . . . .	46
4.11	Control overhead in the Random Drunken model . . . . .	47
4.12	Control overhead in the Random Waypoint model . . . . .	47
4.13	Packet delivery ratio with mobility . . . . .	50
4.14	Average end-to-end delay with mobility . . . . .	51
4.15	Control overhead with mobility . . . . .	52
4.16	Packet delivery ratio at different transmission ranges . . . . .	53
4.17	Average end-to-end delay at different transmission ranges . . . . .	54
4.18	Control overhead at different transmission ranges . . . . .	55
4.19	The impact of location update interval on packet delivery ratio . . . . .	56
4.20	The impact of location update interval on average end-to-end delay . . . . .	57
4.21	The impact of location update interval on control overhead . . . . .	57
4.22	Average end-to-end connectivity . . . . .	60
4.23	Path length ratio . . . . .	61
4.24	Average control overhead in packets . . . . .	62
5.1	Different initial topologies . . . . .	67
5.2	The impact of correlation factor on end-to-end delay in a static network . . . . .	69
5.3	The algorithm for forwarding the RREP packets . . . . .	75
5.4	The ability to find multiple node-disjoint paths . . . . .	77
5.5	The frequency of route discovery for different methods . . . . .	81
5.6	Normalized control overhead for different methods . . . . .	82
5.7	Bandwidth cost for data for different methods . . . . .	83
5.8	Average end-to-end delay for different methods . . . . .	84
5.9	The CoV of network load for different methods . . . . .	85
5.10	The CoV of energy consumption for different methods . . . . .	85
5.11	Average energy consumption for different methods . . . . .	86

5.12	End-to-end delay with different correlation factors . . . . .	87
5.13	The CoV of network loads with different correlation factors . . . . .	88
6.1	Packet delivery ratio with different parameters in $f$ . . . . .	100
6.2	Average end-to-end delay with different parameters in $f$ . . . . .	101
6.3	Control Overhead with different parameters in $f$ . . . . .	101
6.4	Packet delivery ratio (10 sources with background traffic scenario) . .	103
6.5	Packet delivery ratio (10 sources with light traffic scenario) . . . . .	104
6.6	Packet delivery ratio (15 sources with heavy traffic scenario) . . . . .	104
6.7	Average end-to-end delay (10 sources with background traffic scenario)	106
6.8	Average end-to-end delay (10 sources with light traffic scenario) . . .	106
6.9	Average end-to-end delay (15 sources with heavy traffic scenario) . .	107
6.10	Control overhead (10 sources with background traffic scenario) . . . .	108
6.11	Control overhead (10 sources with light traffic scenario) . . . . .	108
6.12	Control overhead (15 sources with heavy traffic scenario) . . . . .	109
7.1	A profile-based node architecture . . . . .	115
7.2	An example of profile-based routing . . . . .	119
7.3	Average end-to-end delay for PBR and DSR . . . . .	122
7.4	Control overhead for PBR and DSR . . . . .	123
7.5	Packet delivery ratio for PBR and DSR . . . . .	124
7.6	Packet delivery ratio with location inaccuracy . . . . .	125

# List of Tables

4.1	Location table for LOTAR . . . . .	29
4.2	Routing table for LOTAR . . . . .	29
4.3	Checking table for LSR . . . . .	30
4.4	Network characteristics in different transmission ranges . . . . .	53
6.1	Packet header for LSR . . . . .	92
6.2	Load information maintained by LSR . . . . .	92
6.3	Path comparison functions with different parameters . . . . .	99

# List of Acronyms

<b>ATM</b>	Asynchronous Transfer Mode
<b>AODV</b>	Ad hoc On-demand Distance Vector routing
<b>API</b>	Application Programming Interface
<b>CBR</b>	Constant Bit Rate
<b>CT</b>	Checking Table
<b>CTS</b>	Clear-To-Send
<b>DCF</b>	Distributed Coordination Function
<b>DBF</b>	Distributed Bellman-Ford
<b>DLR</b>	Direct Link Requirement
<b>DSDV</b>	Destination-Sequenced Distance-Vector routing
<b>DSR</b>	Dynamic Source Routing
<b>FORP</b>	Flow Oriented Routing Protocol
<b>GFH</b>	Global Flow Handoff
<b>GPS</b>	Global Positioning System
<b>GSR</b>	Global State Routing
<b>HA</b>	Handoff Acknowledgement
<b>HR</b>	Handoff Requirement
<b>ISM</b>	Industrial, Scientific, and Medical
<b>LAR</b>	Location-Aided Routing
<b>LL</b>	Local Load
<b>LOTAR</b>	LOcation Trace Aided Routing
<b>LS</b>	Link State
<b>LSR</b>	Load-Sensitive Routing
<b>LT</b>	Location Table

**MAC** Medium Access Control  
**MANET** Mobile Ad hoc NETWORK  
**PBR** Profile-Based Routing  
**PDA** Personal Digital Assistant  
**PL** Path Load  
**RA** Resigning Application  
**RREQ** Route REQUEST message  
**RREP** Route REPLY message  
**RS** Resigning Sanction  
**RT** Routing Table  
**RTS** Request-To-Send  
**TCP** Transmission Control Protocol  
**TL** Traffic Load  
**TORA** Temporally Ordered Routing Algorithm  
**TTL** Time-To-Live  
**UDP** User Datagram Protocol  
**ZBR** Zone Based Routing



# Chapter 1

## Introduction

### 1.1 Background and Motivation

Since their emergence in the 1970s, wireless networks have become increasingly popular because of their ability to provide mobile users with ubiquitous communication capabilities and information access regardless of location. Conventional wireless networks are often connected to a wired network, such as an Asynchronous Transfer Mode (ATM) network or the Internet, so that the ATM or Internet connections can be extended to mobile users. This kind of wireless network requires a fixed wireline backbone infrastructure. All mobile hosts in a communication cell can reach a base station on the wireline network in one hop. In contrast to conventional wireless networks, another type of architecture, based on radio-to-radio multi-hopping, has neither fixed base stations nor a wired backbone infrastructure. In some application environments, such as battlefield communications, disaster recovery, etc., the wired network is not available, and multi-hop wireless networks provide the only feasible means for communications and information access. This kind of network is called a Mobile Ad hoc NETWORK (MANET). It is also expected to play an important role in civilian applications such as campus recreation, conferences, and electronic classrooms, where installing base stations may be too expensive. For example, an ad hoc wireless network can be rapidly deployed to broadcast information for special events such as concerts and festivals on a campus. Instead of installing expensive base stations, a set of laptop computers with wireless transmission cards is capable of constructing such a network “on the fly.”

Compared with other types of wireless networks, a MANET has the following

prominent characteristics: First, it is formed by wireless hosts that are usually mobile. Second, it does not depend on the support of a fixed wireline infrastructure. Third, since a host has only a limited radio transmission range and is able to send messages directly only to the hosts within the range, messages from one host to another host may need multiple wireless hops if the two hosts are far away from each other. Forth, energy resources of mobile hosts are limited. Finally, hosts are required to cooperate for multi-hop message forwarding.

Several basic terms should be clarified. In this dissertation, a *host* refers to a device that has wireless communication and message processing capacity. Such a device could be a Personal Digital Assistant (PDA), a laptop computer, or a cellular phone. When a host may move, it is called a *mobile host*. We will use *host* and *mobile host* synonymously, when the context is obvious or the usage makes no difference. If a host *A* can send packets directly to host *B* in one hop, we say there is a *link from A to B*. If there is a link from host *B* to host *A* as well, we say there is a *link between A and B*, and the link between *A* and *B* is *symmetric*. Since each host in a MANET is required to forward messages for others, it is also referred to as a *network node* or simply a *node*. In this dissertation, we will not distinguish between a host and a node. When host *A* can send packets directly to host *B*, we say *B* is *A's neighbour*. For an arbitrary host *A*, if there is a host *B* in the network such that *A* has no path (directly or through multiple hops) to reach *B*, we say the *network is partitioned*. The *network topology* refers to the abstracted graph, at any point in time, whose nodes represent mobile hosts and whose edges represent links among mobile hosts.

Although MANETs can provide promising services, which are not easily deployed in other network architectures, the implementation of MANETs presents several challenges. First, mobility results in dynamic topology. Hosts in MANETs are free to move randomly and organise themselves in an arbitrary fashion. Links among the mobile hosts can change very quickly, resulting in dynamic changes in network topology, message forwarding routes, and available bandwidth. Second, MANETs are energy-constrained because mobile hosts usually rely on battery energy. The implementation of routing and other services must be energy-efficient, which implies that excessive energy costs for control information are unacceptable. Furthermore, energy constraints require the fair distribution of tasks among the hosts. Heavy loads for a particular set

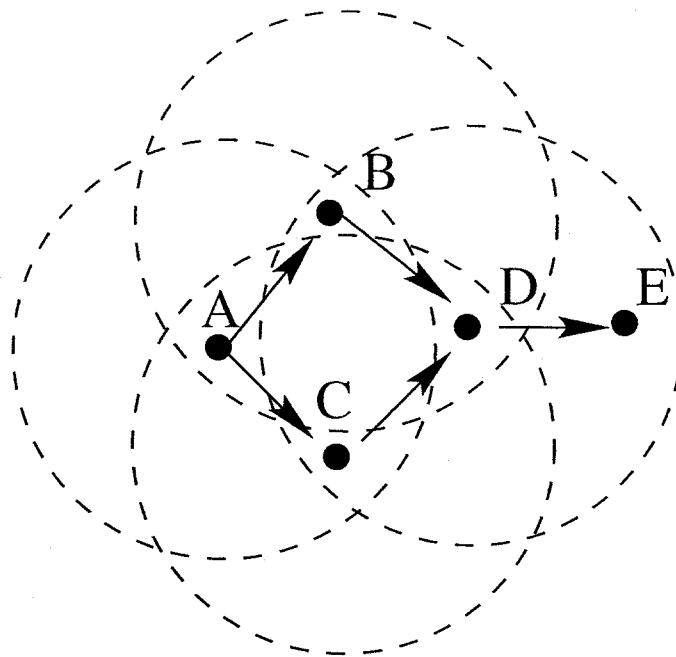


Figure 1.1: An example of a MANET

of hosts will result in rapid energy depletion of these hosts, in turn causing more serious problems such as network partitions. Third, the difference in transmission power of different mobile hosts may result in asymmetric links among the mobile hosts. The asymmetric topology increases the complexity of routing and other service support. Fourth, radio transmission is subject to the effects of multiple access, fading (radio waves from the source may arrive at the destination at slightly different times and cause signal interference), noise, interference, etc. Finally, a MANET may be large, with thousands of mobile hosts. This makes network control difficult.

These difficulties motivate us to investigate effective strategies to support services in MANETs. Finding paths between nodes, or routing, is the essential mechanism to support realistic applications involving multimedia communications and having Quality of Service (QoS) requirements. Other complex services, such as group communications and ad hoc on-line videoconferencing, need to establish message-forwarding paths as a basic step. The implementation of these services depends greatly on effective underlying routing protocols. We expect that our work in routing will provide fundamental support for these complex services in MANETs in the future.

This dissertation is intended to address the challenges in the fundamental network

functionality of routing. As shown in Fig. 1.1, if node A wants to send messages to E, it must rely on C (or B) and D to relay the messages due to the limitation of radio transmission ranges. How to search for and maintain good paths for message forwarding in an efficient manner is the main topic of this dissertation. Our goals are to develop and evaluate effective routing protocols to facilitate communications in MANETs.

## 1.2 Contributions

The main contributions of this dissertation are twofold: designing effective routing protocols and evaluating their performance in MANETs. In particular, we proposed and studied a LOcation Trace Aided Routing (LOTAR) protocol, two on-demand multipath routing methods, a Load Sensitive Routing (LSR) protocol, and a profile-based routing scheme.

A good routing protocol should possess the following properties: resilience to topology changes, fairness for routing task distribution, efficiency in energy consumption, and ease of implementation. These properties, however, are somewhat conflicting, and balances should be carefully met. For example, the complex mechanisms to make the protocol resilient to mobility might not be easy to implement; a profile-based routing strategy is simple and easy to deploy but might not be robust for all scenarios. Currently and possibly in the future, no one-size-fits-all solution can solve all routing problems in MANETs. Based on this observation, we tackle routing difficulties from different points of view and do not try to propose a powerful protocol that can efficiently handle all problems and perform well in all scenarios.

Our first achievement is to keep “always-on” end-to-end connectivity once a traffic flow is established. In our proposed LOcation Trace Aided Routing (LOTAR) protocol, we utilise the mobile users’ location information in routing and based on prediction search for alternative paths before the used one breaks. The main contributions of LOTAR include:

- efficient route searching and route maintenance mechanisms
- dynamic local and global flow handoff mechanisms

To this end we conduct:

- thorough performance investigations in different mobility prediction methods
- a simulation study of performance of flow handoff mechanisms

We have observed that our flow handoff methods can be utilised in other contexts. An example is the preemptive routing protocol [GAPK01], which hands over flows based on link lifetime prediction obtained by received signal strength.

Our second goal is to fairly distribute routing tasks among mobile hosts, since load balancing is extremely important in an energy-scarce environment. Heavily loaded hosts may quickly deplete their energy, and this in turn results in more difficult problems such as network partitions. When there is no path possible between a sender and a receiver due to a network partition, any routing effort will be in vain. We use two strategies to address load balancing problems. First, we use multiple paths between a source-destination pair. Second, we use load information as the main path selection criterion in routing and dynamically search for lightweight paths during data transmission. Our contributions in solving load-balancing problem include the following:

- two algorithms to search effectively for good quality node-disjoint paths without the help of complete topology information
- a method to obtain accurate load information
- efficient route search and dynamic route adaptation mechanisms based on a unified scheme to tune network performance

To this end we provide:

- a thorough performance study of proposed on-demand multipath routing and load-sensitive routing protocols

Most work addresses complex routing problems in general settings, that is, the proposed solutions are supposed to work without knowledge of users' mobility profiles and are expected to be suitable in all application scenarios. Since the end-users' mobility in most application environments is predictable or controllable, however,

utilising the certainties of the mobility profile can simplify routing strategies and improve network performance. If the routing protocol can obtain useful information from users' behaviours in particular environments, its implementation can be simplified and its performance can be improved further. Our contribution in this direction includes:

- pointing out a different promising direction for the implementation of MANETs by providing a feasible node architecture

To this end we:

- demonstrate the efficiency of the profile-based routing method through a concrete realistic application example

### 1.3 Organisation of the Dissertation

The rest of the dissertation is organised as follows. Chapter 2 introduces prior work on the routing problem in MANETs. Chapter 3 describes the methodology used in this thesis to evaluate the routing performance. In Chapter 4, we propose a Location Trace Aided Routing (LOTAR) protocol to utilise location information in routing and evaluate its performance. Chapter 5 and Chapter 6 use different strategies to address load balancing problems in MANETs. In Chapter 5, we propose two different ways to search for good multiple paths for routing. In Chapter 6, we use load information as the main path selection criterion in routing and dynamically search for lightweight paths during data transmission. Chapter 7 presents a scheme to utilise mobile users' profile information to simplify the routing implementation in specific application scenarios. Finally, we conclude the thesis and discuss the need for future research in Chapter 8.

## Chapter 2

# Prior Work on Routing Protocols for MANETs

Routing in MANETs is difficult since mobility causes frequent network topology changes and requires more robust and flexible mechanisms to search for and maintain routes. When the network nodes move, the established paths may break and the routing protocols must dynamically search for other feasible routes. With a changing topology, even maintaining connectivity is very difficult. In addition, keeping the routes loop free is more difficult when the hosts move. Besides handling the topology changes, routing protocols in MANETs must deal with other constraints, such as low bandwidth, limited energy consumption, and high error rates, all of which may be inherent in the wireless environment. Furthermore, the possibility of asymmetric links, caused by different power levels among mobile hosts and other factors such as terrain conditions, make routing protocols more complicated.

### 2.1 Categories of Existing Routing Protocols for MANETs

Many protocols have been proposed for MANETs. These protocols can be divided into three categories: *proactive*, *reactive*, and *hybrid*. Proactive methods maintain routes to all nodes, including nodes to which no packets are sent. Such methods react to topology changes, even if no traffic is affected by the changes. They are also called table-driven methods. Reactive methods are based on demand for data transmission. Routes between hosts are determined only when they are explicitly needed to forward

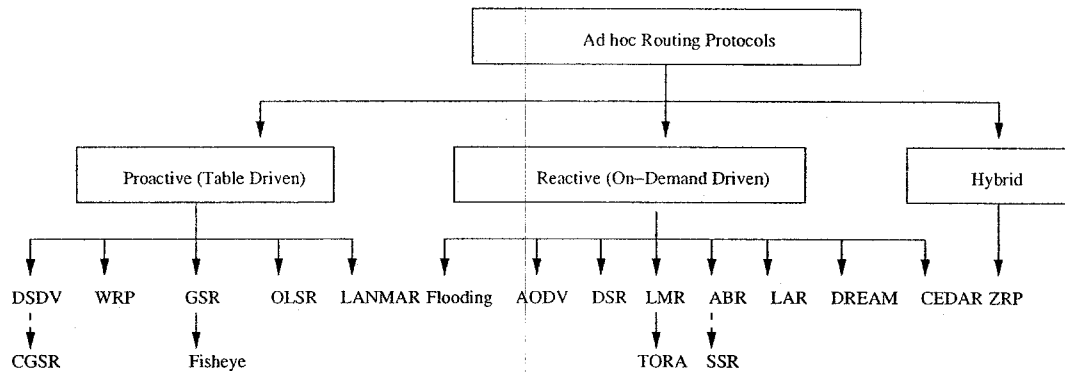


Figure 2.1: A categorisation of routing protocols in MANETs

packets. Reactive methods are also called on-demand methods. They can significantly reduce routing overhead when the traffic is lightweight and the topology changes less dramatically, since they do not need to update route information periodically and do not need to find and maintain routes on which there is no traffic. Hybrid methods combine proactive and reactive methods to find efficient routes, without much control overhead.

Fig. 2.1 is a categorisation of existing routing protocols in MANETs. In the figure, the solid lines represent direct descendants (one protocol is the revision or improvement of the other), and the dotted lines depict logical descendants (one protocol borrows the same path calculation method from the other but uses different metrics in the calculation). Since new routing protocols are always being proposed for MANETs, we do not expect to include all of them. However, brief introductions on some representative routing protocols for MANETs will be helpful to better understand the background and the later discussions in the dissertation.

## 2.2 Proactive Routing Protocols

As stated earlier, proactive routing protocols maintain routes to all destinations, regardless of whether or not these routes are needed. In order to maintain correct route information, a node must periodically send control messages. Therefore, proactive routing protocols may waste bandwidth since control messages are sent out unnecessarily when there is no data traffic. The main advantage of this category of protocols is that hosts can quickly obtain route information and quickly establish a session. This



section describes two representative proactive routing protocols, DSDV and GSR.

### 2.2.1 Destination-Sequenced Distance-Vector Routing (DSDV)

The Destination-Sequenced Distance-Vector (DSDV) [PB94] routing protocol is a table-driven algorithm based on the Distributed Bellman-Ford (DBF) [Bell57] routing mechanism. In DBF, each host maintains a table that includes the shortest distance to each destination and the next hop to get there. These tables are updated by the host exchanging information with its neighbours (local broadcast). DBF is broadly used in networks with static topologies where loop avoidance is easy. However, DBF is not suitable for a network with frequent topology changes because it lacks loop avoidance mechanisms for dynamic networks. DSDV uses the same control message exchange method (local broadcast) as DBF, but it modifies DBF to avoid loops when hosts move frequently.

DSDV uses sequence numbers assigned by the destinations to avoid loops in the routing tables. Other nodes will finally know the destination sequence number if every node periodically exchanges information with its neighbours (local broadcast). Every mobile node has a routing table which stores (1) the next hop to each destination, (2) the hop count for the path to each destination, and (3) a destination sequence number that is created by the destination itself. Each node periodically broadcasts its local routing table to its neighbours (local broadcast). When broadcasting the local routing table, each node increases and appends its own sequence number. Other nodes will attach the sequence number to the route entries created for this node. The sequence number enables the mobile nodes to distinguish stale routes from new ones.

An entry in the broadcast message contains the destination address, the number of hops to reach the destination, the sequence number assigned by the destination, and a new sequence number that is used to identify the broadcast message uniquely. The entry labelled with the latest destination sequence number is always used. If two updates have the same destination sequence number, the route with the smaller number of hops is used. For example, assume that node X receives a routing update from node Y about a route to node Z. Let  $S(X)$  be the sequence number regarding Z stored in X and  $S(Y)$  be the sequence number regarding Z forwarded by the node Y. If  $S(X) < S(Y)$ , then X sets Y as its next node to Z. If  $S(X) > S(Y)$ , then X will

ignore the routing update. If  $S(X) = S(Y)$  and the hops going through  $Y$  are smaller than the route stored in  $X$ , then  $X$  sets  $Y$  as its next node to  $Z$ . If all nodes update their local routing tables under the above constraints, routing loops can be prevented [PB94].

DSDV is a modified DBF method using destination sequence numbers to avoid routing loops. Implementing DSDV is easy since the DBF algorithm needs each node to exchange the routing information only with its neighbours. The main disadvantage is that link changes may not be quickly obtained by the farther nodes, which may in turn make non-optimal routing decisions based on the imprecise information.

### 2.2.2 Global State Routing (GSR)

Global State Routing (GSR) [CG98] is based on the Link State (LS) routing method. In the LS routing method, each node floods the link state information into the whole network (global flooding) once it realises that links change between itself and its neighbours. The link state information includes the delay to each of its neighbours. A node will know the whole topology when it obtains all link information. LS routing works well in networks with static topologies. When links change quickly, however, frequent global flooding will inevitably lead to huge control overhead.

Unlike the traditional LS method, GSR does not flood the link state packets. Instead, every node maintains the link state table based on up-to-date LS information received from neighbouring nodes, and periodically exchanges its LS information with its neighbours only (no global flooding). Before sending an LS packet, a node assigns the LS packet a unique sequence number to identify the newest LS information. LS information is disseminated as the LS packets with larger sequence numbers replace the ones with smaller sequence numbers.

The convergence time required to detect a link change in GSR is shorter than in the Distributed Bellman-Ford (DBF) [Bell57] protocol. The convergence time in GSR is  $O(D*I)$  where  $D$  is the diameter of the network and  $I$  is the link state update interval. The convergence time is normally smaller than  $O(N*I)$  in DBF, where  $N$  is the number of nodes in the networks and  $I$  is the update interval. Since the global topology is maintained in every node, preventing routing loops is easy. The drawbacks of GSR are the large size of the update messages, which consume a considerable

amount of bandwidth, and the latency of the LS information propagation, which depends on the LS information update interval time. “Fisheye” technology [Chen98] can be used to reduce the size of update messages. In this case, every node maintains highly accurate network information about the immediate neighbouring nodes, with progressively fewer details about farther nodes.

## 2.3 Reactive Routing Protocols

Reactive routing protocols can dramatically reduce routing overhead because they do not need to search for and maintain the routes on which there is no data traffic. This property is very appealing in the resource-limited environment. In this section, we introduce several reactive routing protocols to facilitate better comprehension of our proposed protocols and the performance comparison discussed in this dissertation.

### 2.3.1 Dynamic Source Routing (DSR)

The Dynamic Source Routing (DSR) protocol [MBJJ99, BJM01] uses the source routing approach (every data packet carries the whole path information in its header) to forward packets. Before a source node sends data packets, it must know the total path to the destination. Otherwise, it will initiate a route discovery phase by flooding a Route REQuest (RREQ) message. The RREQ message carries the sequence of hops it passed through in the message header. Any nodes that have received the same RREQ message will not broadcast it again. Once an RREQ message reaches the destination node, the destination node will reply with a Route REPLY (RREP) packet to the source. The RREP packet will carry the path information obtained from the RREQ packet. When the RREP packet traverses backward to the source, the source and all traversed nodes will know the route to the destination. Each node uses a route cache to record the complete route to desired destinations. Route failure is detected by the failure of message transmissions. Such a failure will initiate a route error message to the source. When the source and the intermediate nodes receive the error message, they will erase all the paths that use the broken link from their route cache.

The path calculated in DSR is loop-free since loops can be detected easily and

erased by the source routing. A few optimisations are proposed for DSR. For example, a flooded route query can be quenched early by having any non-destination node reply to the query if that node already knows a route to the desired destination; the routes can be refreshed and improved by having nodes promiscuously listen to the conversations between other neighbouring nodes.

DSR is simple and loop-free. However, it may waste bandwidth if every data packet carries the entire path information. The response time may be large since the source node must wait for a successful RREP if no routing information to the intended destination is available. In addition, if the destination is unreachable from the source node due to a network partition, the source node will continue to send RREQ messages, possibly congesting the network.

### 2.3.2 Ad hoc On-Demand Distance Vector (AODV) Routing

Since DSR includes the entire route information in the data packet header, it may waste bandwidth and degrade performance, especially when the data contents in a packet are small. Ad hoc On-Demand Distance Vector (AODV) Routing [PR99, PRD01] tries to improve performance by keeping the routing information in each node. The main difference between AODV and DSR is that DSR uses source routing while AODV uses forwarding tables at each node. In AODV, the route is calculated hop by hop. Therefore, the data packet need not include the total path.

The route discovery mechanism in AODV is very similar to that in DSR. In AODV, any node will establish a reverse path pointing toward the source when it receives an RREQ packet. When the desired destination or an intermediate node has a fresh route (based on the destination sequence number) to the destination, the destination/intermediate node responds by sending a route reply (RREP) packet back to the source node using the reverse path established when the RREQ was forwarded. When a node receives the RREP, it establishes a forward path pointing to the destination. The path from the source to the destination is established when the source receives the RREP. Fig. 2.2 shows an example of how the source node finds a path to the destination.

Dealing with path failures in AODV is more complicated than in DSR. When a node detects the link failure to its next hop, it propagates a *link failure notification*

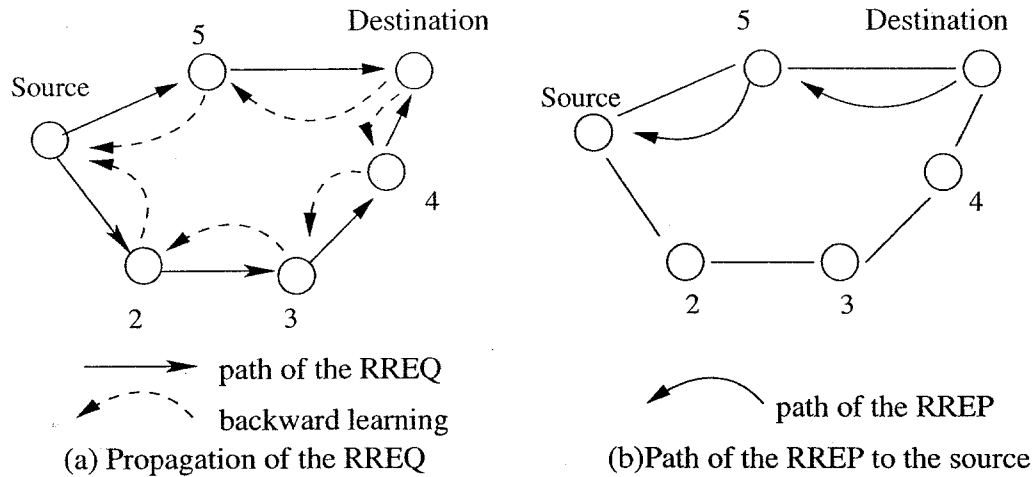


Figure 2.2: An example of AODV

message (an RREP with a very large hop count value to the destination) to each of its active upstream neighbours to inform them to erase that part of the route. These nodes in turn propagate the *link failure notification* message to their upstream neighbours, and so on, until the source node is reached [RT99]. A neighbour is considered active for a route entry if the neighbour sends a packet, which was forwarded using that entry, within the *active\_route\_timeout* interval. Note that the *link failure notification* message will also update the destination sequence number. When the source node receives the *link failure notification* message, it will re-initiate a route discovery for the destination if a route is still needed. A new destination sequence number is used to prevent routing loops formed by the entangling of stale and newly established paths.

AODV saves bandwidth and performs well in a large MANET since a data packet does not carry the whole path information. As in DSR, the response time may be large if the source node's routing table has no entry to the destination and thus must discover a path before message transmission. Furthermore, the same problems exist as in DSR when network partitions occur.

### 2.3.3 Location-Aided Routing (LAR)

The Location-Aided Routing (LAR) [KV98] protocol is an on-demand scheme. It utilises location information to limit the route query flooding area. LAR assumes that every host knows its own location and the global time, which can be provided by

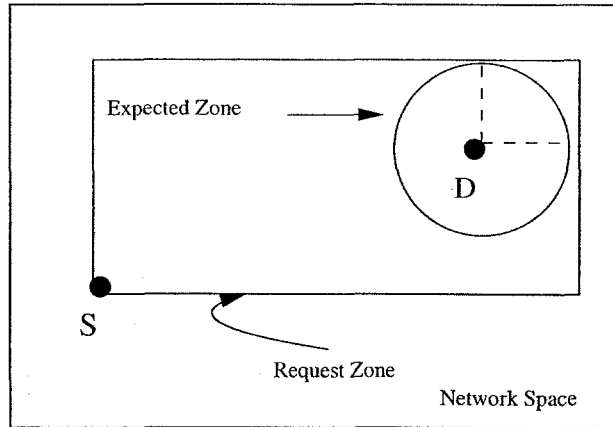


Figure 2.3: The “expected zone” and “request zone”

a Global Positioning System (GPS). The GPS is a worldwide radio-navigation system formed by a constellation of 24 satellites and their ground stations. The recent low-power implementation of GPS receivers [SSL97] makes their presence in mobile hosts a viable option. Note that in LAR, not only does every host need to know its location, but also it needs to know the locations of other hosts, which are not specifically given by GPS. Therefore, a location management mechanism is required in the network.

LAR defines the concepts of “expected zone” and “request zone.” Assume a host  $S$  wants to find a route to a host  $D$ . If it knows  $D$  was at location  $L$  at time  $t_0$  and the current time is  $t_1$ , then the “expected zone” of host  $D$  is the region that host  $S$  expects to contain node  $D$  at current time  $t_1$ . In the following example, the “expected zone” is the circular region with a centre of  $L$  and radius of  $v * (t_1 - t_0)$ , where  $v$  is the mobile speed of host  $D$ . The “request zone” is defined to limit the route query flooding. A node forwards a route request only if it belongs to the “request zone.” In Fig. 2.3, the “request zone” is the smallest rectangular region which includes the “expected zone” of  $D$  and the current location of the source  $S$ . How to define a proper “request zone” is still a topic for further study.

When node  $S$  wants to send messages to node  $D$ , it will broadcast a route query message, which is forwarded only by the nodes in the “request zone.” When a node forwards the route query, it appends its node ID to the head of the packet. After node  $D$  finally receives the route query, it sends a route reply back to the source node

S using the reverse path which is recorded in the head of the route query packet. The route from S to D is established when the source node S receives the route reply packet.

LAR can efficiently reduce the RREQ flooding cost [KV98]. The main problem with this method is that obtaining accurate location information may be difficult in some environments (for example, GPS does not work well indoors, and proximity does not guarantee connectivity).

## 2.4 Hybrid Routing Protocols

A typical hybrid routing protocol is Zone Based Routing (ZBR) [HP00]. ZBR combines the proactive and reactive routing approaches. It divides the network into routing zones. The routing zone of a node X includes all nodes within hop distance at most  $d$  from node X. All nodes at hop distance exactly  $d$  are said to be the peripheral nodes of node X's routing zone. The parameter  $d$  is the zone radius. ZBR proactively maintains the routes within the routing zones and reactively searches for routes to destinations beyond a node's routing zone. Route discovery is similar to that in DSR with the difference that route requests are propagated only via peripheral nodes. ZBR can be dynamically configured to a particular network through adjustment of the parameter  $d$ . ZBR will be a purely reactive routing protocol when  $d = 0$  and a purely proactive routing protocol when  $d$  is set to the diameter of the network.

ZBR discovers routes as follows. When a source node wants to send data to a destination, it first checks whether or not the destination is within its routing zone. If it is, then a route can be obtained directly. Otherwise, it floods a route request to its peripheral nodes. The peripheral nodes in turn execute the same algorithm to check whether the destination is within their routing zone. If it is, a route reply message is sent back to the source. Otherwise, the peripheral node floods the route request to its peripheral nodes again. This procedure is repeated until a route is found.

Flexibility is a major advantage of the ZRP protocol. For instance, the radius (in hops) of the local zones can be chosen to accommodate various types of ad hoc networks. Large routing zones are more efficient for sparse networks with many slow-moving nodes, whereas zones with smaller radii will perform better in a dense network

with few, fast-moving nodes [HP00].

## 2.5 Proactive vs. Reactive vs. Hybrid Routing

The tradeoffs between proactive and reactive routing strategies are quite complex. Which approach is better depends on many factors, such as the size of the network, the mobility, the data traffic and so on. Proactive routing protocols try to maintain routes to all possible destinations, regardless of whether or not they are needed. Routing information is constantly propagated and maintained. In contrast, reactive routing protocols initiate route discovery on the demand of data traffic. Routes are needed only to those desired destinations. This routing approach can dramatically reduce routing overhead when a network is relatively static and the active traffic is light. However, the source node has to wait until a route to the destination can be discovered, increasing the response time.

The hybrid routing approach can adjust its routing strategies according to a network's characteristics and thus provides an attractive method for routing in MANETs. However, a network's characteristics, such as the mobility pattern and the traffic pattern, can be expected to be dynamic. The related information is very difficult to obtain and maintain. This complexity makes dynamically adjusting routing strategies hard to implement.

In this dissertation, we will mainly focus on reactive routing methods because of their prominent advantages in saving bandwidth in resource-scarce environments.



# Chapter 3

## Evaluation Methodology

### 3.1 Simulation Model

Because mobile users move in an arbitrary fashion, it is difficult to build up an analysis model to investigate network performance of MANETs. It is also costly and difficult to set up a real network to include arbitrary moving and therefore arbitrary network topology changes [MBJ00]. Based on the above observations, we use simulation as the main performance evaluation method in this thesis.

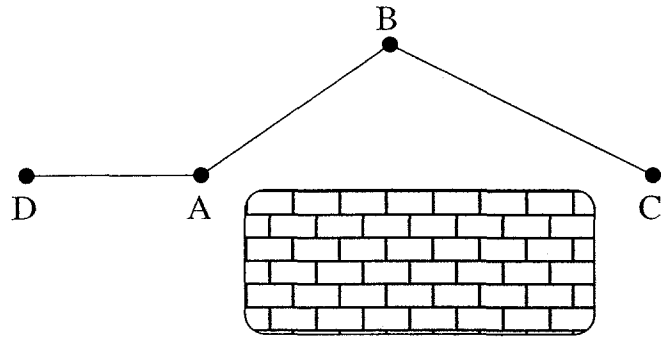
In this dissertation, all simulation studies were performed using GloMoSim [ZBG98], a scalable simulation library for wireless network systems built in the Parsec [BMT98] simulation environment. Parsec is a C-based simulation language for sequential and parallel execution of discrete-event simulation models. A Parsec program consists of a set of entities and C functions. Each entity is a logical process that models a corresponding physical process, and entities can be created and destroyed dynamically. Events are modeled by message communications among the corresponding entities. Parsec supports parallel execution by a set of partitioning algorithms. This is useful for simulating large-scale systems.

In GloMoSim, the networking stack is decomposed into several layers. A number of protocols have been developed at each layer, and models of these protocols or layers can be developed at different levels of granularity. GloMoSim aims to develop a modular simulation environment for protocol stacks. If all protocol models obey the strict Application Programming Interfaces (APIs) defined at each layer, it will be feasible to simply swap protocol models at a certain layer without having to modify the models for the remaining layers in the stack. In addition, simulating a new

protocol at a certain layer is easy because the only thing to do is to insert the new protocol model into the corresponding layer. We need not pay attention to other layers, in which we are not interested.

In the physical layer, the channel capacity of mobile hosts was set to the same value: 2 Mbps, which is a standard support for most wireless cards. A free space propagation model [Rap95] with a threshold cutoff was used as the channel model. In the free space model, the power of a signal attenuates as  $1/r^2$ , where  $r$  is the distance between mobile hosts. In the radio model, capture effects are taken into account, meaning that if the desired (currently being received) signal is 10 dB greater in strength than the interfering signal, the desired signal completely captures the receiver. The value of 10 dB is defined as a constant value in GloMoSim and represents the common setting in the free space model with capture effects. In the simulations, we assume all nodes have the same radio transmission range. Different radio transmission ranges in different mobile nodes will result in asymmetric links. Dealing with problems caused by the asymmetric links in MANETs is very challenging and will be explored in the future research.

In the Medium Access Control (MAC) layer, we used the Distributed Coordination Function (DCF) of IEEE 802.11 for wireless LAN as the MAC layer protocol. It has the functionality to notify the network layer about link failures and can solve the hidden/exposed terminal problems. As shown in Fig. 3.1, host A and host C cannot hear each other. When A is transmitting a packet to B, C cannot sense the transmission from A. Thus C may transmit a packet to B and cause a collision at B. This is the “hidden terminal” problem since A is hidden from C. Similarly, when B is transmitting a packet to C, A cannot initiate a transmission to D, since this can potentially cause collisions of the control packets at both B and A, thereby disrupting both transmissions. This is the “exposed terminal” problem since A is exposed to B. An RTS-CTS dialogue can be used to solve the hidden/exposed terminal problem. In Fig. 3.1, when C wants to send a data packet to B, it first sends a Request-To-Send (RTS) message to B. When B receives the RTS, it broadcasts a Clear-To-Send (CTS) message to C and A. When C receives the CTS, it begins to transmit the data packet. Upon receiving the CTS, A will defer its data transmission because it knows B will receive data from C. This method avoids the possible collisions at host B and thus



A is hidden from C and exposed to B

Figure 3.1: The hidden/exposed terminal problems

solves the hidden terminal (A is hidden from C) and exposed terminal (A is exposed to B) problems. During data transmission, the sender sends a data packet (PKT) to the receiver and the receiver immediately responds with an acknowledgement packet (ACK) to the sender if the data packet is correctly received. Failure to receive the ACK will prompt a retransmission after a short timeout. A packet is dropped when no acknowledgement is received after seven retransmissions or there is no buffer to hold the packet.

In the network layer, GloMoSim has implemented several useful routing protocols, such as DSR and AODV to mention a few, which can be used for comparison. Since well-defined APIs between the network layer and the MAC layer and APIs between the network layer and the transport layer are provided in GloMoSim, they greatly facilitate the simulation of our proposed routing protocols.

In the transport layer, the transport layer protocol is the User Datagram Protocol (UDP). We did not use the Transmission Control Protocol (TCP) since it does not perform well in multi-hop wireless networks [GTB99]. If TCP is used, the performance observed in the application layer is highly correlated with the congestion avoidance and congestion control mechanisms in TCP [GTB99]. Therefore, from the application layer, we are not able to investigate the performance difference caused by different network layer protocols. In contrast, UDP does not provide reliable end-to-end transmission, and it does not include congestion control and message retransmission mechanisms. Using UDP, we can easily observe the performance of routing protocols in the application layer.

In the application layer, the simulated traffic was Constant Bit Rate (CBR) traffic: a source node sends packets at a fixed interval time to a destination. Two mobility models, the Random Waypoint model and the Random Drunken model, were simulated. In the Random Drunken mobility model [ZBG98], each node moves independently with the same average speed. Each node moves continuously within the region without pausing at any location. It changes direction, randomly chosen, after every unit of distance. In the Random Waypoint mobility model [ZBG98], each node randomly selects a destination in the simulated area and a speed from a uniform distribution of specified speeds. The node then travels to its selected destination at the selected speed. The transit from one position to another position is called a movement epoch. On arriving at the destination, it is stationary for a given pause time. After that, a new movement epoch begins: the node resumes its movement to a newly selected destination with a newly selected speed.

It has been observed that a certain mobility model may present certain node density fluctuation, that is, the degree of a network node may present periodical changes [RMM01]. This phenomenon is caused mainly by the boundaries of the simulated area: nodes bounce back when meeting the boundaries. It is not clear, however, whether or not the node density changes match real world situations. While most simulation studies assume the validity of using the Random Waypoint model, and the model demonstrates dynamic topology changes desired for performance investigation, it is reasonable to choose the Random Waypoint model for our most simulation studies in this thesis.

## 3.2 Methodology

For each scenario, eight runs with different random seeds were conducted, and the results were averaged. When confidence intervals were calculated, the confidence levels were set to 95%. A confidence interval estimation normally represents the degree of assurance that the actual value falls within the specified intervals [PW93]. All traffic was generated, and the statistical data were collected after a warm-up time of 30 seconds in order to give the nodes sufficient time to finish the initialisation process. We generated the traffic in the beginning of simulation and monitored the

route table changes. Usually, the initial route discovery phases were finished, and data packets began to transmit within 5 seconds. Therefore, 30 seconds of simulation time is long enough for the warm-up period.

### 3.3 Performance Metrics

We evaluated performance according to the following metrics:

- *Packet delivery ratio*: The packet delivery ratio is defined as the ratio of the total number of data packets received by destinations over the total number of data packets transmitted by sources.
- *Average end-to-end delay*: The end-to-end-delay is averaged over all surviving data packets from the sources to the destinations.
- *Normalized Control overhead*: The normalized control overhead is defined as the total number of routing control packets normalized by the total number of received data packets.

We sometimes calculated the control overhead in terms of the total number of bits transmitted for control information. How to calculate the control overhead depends on the size of control packets: we use the total number of packets if the control packets are small; otherwise, we use the total number of bits. This is reasonable since the control overhead depends mainly on the number of times that a radio channel is captured for transmitting when the control packets are small.

Also, we used other metrics to evaluate performance for specific purposes, such as load balancing, energy consumption, etc. These metrics and more detailed parameters are described in the following chapters.

### 3.4 Validation

Wireless mobile networks present extreme challenges in validation of simulations or experiments because the system involves arbitrary movements and unreliable radio transmission environments (such as buildings, trees, vehicles, hills, rain, etc.), making

the simulations or experiments of MANETs not accurately repeatable, and validation can only be approximated [Johnson99].

GloMoSim has implemented the free space radio propagation model and the IEEE 802.11 MAC protocol. The free space radio propagation model is valid for modelling radio transmission in an open area in real systems. We use the simple free space radio propagation model instead of other complex radio models because complex radio models increase simulation complexity and simulation time. Besides, it is reasonable to expect that relative performance of routing protocols should remain the same with different physical layer models [LGT99]. The IEEE 802.11 MAC protocol is used in all real world MANET testbeds and has demonstrated its effectiveness in a small-size real world MANET [MBJ00]. DSR, which is used for comparison in this thesis, is provided by GloMoSim and has been implemented in a real MANET [MBJ00]. We have not validated our proposed protocols in real systems due to the difficulties and the tremendous cost involved in building a real world testbed [MBJ00]. Nevertheless, simulation validation will be a challenging topic deserving further investigation in the future.

### 3.5 Verification and Calibration

The implementations of the free space radio propagation model and the IEEE 802.11 MAC protocol in GloMoSim have been used broadly and demonstrated their correctness. DSR is also included in the GloMoSim library, and all parameter settings for DSR were based on the GloMoSim implementation. This implementation has been accepted and adopted by many academic users.

For each simulation, we first tested it with a static network topology with a small number of nodes to guarantee that the functionalities work correctly. We traced and recorded all route changes and all control messages and stored them in a log file. In a static small network, it is not difficult to verify the correctness of the simulation by analyzing the log file because the route table changes and the control messages can be known in advance. After all functionalities were tested in small static networks, we introduced mobility by using a fixed mobility trace file. We recorded all topology changes, routing table changes, and key events (such as RREQ, RREP messages,

etc.) in a log file. Since the mobility pattern is fixed, we can know when and how the topology changes and what kinds of control messages are sent, which were used to find possible errors in the log file. After passing the test by using several mobility trace files, the simulation was most likely, though not guaranteed, to be correct and valid. In order to compare with the results of other papers [BMJHJ98, DCYS98, DPR00, LGT99], the parameter settings, such as the size of simulation area, the number of nodes, the mobility patterns, the buffer size, the simulation time, etc., are similar to or the same as ones used in these papers. For the same parameter settings, our results show similarity to others', demonstrating that our simulation is correct.

# Chapter 4

## Location Trace Aided Routing

In this chapter, we present a Location Trace Aided Routing (LOTAR) protocol, which is an on-demand scheme that utilises mobile users' location information to assist in routing. The main motivation behind LOTAR is to keep connectivity once a path is established on demand. In LOTAR, the location information is used to reduce the control overhead in the route discovery phase, to search quickly for a feasible path upon link breakage, and to hand off a flow to a stable path if the active one breaks based on prediction. Compared with other on-demand protocols—Dynamic Source Routing (DSR) and Location-Aided Routing (LAR)—simulation studies show that LOTAR can keep a very high packet delivery ratio even in a high mobility environment with acceptable control overhead and end-to-end delay.

The rest of the chapter is organised as follows: in Section 4.1, we briefly introduce related work. In Section 4.2, we present the Location Trace Aided Routing (LOTAR) protocol. Section 4.3 describes link prediction models, and Section 4.4 discusses the criteria for setting parameters of LOTAR. Section 4.5 and Section 4.6 present the simulation model and simulation results respectively. Finally, we conclude the chapter in Section 4.7.

### 4.1 Related Work

Several research studies [KV98, KK00, SL02, SG99] have been done to utilise mobile users' location information to facilitate routing implementation and enhance routing performance. They fall roughly into two categories: the purely geographic forwarding method and the location auxiliary method. In the purely geographic forwarding



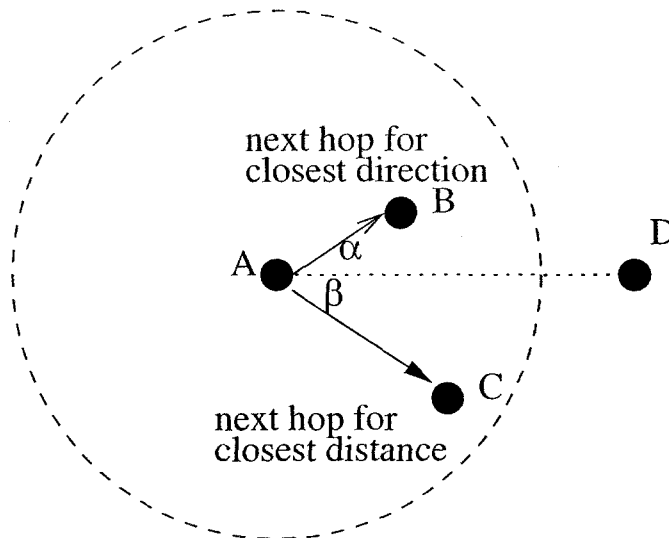


Figure 4.1: The closest direction principle and the closest distance principle

method, a routing decision is made based solely on the mobile users' geographic locations. In the location auxiliary method, routing protocols use the location information as an auxiliary criterion to improve routing performance.

The purely geographic forwarding method can scale well to large networks [KK00, SL02] since no or little route state information is stored. A routing decision in a node is made with the information about the node's current location, its neighbours' locations, and the destination's location. In order to obtain the neighbours' locations, each node announces its position and velocity information to its neighbours. Furthermore, it is supposed to know the locations of the intended message destinations via some mechanism such as a location database query, predefined well-known locations, etc. A localised routing decision is made via heuristic searching rules. These rules include the closest direction principle and the closest distance principle [SL02]. In Fig. 4.1, according to the closest direction principle, node A chooses node B as its next node to destination D since B is the closest direction to D ( $\alpha$  is the smallest positive angle). According to the closest distance principle, A chooses C as its next node to D since C has the closest distance to D. The routing decisions are made in this way step by step until the messages reach the destination.

Several problems must be handled in order to deploy the purely geographic forwarding protocols. First, a purely geographic forwarding protocol must be carefully

designed to deal with the “geographic holes” when a feasible route may actually exist but the local next hop decision cannot be made based solely on the geographic forwarding rules [KK00, SL02]. Second, loop freedom is difficult to guarantee without memorising the forwarded data messages. In [SL02], Stojmenovic and Lin proved that any memory-less direction-based purely geographic forwarding protocol is not loop-free. They proved that distance-based geographic forwarding methods are loop-free except for some specific mobility patterns, which they argued were “unrealistic” cases. Third, location inaccuracy is unavoidable and may result in a high packet loss ratio caused by incorrect local routing decisions. Finally, the environment will affect radio propagation, and the physical locations might not reflect the real network topology. For example, two nodes may be close enough but cannot communicate with each other because of an obstacle (such as a high building) between them; two nodes may be far away but are within each other’s radio range because of favourable echoes, for example, two nodes in a long narrow street between steel-glass buildings. In this case, purely geographic routing decisions that do not take the environment into account are likely to be wrong.

In contrast, the location auxiliary protocols do not need to cope with these difficulties since the location information is used to improve routing performance under the prerequisite that feasible paths can be explicitly claimed to exist for data transmission. In other words, the feasible paths are confirmed by control messages before they are actually used. The Location Aid Routing (LAR) [KV98] protocol searches for routes in a way very similar to the Dynamic Source Routing (DSR) protocol [BJM01] except that it uses location information to limit the flooding range of route request messages during the route discovery phase. LAR can reduce control overhead since the route request messages do not need to be propagated to the whole network, and it is loop-free since it uses source routing to transmit data packets once a feasible path is found (more details are discussed in Section 2.3.3). The Flow Oriented Routing Protocol (FORP) [SG99] selects paths with the longest lifetime, which is predicted with information about the mobile nodes’ speeds and directions. When the predicted lifetime of an active path is lower than a given threshold time, an entire new path will be searched for, and the sender hands off its data stream to the new path.

Obtaining location information will increase system cost, depending not only on

the means of getting and maintaining location information but also on specific application environments. For instance, the Global Positioning System (GPS) is a widely used technique to obtain synchronised location and time information in open areas; the Cricket location-support system [PCB00] can find locations in buildings; the locations of buses can be easily obtained from the bus schedules and a city map in a city public transit system [WHE01]; and the Grid Location Service (GLS) [LHCKM00] can provide a scalable solution for location management. Whether this part of the system cost should be accounted for in routing overhead depends on the availability of the location information from the application layer. In addition, this part of the cost greatly depends on location management mechanisms and is hard to evaluate in routing protocols. For this reason, all purely geographic forwarding and location auxiliary-based routing protocols assume the availability of location information without taking into account location management costs. Although we, as well, will not calculate the location management costs into the routing overhead, we propose a simple location update method and present its bandwidth cost for reference.

## 4.2 Location Trace Aided Routing Protocol (LOTAR)

### 4.2.1 Introduction

Our proposed Location Trace Aided Routing (LOTAR) protocol is a type of location auxiliary method [WH00]. In order to utilise location information, every node must know its own location, its own mobile speed, and the global time. It also knows other nodes' approximate positions and mobile speeds. In addition, we assume that the link lifetime can be roughly predicted in the mobility model. In this chapter, we study two mobility models, namely the Random Drunken model and the Random Waypoint model, and present how to predict link lifetime in each model. For other well-behaved mobile environments where the mobility is predictable or controllable, link lifetime prediction may be easier and more accurate.

In LOTAR, the location information is utilised to limit the route-searching area, predict route lifetime, and hand over a flow to a stable route if necessary. In this chapter, the term “flow” refers to all data traffic between a particular source-destination pair. As a basic requirement, LOTAR can work effectively under the condition that

no location information is available and can work more efficiently with the help of mobility information.

In LOTAR, when a source node has no route information to an intended destination, it initiates a route discovery phase to search for a feasible path. In order to save bandwidth, the flooding area of the route request messages is limited by the “request zone”.

During data transmission, each node along the used path will monitor the connectivity to its prior and next node and predict the link lifetime to its next node. If it predicts the link lifetime to its next node is too short, it will search locally for another path with a longer link lifetime and hand over the flow to the newly-found path (local flow handoff). If a candidate node for local flow handoff could not be found, a global flow handoff message will be sent back to the source, and the source will initiate a new route discovery phase. In order to optimise the used path, if a node predicts that its prior node and its next node are near enough so that they can establish a direct link, it will notify them to do so and try to give up its role as an intermediate node.

Note that flow handoffs may not be able to repair all possible link breakage. If a link breaks due to mobility, an error message will be sent to a sponsor node that is most likely to be the nearest node (among the nodes from the source to the link breakage point) to the destination. The sponsor node will initiate a route discovery to the destination node.

In the following sections, we will describe LOTAR in detail.

## 4.2.2 Basic Data Structures

Each mobile host (node) in the network maintains three basic data structures: Location Table (LT), Routing Table (RT), and Checking Table (CT).

A node maintains an LT (Table 4.1) to keep other nodes' recent locations. Every entry in an LT includes four parts:  $\text{Timestamp}(j)$ ,  $\text{Location}(j)$ ,  $\text{Speed}(j)$ , and  $\text{Direction}(j)$ . The  $\text{Timestamp}(j)$  denotes the time of this entry.  $\text{Location}(j)$  denotes the location information of node  $j$  at the  $\text{Timestamp}(j)$ .  $\text{Speed}(j)$  and  $\text{Direction}(j)$  denote the velocity information (speed and direction) of node  $j$  at the  $\text{Timestamp}(j)$ . For convenience, we use  $\text{Timestamp}_i(j)$ ,  $\text{Location}_i(j)$ ,  $\text{Speed}_i(j)$ , and  $\text{Direction}_i(j)$  to

Table 4.1: Location table for LOTAR

Timestamp	Location	Speed	Direction
-----------	----------	-------	-----------

Table 4.2: Routing table for LOTAR

Source ID	Destination ID	Incoming node ID	Outgoing node ID	Timestamp
-----------	----------------	------------------	------------------	-----------

denote the entry for node  $j$  in node  $i$ 's LT.

If a node is part of an active route in the network, it stores the route information in its RT (Table 4.2). An entry in an RT includes the source and destination IDs, the incoming and outgoing node IDs for the source-destination pair, and the timestamp of the route. The timestamp is used to check if the routing information is obsolete.

A route query message is sent by the source in order to find a path to the destination. The purpose of the CT (Table 4.3) is to prevent an intermediate node from processing the same route query message multiple times. The CT keeps key information on recently received route queries. An entry in a CT includes the source ID, the destination ID, the sponsor ID, and the sequence number of the route query. Note that the sponsor of a route query may be a node other than the source. We will explain the use of the sponsor node in Section 4.2.4.

### 4.2.3 Route Discovery

Route discovery is used to set up a session. The route discovery mechanism in LOTAR is very similar to the one in LAR (Section 2.3.3). However, with the help of the LT table, the “request zone” and “expected zone” in LOTAR can be more accurately and easily defined. With an effective location management mechanism, the information in an LT can be accurate for all entries. However, this idealistic assumption may be difficult to implement in reality. Instead, as a reasonable assumption, a node has more accurate location information about its nearby nodes, while it knows relatively less about the location information about the nodes further away.

We define the “expected zone”  $EZ_S(D)$  of node  $D$ , from the viewpoint of  $S$ , as the circular region with the centre at  $Location_S(D)$  and a radius of  $(\text{current time} - \text{Timestamp}_S(D)) * \text{Speed}_S(D)$ . The “request zone”  $RZ_S(D)$  of node  $D$ , from the

Table 4.3: Checking table for LSR

Source ID	Destination ID	Sponsor ID	Sequence number
-----------	----------------	------------	-----------------

viewpoint of  $S$ , is the smallest rectangular area which includes  $EZ_S(D)$  and the current location of  $S$ . There are other ways to define “request zone”. We can define the “request zone”, for example, as the smallest cone area which includes the source node as the convergence point and the “expected zone.” A smaller “request zone” may reduce the flooding cost but also reduce the chances to find possible paths. As we point out in Chapter 2, how to define a proper “request zone” is still a topic for further study. Since the GloMoSim library includes the implementation of LAR [KV98] using a rectangular area as the “request zone,” and the simulation results in [KV98] demonstrate its effectiveness, we adopt the same definition in LOTAR for easy comparison with LAR.

When node  $S$  wants to find a route to  $D$ , it broadcasts a route query to its neighbours. A route query includes the source ID, the destination ID, the sponsor’s ID, which is the same as the source ID in the route discovery phase, the  $RZ_S(D)$ , and the  $Timestamp_S(D)$ .

We assume that node  $j$  is most likely nearer to  $D$  than node  $i$  if node  $j$  belongs to  $RZ_i(D)$  or  $Timestamp_j(D)$  is newer than  $Timestamp_i(D)$ . When a node  $j$  receives a route query from node  $i$ , it first looks in its CT to check if the same route query has been received before. If this is the first time for node  $j$  to receive this route query, node  $j$  forwards the route query only if it belongs to  $RZ_i(D)$  or  $Timestamp_j(D)$  is newer than  $Timestamp_i(D)$ . Otherwise, it discards this route query packet. Note that the information about  $RZ_i(D)$  and  $Timestamp_i(D)$  are recorded in the route query packet, and  $Timestamp_j(D)$  can be found in node  $j$ ’s LT. Before node  $j$  forwards the route query, it updates the  $RZ_i(D)$  and the  $Timestamp_i(D)$  in the route query packet with the  $RZ_j(D)$  and the  $Timestamp_j(D)$  respectively.

When the route query finally reaches the destination  $D$ ,  $D$  sends back a route reply to  $S$ , using the reverse path, which is recorded in the route query packet as it traverses the intermediate nodes. The source and the nodes along the reverse path update their Routing Table (RT) when the route reply arrives. Fig. 4.2 illustrates an

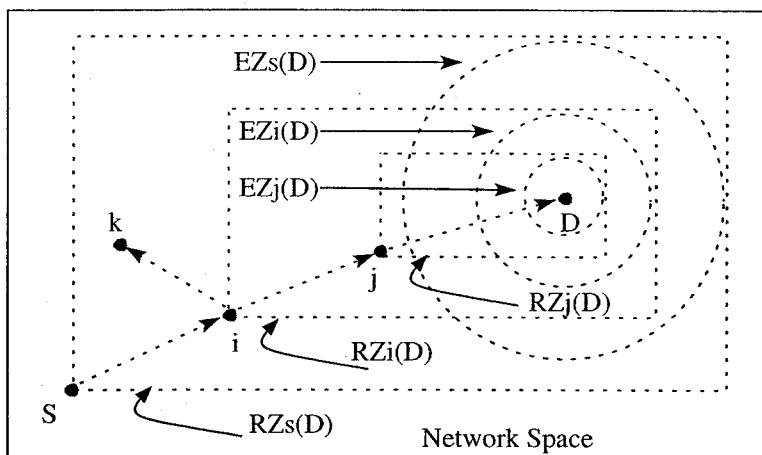


Figure 4.2: An example of the route discovery mechanism in LOTAR

example of how to find a route from  $S$  to  $D$ . In this example, as the route query is forwarded toward the destination  $D$ , the “expected zones” and “request zones” get smaller and smaller. This is the best case in limiting the flooding area. However, the protocol can still work correctly if the “expected zones” and “request zones” do not shrink, since the timestamp depends on the location accuracy model and represents approximately the hop distance to the destination according to our prior assumption that a node knows more recent (thus more accurate) location information about its nearby nodes.

Using the timestamp to aid flooding can enhance the possibility of finally finding the correct routes in case the definition of “request zone” is not suitable for special geographic conditions. For example, if the “request zone” is a dangerous area, no mobile node can go into it. In this case, having a newer timestamp in the LT table represents being nearer to the destination in terms of hop counts. With the help of the timestamp, the possibility of not finding a route in LOTAR is small. However, when a route query times out (the source cannot receive a route reply in a given time), we resort to global flooding to search for a route.

#### 4.2.4 Route Reconstruction

In circumstances in which the nodes’ mobility causes a selected route to be invalid, the Route Reconstruction mechanism is invoked. When a node in the active path becomes unreachable, instead of initiating the Route Discovery mechanism from source

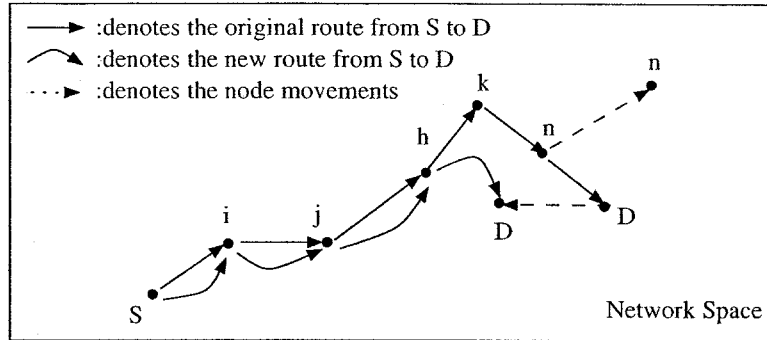


Figure 4.3: An example of route reconstruction

S directly, LOTAR utilises Route Reconstruction from a sponsor node to discover alternative partial routes.

To utilise the channel efficiently, LOTAR uses a simple and short data packet header. Rather than carrying all the nodes in the route, each data packet header in LOTAR contains only the source ID, the destination ID, the sponsor ID, and the timestamp. The purpose of the sponsor and timestamp fields in the data header is to find a proper node to activate the Route Reconstruction in case the link is broken.

Initially, the sponsor ID in the data header is set to the source S, and the timestamp is set to  $\text{Timestamp}_S(D)$ . When a data packet is forwarded from node i to node j along the route, if  $\text{Timestamp}_j(D)$  is newer than the timestamp in the data packet, then the sponsor ID in the data packet is updated to j and the timestamp in the data packet is updated to  $\text{Timestamp}_j(D)$ . Otherwise, the sponsor ID and timestamp in the data packet are unchanged. This way of setting the sponsor ID can find a sponsor node that has the newest destination location information. The sponsor node is most likely the nearest to the destination among the nodes from the source to the link breakage point, because a node nearer to destination D most likely has a newer timestamp in its location table. Fig. 4.3 shows an example of using a sponsor node to search for a partial path.

As shown in Fig. 4.3, if the next node of the route is unreachable at node k, node k will check the sponsor ID in the data header. If the sponsor ID is k, then node k will broadcast a route query to the destination using the method described in the Route Discovery phase. The difference is that the sponsor ID in the route query packet is k rather than the source S. If the sponsor ID is not k, then the sponsor must be a node



on the path from the source  $S$  to node  $k$ . Assuming that sponsor ID is  $h$  as in Fig. 4.3, node  $k$  will send a “sponsor requirement” to node  $h$ . As the “sponsor requirement” is sent back from node  $k$  to node  $h$ , it will erase the entry about this route from all intermediate nodes’ Route Tables. When node  $h$  receives the “sponsor requirement,” it broadcasts a route query to destination  $D$  using the method described in the Route Discovery phase. To avoid loops, any node that already has the corresponding routing information will ignore the route query packet. When destination  $D$  finally receives the route query, it will send a route reply back to the sponsor. Thus the path from the sponsor node to the destination is established.

When data packets are forwarded along a route, the timestamp field in the RT for this route must be updated. If an entry in the RT has not been updated for a long time, it is thought to be obsolete and must be erased in order to save memory.

## 4.2.5 Local Flow Handoff

### Case 1: Avoiding Broken Routes

After a path is established, each node in the path will monitor the connectivity to its next node. We assume that the link lifetime should be predicted in the mobility model. If the predicted lifetime of the link from node  $i$  to node  $j$  is less than a given threshold time,  $T_1$ , then node  $i$  will activate a local flow handoff procedure to avoid possible flow disruption due to a broken link.

In Fig. 4.4, for example, if node  $i$  finds that the lifetime of the link to its next node  $j$  is less than the threshold time  $T_1$ , node  $i$  will search its LT to select a proper node as an intermediate node to node  $j$ . The selected intermediate node must be in the transmission range of both node  $i$  and node  $j$ . In Fig. 4.4, the proper nodes are  $m$  and  $n$ . If we assume that  $t_{im}$  and  $t_{mj}$  denote the lifetime of connectivity from  $i$  to  $m$  and from  $m$  to  $j$  respectively, the minimal value of  $t_{im}$  and  $t_{mj}$  will be the lifetime of the path  $i \rightarrow m \rightarrow j$ . If it is larger than a given threshold time,  $T_2$ , then node  $m$  is a candidate for the local flow handoff. To avoid handing off the flow to a short-life path,  $T_2$  should be larger than  $T_1$ . We choose the best candidate through which node  $i$  and  $j$  can keep the longest lifetime. If node  $i$  cannot find a proper candidate, it will send a Global Flow Handoff (GFH) message to source node  $S$ .

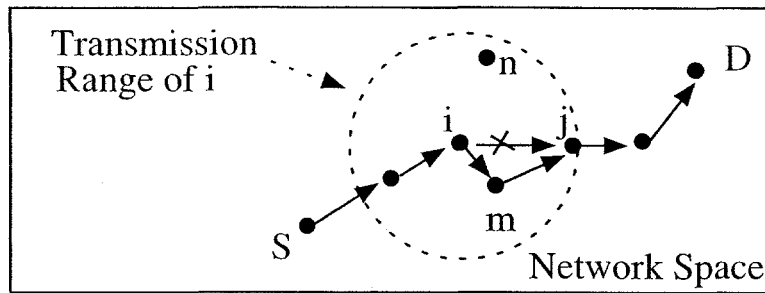


Figure 4.4: An example: local flow handoff case 1

In Fig. 4.4, if we assume that the best candidate is node  $m$ , node  $i$  uses source routing to send a Handoff Requirement (HR) message to  $j$  through node  $m$ . To avoid a loop, node  $m$  will ignore the HR message if it already has the corresponding routing information. If node  $j$  receives the HR message, it sends back a Handoff Acknowledgement (HA) message to node  $i$  through node  $m$ . The HA message will set up the new path between  $i$  and  $j$ , that is,  $i \rightarrow m \rightarrow j$ . The RT in node  $i$ ,  $m$ , and  $j$  will be updated. After the new path is established, the flow will be transferred along the new path.

### Case 2: Improving Active Routes

Note that each local flow handoff described above will expand the route length by one. In order to optimise the route, we propose another local flow handoff that can shorten the route length. When a node in an active path finds that its prior node and next node are near enough, it notifies them to establish a direct link and tries to give up its role as the intermediate node.

In Fig. 4.5, for example, if node  $m$  finds that its prior node  $i$  and its next node  $j$  move near enough to each other that they can communicate with each other directly, it first estimates the lifetime of the link from  $i$  to  $j$ . If the estimated lifetime is larger than the threshold value,  $T_2$ , node  $m$  sends a Resigning Application (RA) message to  $i$ . When node  $i$  receives the RA message, it sends a Direct Link Requirement (DLR) message to node  $j$ . After node  $j$  receives the DLR message, it records the DLR message and sends back an acknowledgement to node  $i$ . When node  $i$  receives the acknowledgement, it updates its next node to node  $j$  and sends a Resigning Sanction (RS) message to node  $m$ . Node  $j$  will change its prior node to node  $i$  after it receives

the DLR message and the data packets of the same flow from node  $i$ . When node  $m$  receives an RS message, it removes the route entry about this path from its RT after finishing the transmission of this flow's packets, which are still in its buffer.

No inconsistency will occur if some of the control messages are lost. The reasons are as follows.

- If the RA, or the DLR, or the acknowledgement from node  $j$  to node  $i$  is lost, there is no change in the RTs.
- If the RS message is lost, the entry in node  $m$ 's RT corresponding to the old path is not erased immediately. However, since node  $i$  and node  $j$  have established a direct link, this entry will be obsolete and eventually will be erased because no traffic from node  $i$  updates this entry.

Errors could occur, however, if two continuous nodes perform the above operation simultaneously. For example, in Fig. 4.5, if node  $i$  and node  $j$  are near, and in the meantime node  $S$  and node  $m$  move near too, then node  $i$  will notify node  $S$  to establish a direct link to node  $m$ , and node  $m$  will notify node  $i$  to establish a direct link to node  $j$ . In this case, the path will be broken even if both direct links  $S \rightarrow m$  and  $i \rightarrow j$  can be successfully established. To avoid this situation, the following rule is added: after a node sends a Resigning Application (RA) message, this node will not respond to any Direct Link Requirement (DLR) message. For example, after node  $m$  sends an RA message to node  $i$ , it will not acknowledge the DLR message from node  $S$ .

We have introduced two different kinds of local flow handoff methods. To avoid sending redundant flow handoff messages, a node uses a semaphore mechanism to block its monitoring operation after it sends a flow handoff requirement message (the HR message in Case 1 or the RA message in Case 2). Once the flow handoff is finished, the node reverts back to its monitoring status. If a node cannot finish a local flow handoff after a given timeout value, it assumes that some flow handoff messages are lost, and it returns back to its monitoring status.

Note that the first case of flow handoff has priority over the second case. The monitoring operations are activated only when a node receives new data packets and is in its monitoring status.

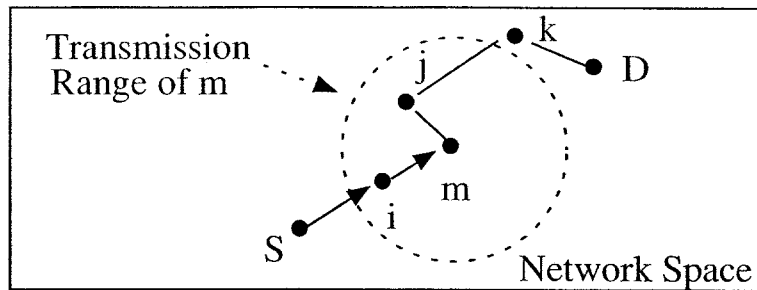


Figure 4.5: An example: local flow handoff case 2

## 4.2.6 Global Flow Handoff

When the source node receives a Global Flow Handoff (GFH) message, it realises that the used route will be broken. It then utilises the Route Discovery mechanism to find a new route to the destination node. Once a new route is established, the source hands over the flow to the new route.

## 4.2.7 Correctness of LOTAR

LOTAR is loop-free for the following reasons:

- For route discovery and global flow handoff, the path established by the route reply is loop-free because the route reply message is sent using source routing, which can easily detect loops and erase them.
- At the route reconstruction phase, the new path from the sponsor node to the destination node is loop-free (for the same reason as above) and does not include any node on the path from the source to the sponsor node. This is because any node that already has the corresponding routing information will ignore the route query packet. Since the partial path from the source to the sponsor node is also loop-free, the whole path from the source to the destination is loop-free.
- For the local flow handoff case 1, any node that will be used for a flow handoff does not include the corresponding routing information. This means that the node which will be added into the path is a new node. So the new path is loop-free.

- For the local flow handoff case 2, the path is shortened if the handoff succeeds or remains unchanged if the handoff fails. In both cases, the new path has no chance to include loops.

In any case, the next-hop pointers are consistent. Temporary inconsistency for prior-hop pointers may exist, however, if some local flow handoff packets are lost. For example, in the local flow handoff case 1, if an HA packet is lost, the prior node of node  $j$  will be to node  $m$  while the next node of node  $i$  is still to node  $j$ . Incorrect prior-hop pointers will result in the losses of packets (error or sponsor requirement messages) going back to the source or to the sponsor node. To overcome this problem, we allow the data packets to update a node's prior node if the inconsistency exists, since the node transmitting data packets for the same flow should include the correct prior node. For example, for the above problem, node  $j$  will update its prior node to node  $i$  after it receives data packets from node  $i$  for the same flow. For local flow handoff case 2, the loss of the acknowledgement packet from node  $j$  to node  $i$  will result in transit prior node inconsistency (node  $j$ 's prior node points to node  $i$  while node  $m$ 's next node is node  $j$ ). This problem can be solved using the same method.

### 4.3 Link Lifetime Prediction Methods

In order to implement flow handoff mechanisms, an effective link lifetime prediction method is indispensable. The link lifetime prediction depends on the characteristics of application environments. Several link lifetime methods have been proposed. In [SG99], the link lifetime is predicted based on mobile hosts' speeds and directions. In [GAPK01], the link lifetime is predicted based on received signal power. If the received signal power falls below a preemptive threshold value, the receiver assumes that the link from the sender to it will break. In [WHE01], link prediction can be made easily based on specific application information such as a timetable and a city map in a city transportation system.

Since the link lifetime prediction method depends on particular mobility models, we consider two different mobility models: the Random Drunken model [ZBG98] and the Random Waypoint model [ZBG98], and discuss the relative link lifetime prediction methods in this section. To simplify the problem, we assume that the received signal

strength in a mobile host depends solely on its distance to the transmitter. In this case, two mobile hosts are assumed to be connected if they are within each other's radio transmission range. A free space propagation model [Rap95] is consistent with this assumption. The free space propagation model is suitable for out-door radio propagation, and it calculates the power of a signal attenuating as  $1/r^2$ , where  $r$  is the distance between mobile hosts. We do not study more complex propagation models in which other environmental factors, such as building obstacles and weather conditions, are taken into account for link prediction. However, any other propagation model and its corresponding link prediction method can be applied within the framework provided by LOTAR.

### 4.3.1 Link Lifetime Prediction in the Random Drunken Mobility Model

In the Random Drunken mobility model (see Chapter 3), the pessimistically estimated lifetime between node  $i$  and node  $j$  is given by [WH00]:

$$\frac{R - L}{v_i + v_j}$$

where  $R$  is the transmission range (assuming all nodes have the same transmission range),  $L$  is the distance between node  $i$  and node  $j$ , and  $v_i$  and  $v_j$  are the speed of node  $i$  and node  $j$  respectively. In the simulation, given  $R$  and the free space radio propagation model, GloMoSim calculates the corresponding maximal radio transmission power so that any node within the range of  $R$  can receive packets from the sender.

The above formula is pessimistic since it assumes two mobile hosts are moving in opposite directions. However, it gives the minimum link lifetime. Considering the fact that two mobile hosts change directions frequently after every unit of distance, the real link lifetime should be larger than this minimum value.

### 4.3.2 Link Lifetime Prediction in the Random Waypoint Mobility Model

Let  $(x_i, y_i)$  and  $(x_j, y_j)$  be the x-y position for node  $i$  and node  $j$ . Let  $R$  be the radio transmission range. Let  $V_i$  and  $V_j$  be the speeds of node  $i$  and node  $j$ , and let  $\theta_i$

and  $\theta_j$  be the directions of node  $i$  and node  $j$  respectively. In the Random Waypoint mobility model (see Chapter 3), the estimated link lifetime between node  $i$  and node  $j$  is given by [SG99]:

$$D_t = \frac{-(ab + cd) + \sqrt{(a^2 + c^2)R^2 - (ad - cb)^2}}{(a^2 + c^2)}$$

$$\text{where } a = V_i \cos \theta_i - V_j \cos \theta_j$$

$$b = x_i - x_j$$

$$c = V_i \sin \theta_i - V_j \sin \theta_j$$

$$d = y_i - y_j$$

The above formula is correct if both mobile hosts are in one movement epoch during the time period  $D_t$ . A more accurate link lifetime prediction should consider the pause time and several movement epochs, but doing so will require more mobility information and increase computing complexity. For simplicity, we use the above formula to provide a rough estimation of the link lifetime between two nodes in the Random Waypoint mobility model.

## 4.4 The Criteria for Setting Parameters

Several parameters in LOTAR need to be clarified. They are the timeout value for Route Discovery, the timeout value for local flow handoff, and the threshold time for link lifetime prediction. The criteria for setting these parameters are as follows.

- The purpose of the timeout value for Route Discovery is to avoid unlimited waiting for the route reply. Since this value is used at the Route Discovery phase, it has little influence on the performance once a route is established. However, this value should be at least two times longer than the estimated maximal round-trip time in the network. On the other hand, too large a value will increase the average route setup time because the source will wait for a long time to activate a new route search if the prior route reply is lost.

- The purpose of the timeout value for local flow handoff is to enable a node to switch back to its monitoring status. As shown in Fig. 4.4, this timeout value should be longer than the time for transmitting a message from  $i \rightarrow m \rightarrow j$  and back. On the other hand, too large a timeout value will keep a node from monitoring for a long time. Note that when a link is broken, the protocol will activate the Route Reconstruction.
- The threshold time for link lifetime prediction depends on the prediction model. There are two threshold values for local flow handoff,  $T_1$  and  $T_2$ . As discussed in Section 4.2.5, the second value should be larger than the first to avoid handoffs to short lifetime paths. Different mobility models and relative prediction models should use different threshold values. A rule of thumb is that too long a threshold time will cause unnecessary flow handoffs and too short a threshold time will result in tardy flow handoffs.

## 4.5 Simulation Settings

We used the same simulation model described in Chapter 3. In order to investigate the benefit gained from location information, we compared the performance of LOTAR with that of the Dynamic Source Routing (DSR) protocol, which is also an on-demand method but does not seek help from location information. The Location-Aided Routing (LAR) protocol, which has a similar design principle to LOTAR, was studied as well. The difference between LAR and LOTAR is that LAR does not include link prediction and flow handoff mechanisms. The simulated LAR is based on version 1 described in [KV98].

We did not compare LOTAR with purely geographic forwarding-based protocols [KK00, SL02], since the rudimentary design goals of LOTAR and the purely geographic forwarding-based protocols are totally different. Unlike the purely geographic forwarding methods, LOTAR explicitly establishes a path before using it, excluding the possibility of incorrect routing decisions caused by signal fading due to a node's surrounding environment, which is an inherent problem in the purely geographic forwarding methods. Without location information, purely geographic forwarding-based protocols cannot work. In contrast, LOTAR can work like a simplified DSR if it lacks



location information.

Two mobility models, the Random Waypoint model and the Random Drunken model, were simulated. In the Random Waypoint model, the *pause time* was set to 0 seconds. In each movement epoch, the speed was uniformly chosen between the *minimal speed* and the *maximal speed*. In the Random Drunken model, the movement granularity was set to 1 metre, that is, each node randomly re-selects a direction every metre. The nodes' speed is controlled by the *mobility interval time*, which indicates how long it takes for a node to travel 1 metre. For example, a *mobility interval time* of 90 ms is equivalent to 40 km/h. In the simulation, the channel capacity of mobile hosts is set to the same value: 2 Mbps. The buffer size was set to 64 packets.

The above are the basic parameter settings used in this chapter. Other parameters are changed for different simulation studies. They are described in the following sections.

## 4.6 Simulation Results

In this section, we perform several simulation studies and present their related results. First, we discuss the mobility features of different mobility models, showing that the network topology changes more frequently in the Random Waypoint model than in the Random Drunken model. Second, we present the simulation results of LOTAR in different mobility models. Third, we introduce a location accuracy model and propose a simple location update mechanism, which is coincident with the location accuracy model. Based on this location update mechanism, we present the performance results of LOTAR in the Random Drunken mobility model. Finally, we evaluate the ideal performance of the flow handoff mechanisms in LOTAR.

### 4.6.1 Mobility Features

In the first experiment, we studied the features of different mobility models. We have noticed that different mobility models have very different topology change frequencies. We assumed that 50 nodes with a radio transmission range of 200 metres moved in a 1500 metre x 300 metre rectangular area and the sampling interval was 500 ms for 900 seconds of simulation time. Fig. 4.6 shows the results of total link changes at

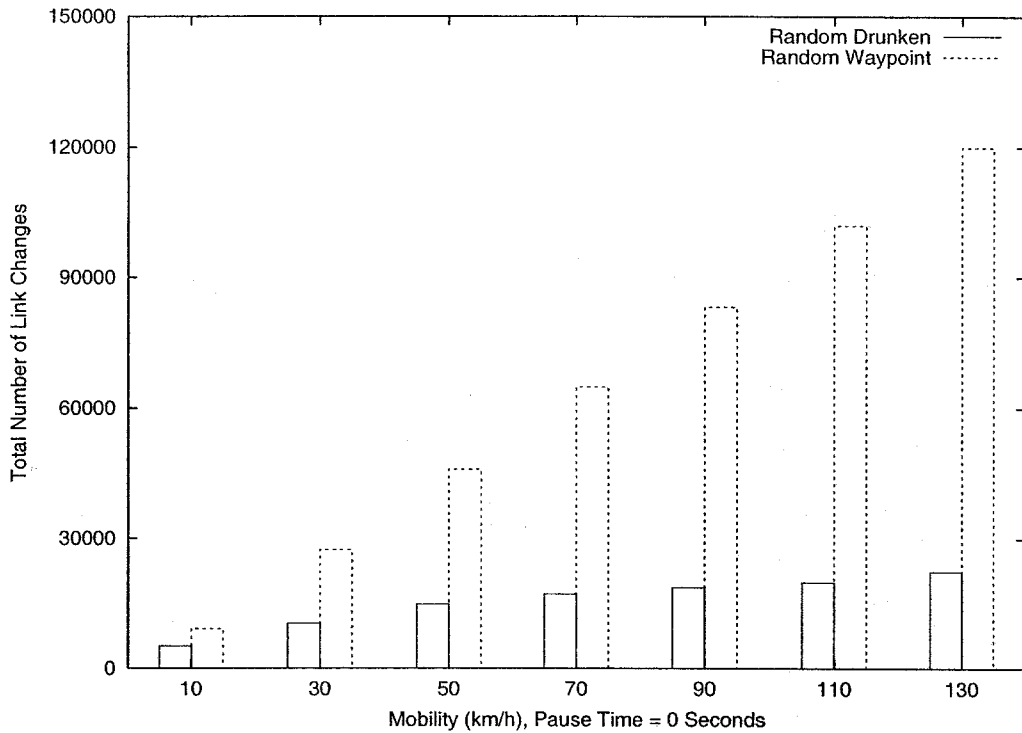


Figure 4.6: Link change frequency in different mobility models

different mobile speeds in different mobile models. As speed increases, the topology changes much faster in the Random Waypoint model. In contrast, links are more stable in the Random Drunken model. Although the Random Drunken model is not likely valid in a realistic application, it can provide us with more knowledge of routing performance in an environment where hosts change directions very quickly but links stay relatively stable.

#### 4.6.2 Performance in Different Mobility Models

In the second experiment, we compared the performance of LOTAR, DSR, and LAR in different mobility models. We assumed that each node moved independently and all nodes had the same transmission range of 200 metres. Furthermore, we assumed that every node knew other nodes' accurate location information. In this simulation, 50 mobile nodes moved in a 1500 metre x 300 metre rectangular region for 900 seconds of simulation time. Compared with a square region, the rectangular region can enlarge the average route length so that we can easily observe the performance difference with smaller simulation time. Initial locations of the nodes were obtained using a uniform

distribution.

In the Random Waypoint mobility model, we changed the minimal speed and the maximal speed to investigate the performance influence of different mobility. As shown in Fig. 4.6, an average mobile speed of 130 km/h can create around 120,000 link changes in the 900 seconds of simulation time. In this highly dynamic scenario, we cannot expect routing protocols to perform well for globally heavy traffic. For example, for 15 CBR traffic with packet interval time of 250 ms, even LOTAR loses over 40% of the data packets at a speed of 130 km/h. Therefore, we randomly selected only 5 source-destination pairs with uniform probabilities to send CBR traffic; three of them had light end-to-end traffic with packet interval time of 400 ms, and two of them had heavy end-to-end traffic with packet interval time of 50 ms. The size of all data packets was set to 512 bytes. For different mobility models, we used different link prediction methods as described in Section 4.3.

Fig. 4.7 and Fig. 4.8 show the results for packet delivery ratios. LOTAR has the highest packet delivery ratio compared with LAR and DSR. In both mobility models, LOTAR can maintain a very high packet delivery ratio at a speed range from 10 km/h to 130 km/h. In the Random Waypoint model, the packet delivery ratios for LAR and DSR decrease quickly at high speed. The reason is that the implementation of LAR does not include a route maintenance mechanism. The intermediate nodes have no salvaging methods to discover alternative routes. So if a path breaks, an error message must be sent back to the source node, and the source node starts a new route discovery phase. During this period, packets may be lost because of the path breakage. For DSR, the cached routes may quickly become obsolete, resulting in more packet losses. In contrast, LOTAR can switch the flow to a different route before the used route is broken and therefore reduces the packet losses due to link breakage. In the Random Drunken model, all protocols perform relatively better than in the Random Waypoint model, due to less frequent topology changes in the Random Drunken model.

Although the overall trend is that the packet delivery ratio becomes smaller when the mobile speed is increased, we observe that the curves in Fig. 4.7 and Fig. 4.8 are not strictly “smooth.” This is because the connectivity of the network is different at different speeds. As the average speed increases, for a given simulation time,

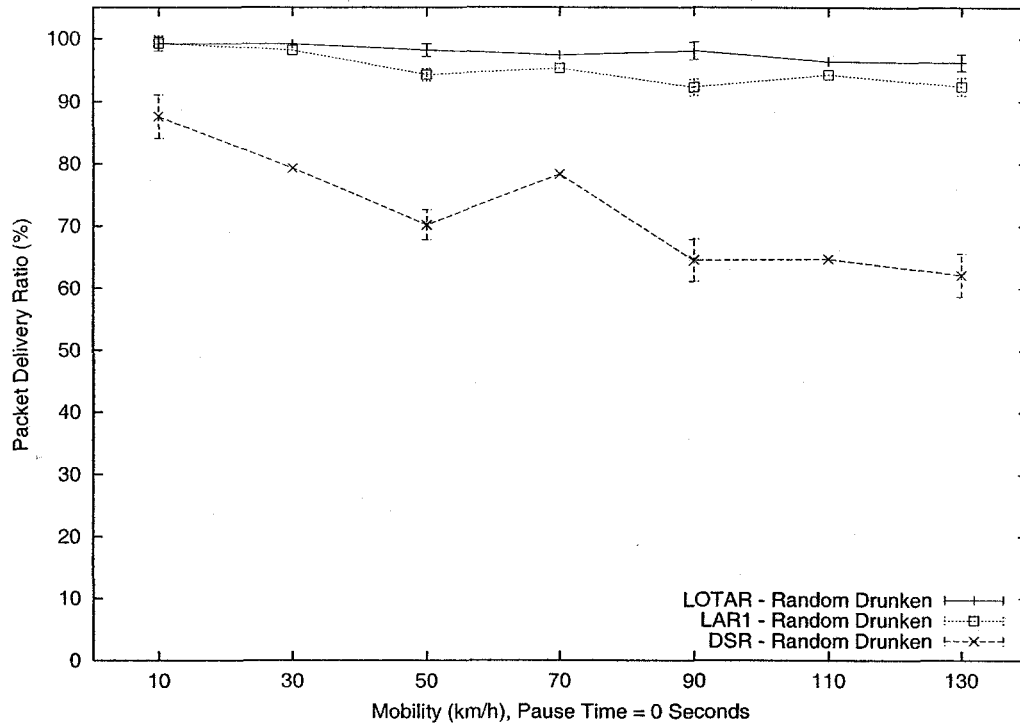


Figure 4.7: Packet delivery ratio in the Random Drunken model

the number of simulated nodes' movements increases. Thus, a particular network configuration, such as the occurrence of partitions, that may not have occurred at a lower speed could occur at a higher speed. This has been observed in [KV98, BMJHJ98] as well. A network partition is a temporary state that occurs when a network node has no way to reach another network node. This is likely the main reason for the non-monotonic curves.

Fig. 4.9 and Fig. 4.10 show the average end-to-end delay in different mobility models. In the Random Drunken model, LOTAR has the highest average end-to-end delay, followed by DSR and LAR. The average end-to-end delay depends on the number of packets received by the destination, the hops that the received packets traverse, and the waiting delay in the intermediate nodes. LOTAR has the largest average end-to-end delay because the local flow handoff will contend with data packets for a radio channel. DSR has slightly higher average end-to-end delay than LAR due to the influence of cached obsolete route information. In the Random Waypoint model, the same phenomena can be observed at high speed. However, the large confidence interval in Fig. 4.10 shows that the difference might not be significant.

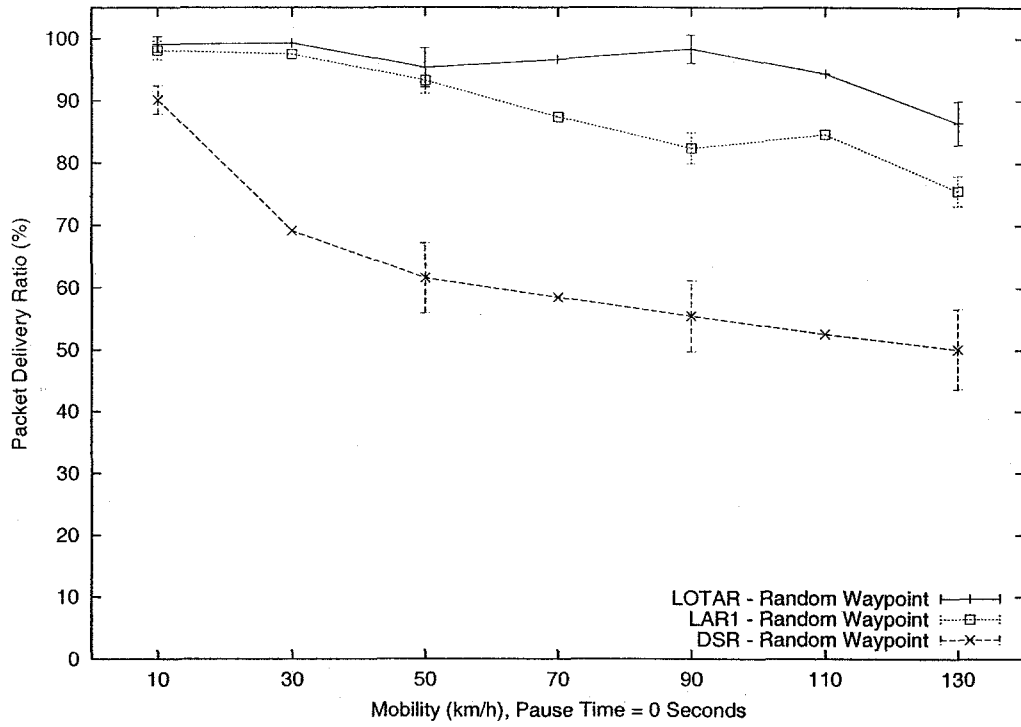


Figure 4.8: Packet delivery ratio in the Random Waypoint model

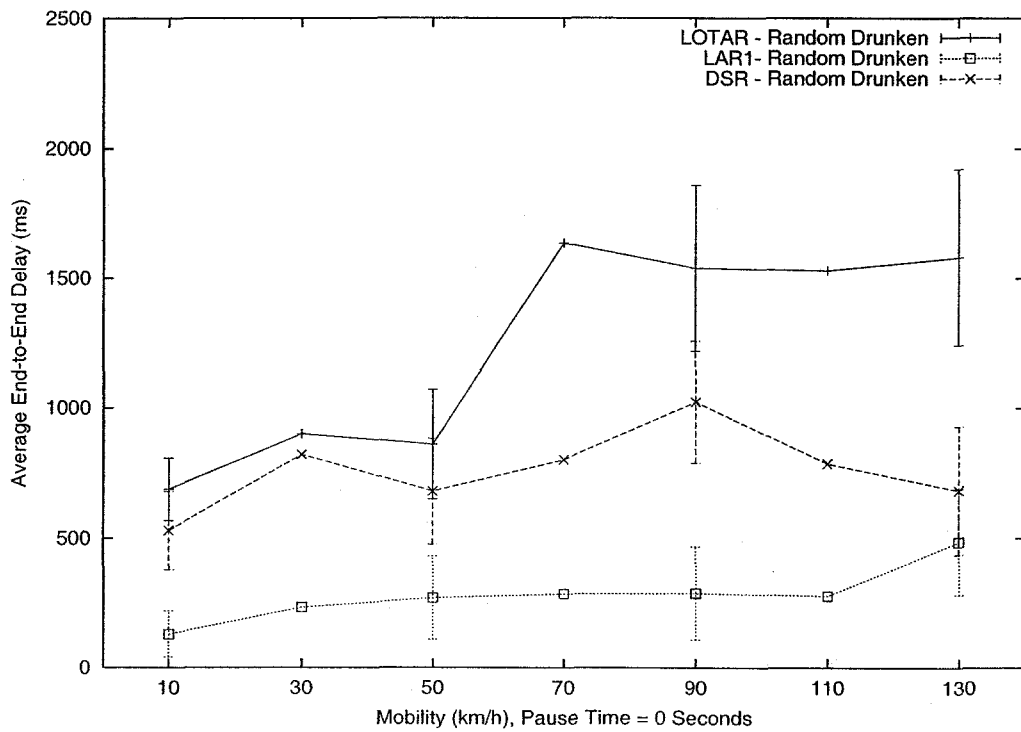


Figure 4.9: Average end-to-end delay in the Random Drunken model

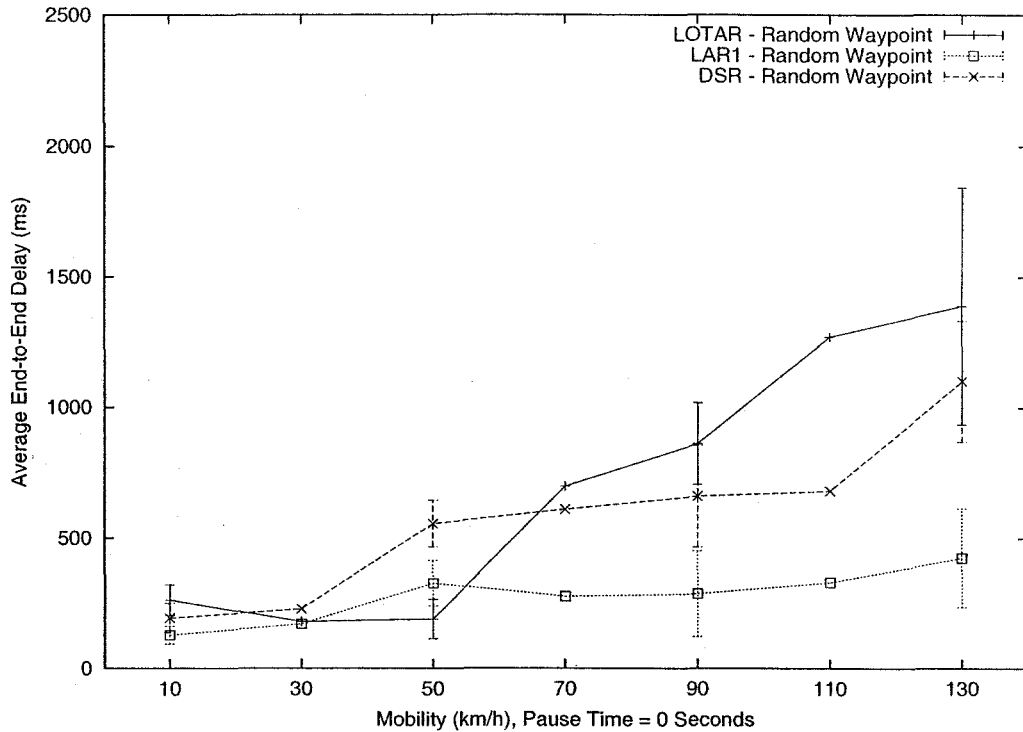


Figure 4.10: Average end-to-end delay in the Random Waypoint model

Fig. 4.11 and Fig. 4.12 show the results of control overhead. At low speed, LOTAR, LAR and DSR have no significant differences. However, in the Random Waypoint model, LOTAR has the highest control overhead at high speed, followed by LAR and DSR. In the Random Drunken model, DSR presents the highest control overhead at high speed. The main reason is that mobility has relatively less influence on the control overhead of DSR due to the fact that any intermediate node having routing information to the destination can reply to route requests. Therefore, compared with LOTAR and LAR, which do not include route-caching mechanisms, the difference in control overhead in different mobility models (thus different topology change frequency) for DSR is not large, resulting in different relative relations among the three protocols in different mobility models.

In summary, the second experiment demonstrates that LOTAR can keep very high packet delivery ratios in both mobility models. In terms of average end-to-end delay, LOTAR, DSR, and LAR have no significant differences. As for control overhead, DSR has the highest control overhead in the Random Drunken model. In the Random Waypoint model, it has the highest control overhead at low mobility and

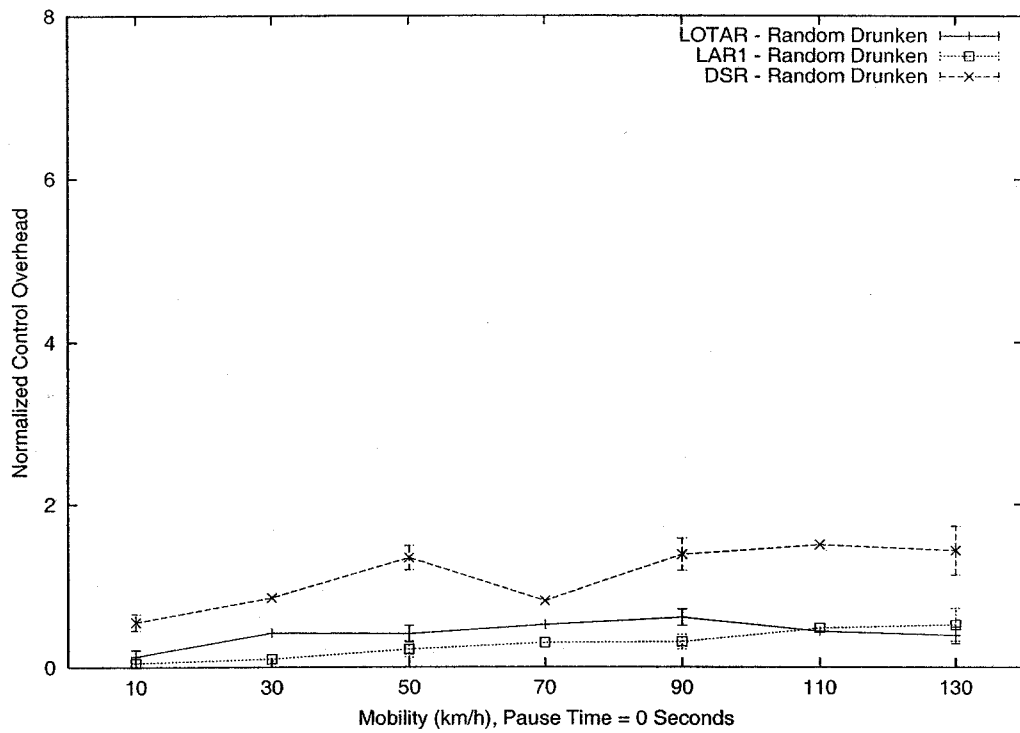


Figure 4.11: Control overhead in the Random Drunken model

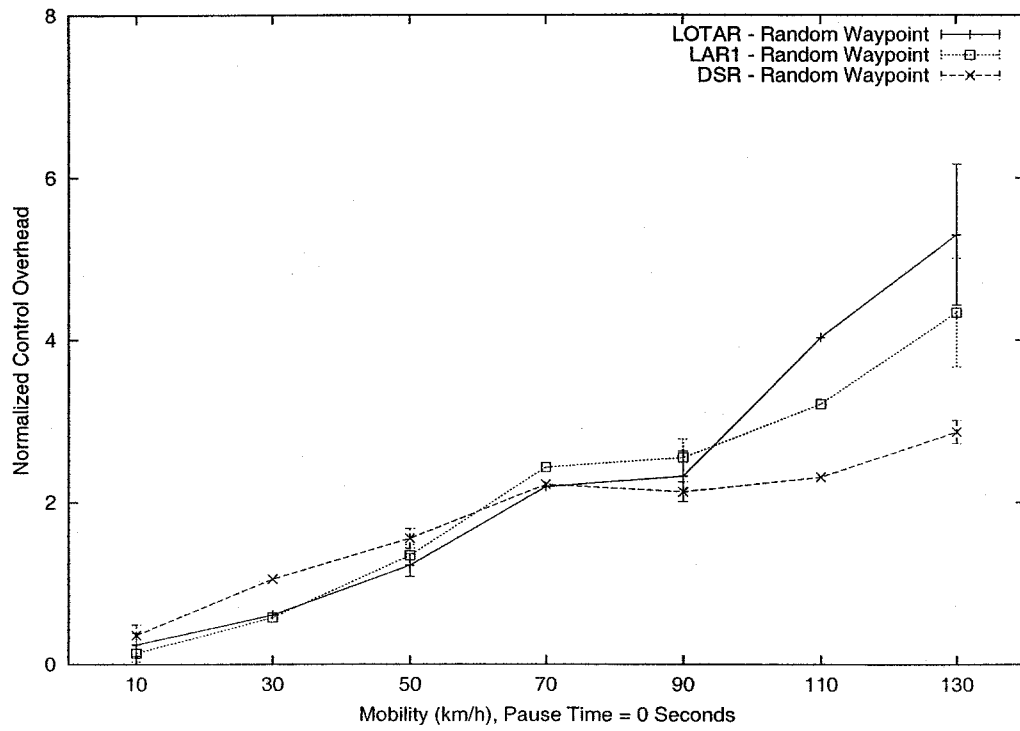


Figure 4.12: Control overhead in the Random Waypoint model

the smallest one at high mobility.

### 4.6.3 The Influence of the Location Update Mechanism

The results shown in the second experiment are based on the assumption that a node knows exactly the locations of destinations without additional routing overhead. This assumption may not be true due to location inaccuracy. In the third experiment, we presented a location accuracy model and studied the influences of mobility, radio transmission range, and location update intervals on the performance of LOTAR with inaccurate location information.

It is necessary to mention that the assumption of location information available to routing protocols might be too strong in real systems. In the circumstance that obtaining and maintaining location information requires non-trivial system cost, all location-based (pure geographic forwarding and location auxiliary) routing protocols might not be overall efficient schemes. Nevertheless, location-based routing can be an efficient approach in the systems in which location management is a component in the application layer and location information is re-usable by the network layer, or location information can be obtained easily (such as city transit systems and tourist guide systems). For example, schemes that require periodic flooding of location information may benefit performance-wise by combining the routing with the location update.

In order to investigate how location inaccuracy can affect the performance of LOTAR, we propose a simple location update method in the following section. Since all other location-based routing protocols do not take location management into account, for a clear and fair comparison, we separately calculated the location update cost and the routing cost.

#### The Location Accuracy Model

We propose a simple location updating method as follows. Every node exchanges its LT table with its neighbours at a fixed interval time called the location update interval time. If a node receives an entry with a newer Timestamp, it updates its LT with the newer entry. Initially, the LT table is empty in every node. Eventually, every node's LT table will include the location information about all nodes in the network.



In this way, a node has more accurate location information about its nearby nodes, because every node exchanges its LT table only with its neighbours at each location update interval and thus it requires more update intervals for the more distant nodes' location information to arrive.

According to the above observation, the location accuracy degrades with nodes' distances. In other words, a mobile host A knows a mobile host B's position with location inaccuracy  $k*d$ , where  $k$  is a small constant factor and  $d$  is the distance between A and B. Note that the location accuracy model greatly depends on the specific location update and location management mechanism. The above model is suitable for our proposed location update method. However, we do not expect it to be suitable for all location management mechanisms.

### **The Impact of Mobility**

Considering the scalability problem when taking the location update mechanism into account in the simulation, we studied the performance in a network with fewer nodes and in the Random Drunken mobility model, which is relatively stable. We studied the performance of LOTAR, LAR, and GSR. The GSR protocol (Section 2.2) is studied because our location management mechanism is very similar to the link exchange method in GSR.

In this experiment, we ran simulations under various mobility conditions from 10 km/h to 130 km/h for 30 minutes of simulation time. 30 nodes moved in a 1000 metre x 1000 metre square region. We set the radio transmission range at 400 metres. Two randomly chosen source-destination pairs sent CBR traffic with packet interval time of 50ms. The data packet size was set to 768, which made the end-to-end traffic very heavy and allowed us to observe the performance difference easily. We set the link state update interval time in GSR to 15 seconds and the location update interval time in LOTAR to 30 seconds. The simulation studies show that the location update interval time chosen in LOTAR is accurate enough to obtain a high packet delivery ratio.

Fig. 4.13 shows the results for packet delivery ratios. LOTAR has the highest packet delivery ratio compared to GSR and LAR. We see that LOTAR can maintain a very high packet delivery ratio in the speed range from 10 km/h to 130 km/h, while

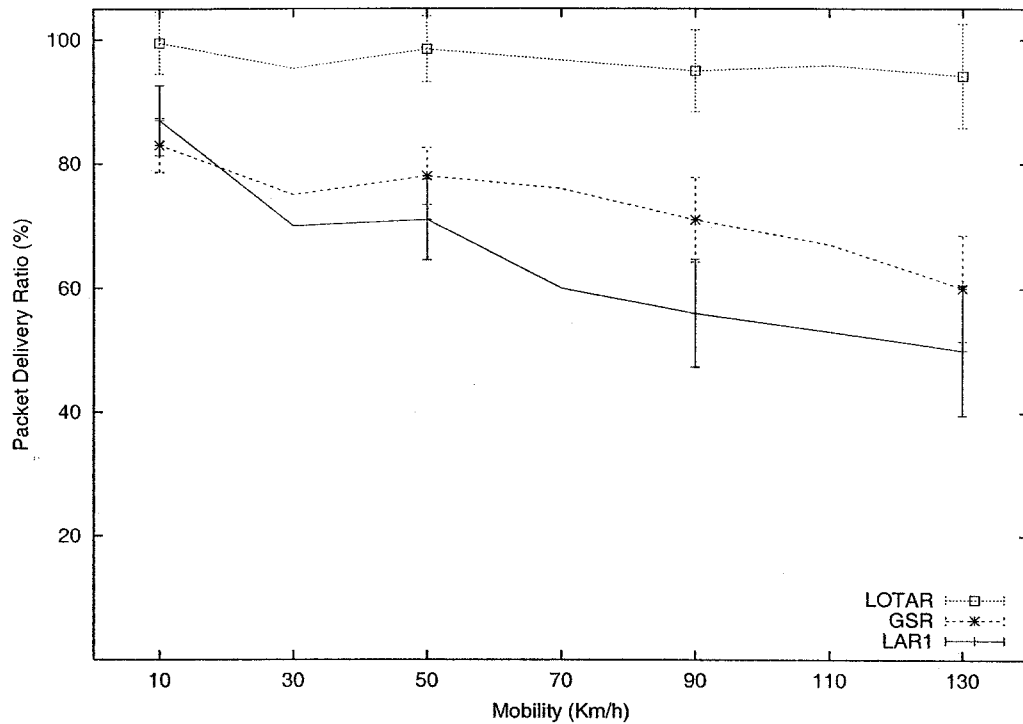


Figure 4.13: Packet delivery ratio with mobility

the packet delivery ratios for LAR and GSR decrease with increased speed. The packet delivery ratio for LAR decreases significantly at high speed. The reason is that the implementation of LAR does not include a route maintenance mechanism. When the mobile speed is high, most route error messages cannot successfully reach the sender due to heavy end-to-end traffic and the intermediate nodes have no salvaging methods to discover alternative routes. In contrast, LOTAR has effective ways to search for an alternative route when a link is broken. We observed in the simulation that most sponsor nodes are near the broken link. This means that most sponsor messages need not be sent back to the source node. In addition, LOTAR includes flow handoff mechanisms that can switch the flow to a different route before the used route is broken.

Fig. 4.14 shows the average end-to-end delay. The average end-to-end delays for LOTAR, GSR, and LAR are similar. At high mobility (130 km/h), however, LOTAR has the highest average end-to-end delay, followed by GSR and LAR. The average end-to-end delay depends on the number of packets received by the destination and the hops that the received packets traverse. As seen in Fig. 4.13, a large number of

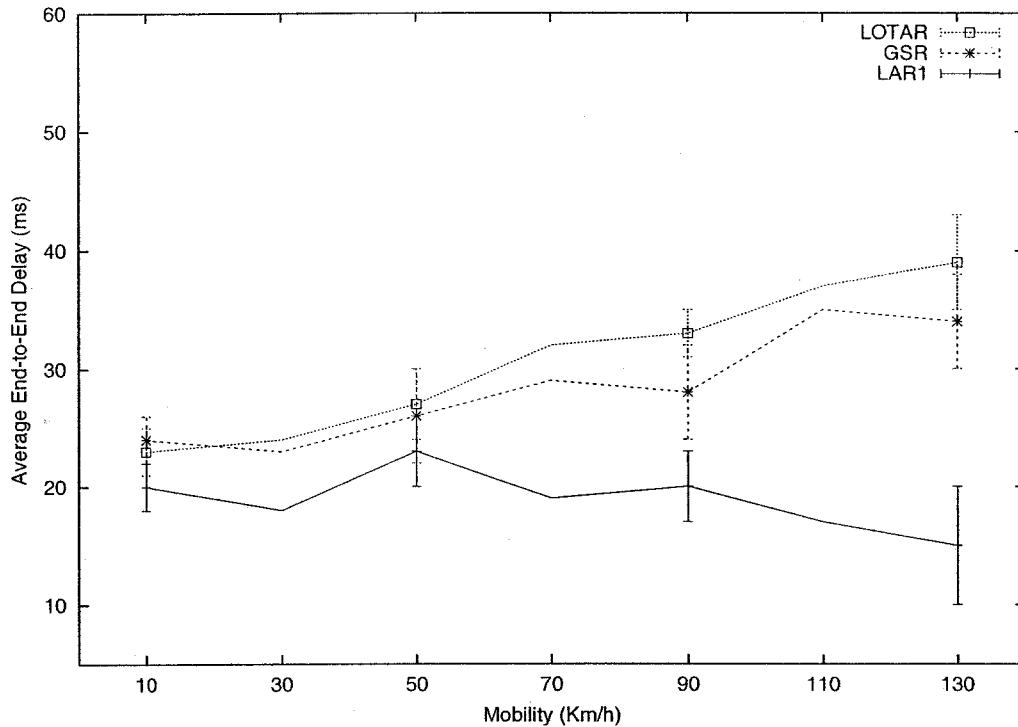


Figure 4.14: Average end-to-end delay with mobility

packets are dropped in LAR when the speed is high. This contributes to the relatively low average end-to-end delay for LAR. LOTAR has the largest average end-to-end delay at high speed because more local flow handoff messages will contend with data packets for a radio channel. The average end-to-end delay for LOTAR is acceptable anyway, considering that LOTAR can keep a high packet delivery ratio even with high-speed mobile nodes.

Fig. 4.15 shows the result of control overhead. Note that we separate the location update cost and routing cost for LOTAR in the figure. Since the control packets in GSR are large, we calculate the control overhead as the total number of bits in control packets averaged by simulation time. GSR has larger control overhead than LAR and LOTAR because a node in GSR periodically changes large link state information with its neighbours. The difference in control overhead for LAR and for LOTAR is not significant.

GSR also has a larger overhead than the location management cost in LOTAR. This is because the update interval time for location information in LOTAR is larger than the update interval for link state information in GSR. If the update interval time

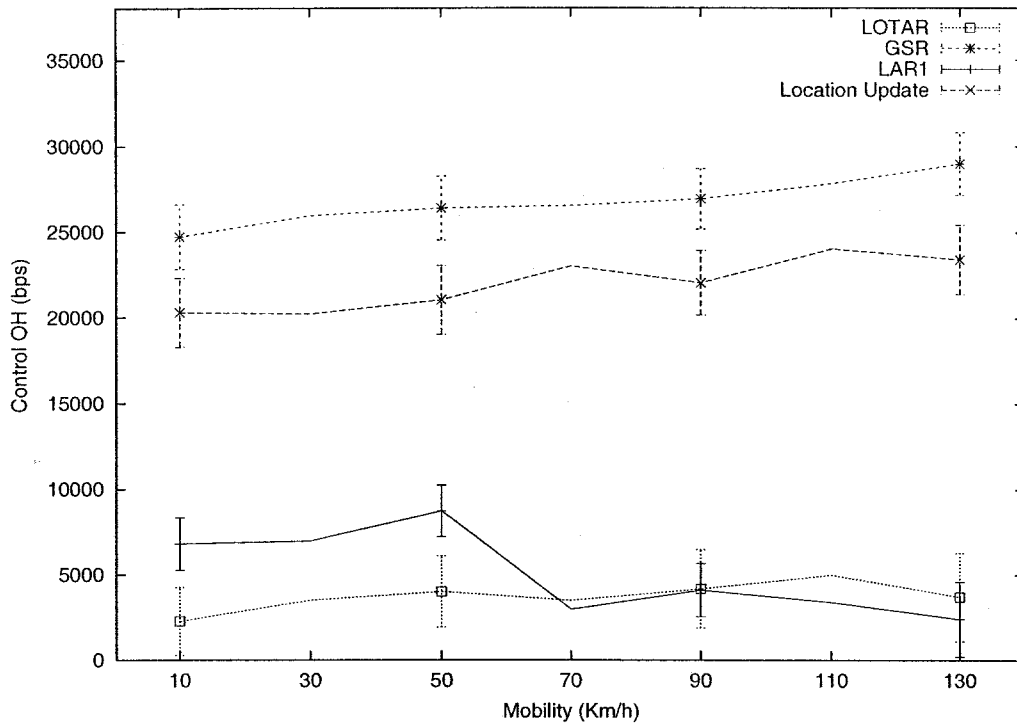


Figure 4.15: Control overhead with mobility

for the link state information in GSR and for location information in LOTAR is set to the same value, such as 30s, the control overhead in GSR will be lower. But the route inaccuracy [CG98] in GSR will be higher, and thus the packet delivery ratio for GSR will be lowered further.

### The Impact of Radio Transmission Range

The range of radio transmission determines the degree of node connectivity. The degree of a node's connectivity is equal to the number of the neighbouring nodes that can be reached in one hop. Table 4.4 shows the average degree of connectivity and the maximal end-to-end hops in the initial networks. X in the table means that there is a partitioned network among the initial networks. As shown in Table 4.4, the larger the transmission range, the larger the connectivity degree, and the smaller the maximal end-to-end hops.

In order to study the impact of radio transmission range, we chose 50 mobile nodes and varied the radio transmission range from 300 metres to 700 metres. We set a moderate moving speed, 60 km/h.

Table 4.4: Network characteristics in different transmission ranges

Radio range	Maximal end-to-end hops		Avg. degree of connectivity	
	30 nodes	50 nodes	30 nodes	50 nodes
200m	X	X	2.60	4.16
300m	X	6	6.07	9.84
400m	4	4	9.87	16.32
500m	3	3	15.20	23.00
600m	2	3	19.70	30.36
700m	2	2	22.20	36.28

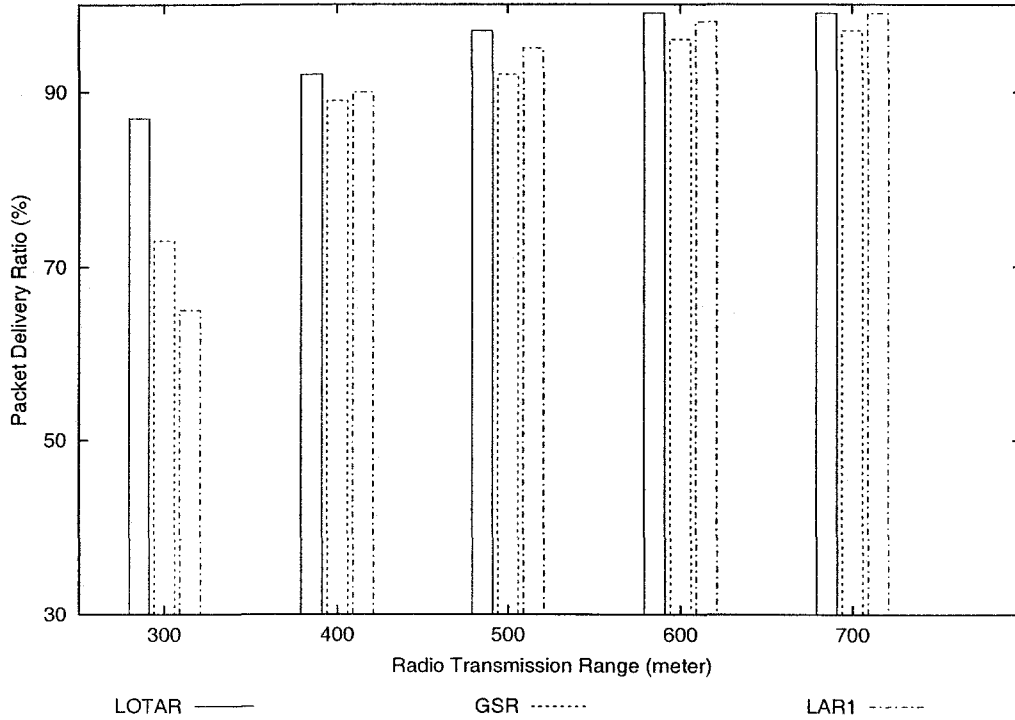


Figure 4.16: Packet delivery ratio at different transmission ranges

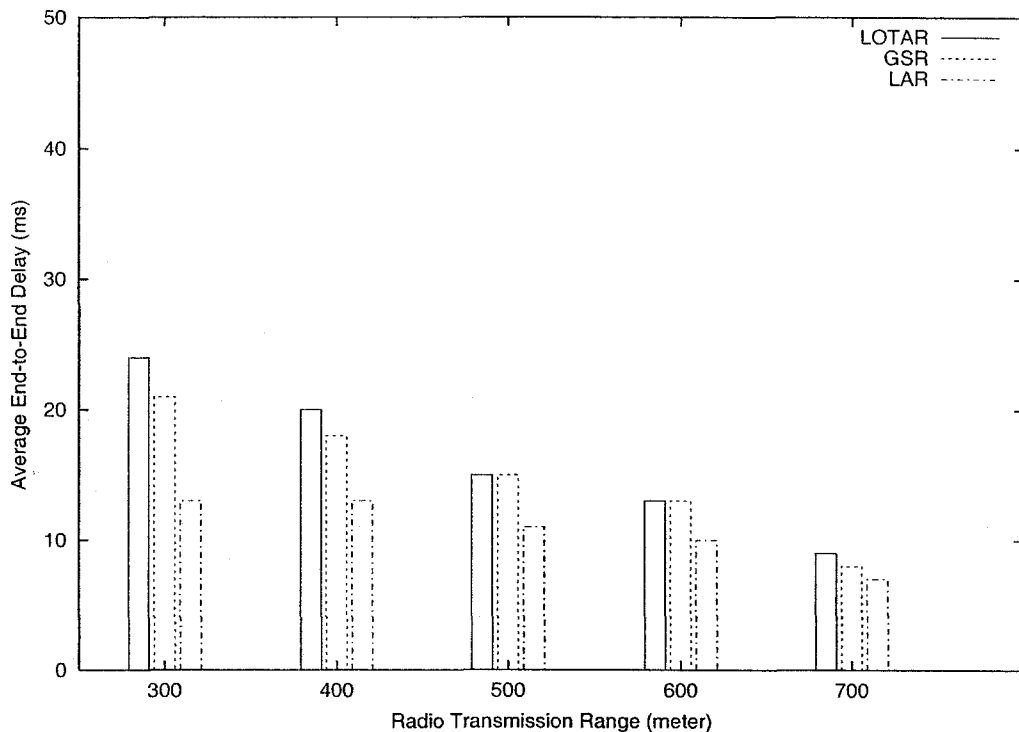


Figure 4.17: Average end-to-end delay at different transmission ranges

Fig. 4.16 shows the packet delivery ratios at different transmission ranges. As the transmission range increases, the packet delivery ratios for all protocols increases, since the average hops that the packets traverse are less and thus the packet loss rate will decrease. The packet delivery ratio for LOTAR is the highest for all transmission ranges. However, as the transmission range increases, the difference among the three protocols becomes smaller. When the transmission range is 700 metres, the packet delivery ratios of the three protocols are almost the same. This is because more nodes can be reached in one hop without requiring a routing decision. Path breakage and path re-discovery has little influence in this case.

Fig. 4.17 shows the average end-to-end delay at different transmission ranges. As the transmission range increases, the end-to-end delay for all protocols is reduced and the difference for all three protocols becomes smaller because the packets traverse fewer hops.

Fig. 4.18 illustrates the result of the control overhead at different transmission ranges. As the transmission range increases, the control overhead for GSR and the location update cost for LOTAR increase. The larger control packet size for GSR

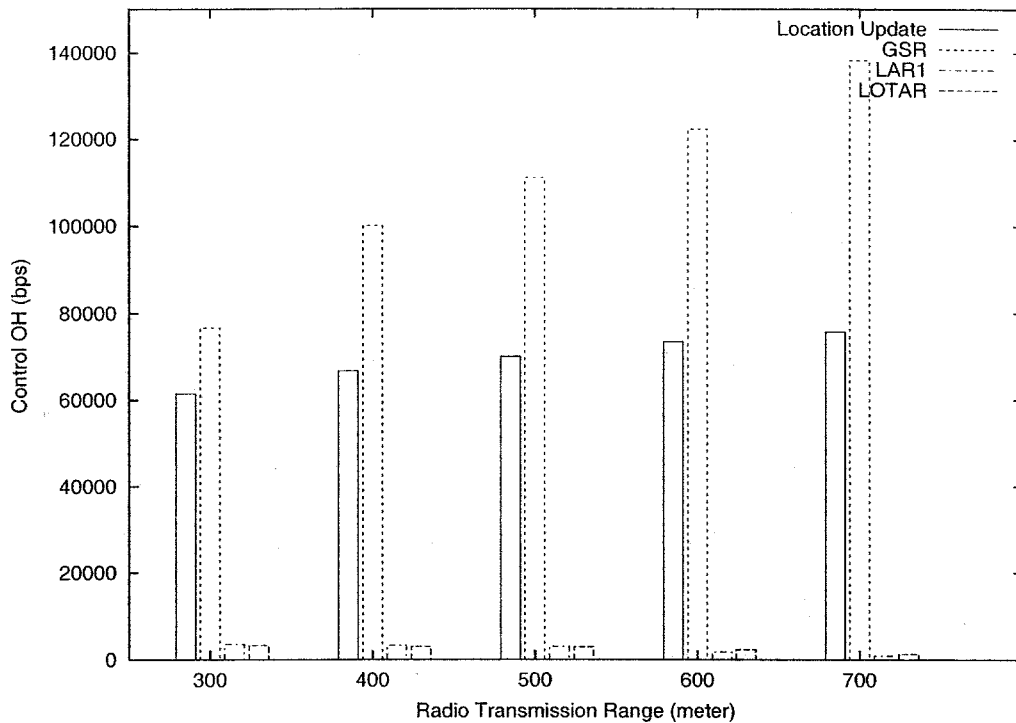


Figure 4.18: Control overhead at different transmission ranges

should be used when the degree of node connectivity is larger because more link state information is included in the control packets. In LOTAR, a larger degree of node connectivity means that a node will locally broadcast location information to more nodes. Compared with the control overhead for GSR and the location update cost for LOTAR, the control overhead for LAR and for LOTAR demonstrates little change with different transmission ranges.

### The Impact of Location Update Intervals

In this section, we study the impact of the update interval for location information in LOTAR. In this experiment, we chose three different location update interval values: 2 seconds (small), 30 seconds (moderate), and 2 minutes (large) to illustrate performance differences.

Fig. 4.19 shows that the packet delivery ratio of LOTAR is higher at the update interval of 30 seconds than at the update intervals of 2 seconds and 2 minutes. Fig. 4.20 gives the results of average end-to-end delay. We can see that the average end-to-end delay is larger at the update interval of 2 seconds than at the update

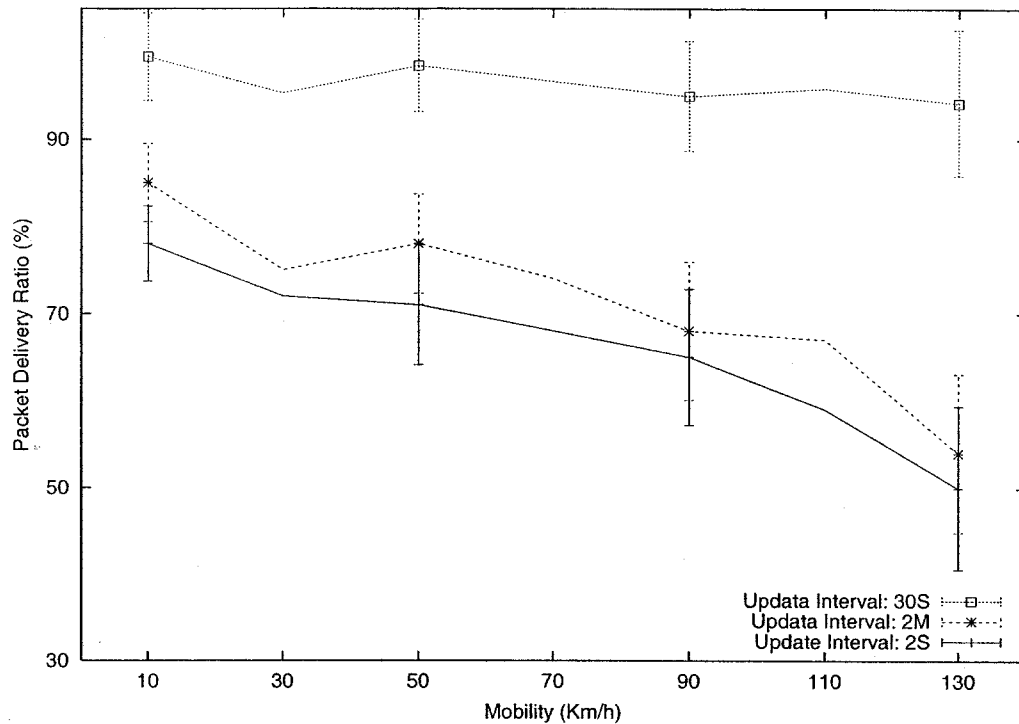


Figure 4.19: The impact of location update interval on packet delivery ratio

intervals of 2 minutes and 30 seconds. Fig. 4.21 gives the results of location update cost at different update interval values. The location update cost is much higher at the update interval of 2 seconds.

If the update interval is too large, the location information is not accurate. Thus, the link lifetime prediction based on the location information is not correct, causing unnecessary or incorrect flow handoff. That is the reason that the packet delivery ratio is low at a large update interval as shown in Fig. 4.19. On the other hand, if the update interval is too small, each node exchanges its location information more frequently, leading to the higher location update cost at a smaller update interval as seen in Fig. 4.21. Although smaller update intervals can provide more precise location information, the location update cost must be controlled in a proper range in order to avoid network congestion. Otherwise, more control packets will contend for resources with the data packets and thus increase the data packet loss rate and end-to-end delay. From the figures, we note that the performance of LOTAR at too small a location update interval is worse than at too large a location update interval.

In summary, we studied the influence of the location update mechanism in the



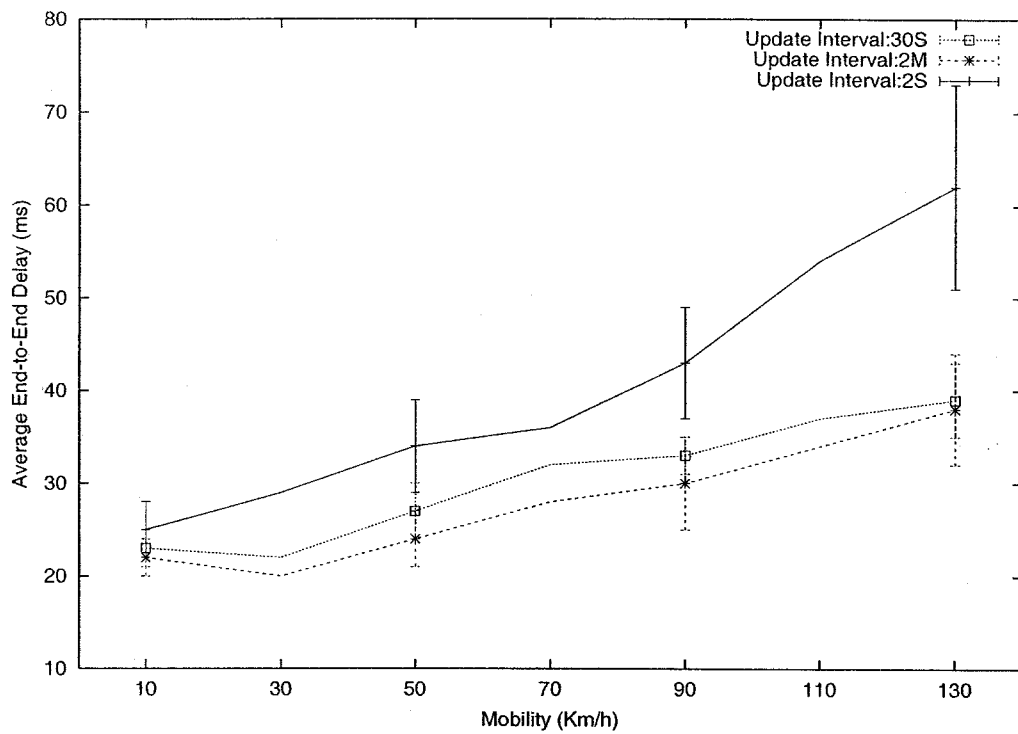


Figure 4.20: The impact of location update interval on average end-to-end delay

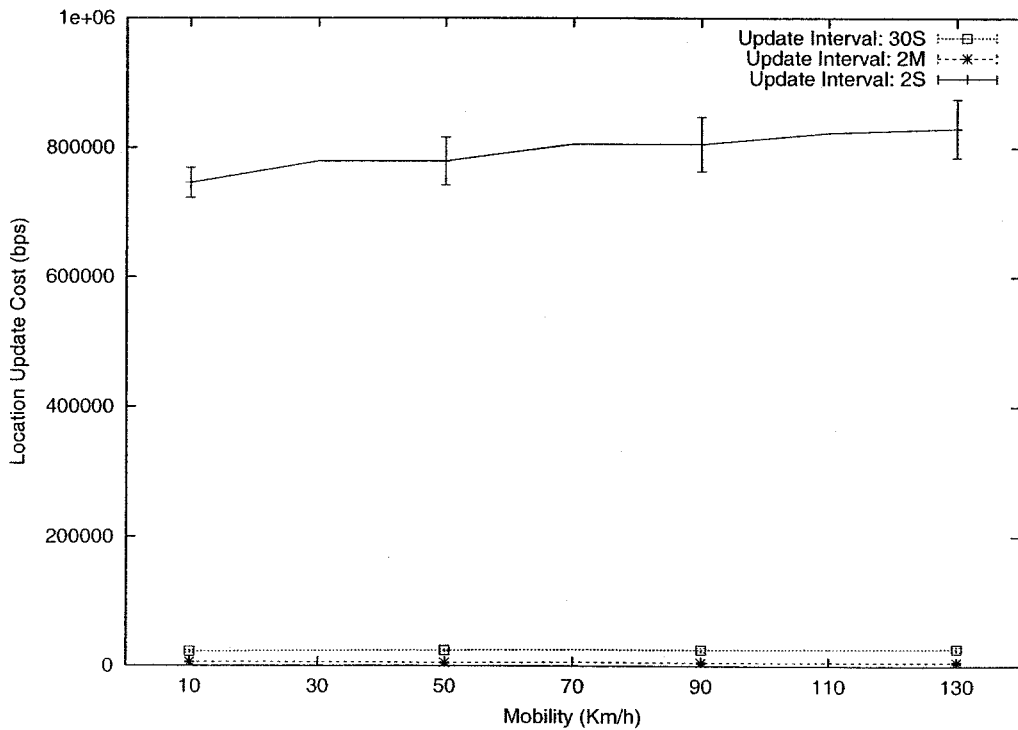


Figure 4.21: The impact of location update interval on control overhead

third experiment. The results illustrated the efficiency of LOTAR in maintaining high packet delivery ratios with acceptable average end-to-end delay and control overhead. In addition, in order to obtain good performance, the location update intervals should be carefully selected.

#### 4.6.4 Evaluation of Flow Handoff- Ideal Performance

Through the simulation studies described in Section 4.6.2 and Section 4.6.3, we have seen that flow handoff may help to maintain longer end-to-end connectivity time and higher packet delivery ratios. In the fourth experiment, we try to answer the following question: is the local flow handoff mechanism effective enough to maintain good performance? What is the ideal performance if flow handoffs are always successful and in time?

Theoretically, successful and on-time flow handoff (local and global) should keep continuous end-to-end connectivity if the source and the destination are never partitioned during the session (two nodes are called partitioned if there is no path between them in the network). However, these ideal assumptions may not be practically true due to handoff control packet losses and inaccurate link prediction. In order to obtain ideal handoff performance, we eliminate the influence of data traffic in the simulation. To do so, we simulated a simple scenario in which there is no data traffic and each local/global flow handoff is activated by periodical link prediction (periodical sampling). In addition, we assumed that there was an offline algorithm with whole topology information to calculate the shortest path from a source to a destination. The following metrics were evaluated.

- *Average end-to-end connectivity*: The average end-to-end connectivity is defined as follows:

$$\frac{\sum_{i=1}^K T_i}{T * K}$$

where T is the simulation time (excluding the warm-up time);  $T_i$  is the connectivity time for the i-th end-to-end connection during the simulation time (excluding the warm-up time); and K is the total number of end-to-end connections. All connections are established during the warm-up period and last

to the end of simulation. A source-destination pair is considered connected if the source knows the correct route information to the destination. Otherwise, they are considered partitioned even if there is a feasible path from the source to the destination determined by the offline algorithm.

- *Path length ratio*: The path length ratio is defined as follows:

$$\frac{\sum_{j=1}^{M_1} \frac{L_j^1}{SL_j^1} * T_j^1 + \sum_{j=1}^{M_2} \frac{L_j^2}{SL_j^2} * T_j^2 + \dots + \sum_{j=1}^{M_k} \frac{L_j^k}{SL_j^k} * T_j^k}{T * k}$$

where T is the simulation time (excluding the warm-up time); k is the total number of end-to-end connections; and  $M_i(1 \leq i \leq k)$  is the number of time periods during which the path remains unchanged for the i-th end-to-end connection.  $\frac{L_j^i}{SL_j^i}$  expresses the ratio of the path length obtained by the evaluated routing protocol over the shortest path length obtained by the offline algorithm in a time period  $j(1 \leq j \leq M_i)$ , during which the path keeps unchanged for the i-th end-to-end connection.  $T_j^i$  is the duration time of time period  $j(1 \leq j \leq M_i)$  for the i-th end-to-end connection. The larger the path length ratio, the longer the path used in the evaluated routing protocol.

- *Control overhead*: The control overhead is defined as the total number of routing control packets averaged over the number of end-to-end connections.

We simulated 50 mobile nodes moving in a 1500 metre x 300 metre rectangular region for 900 seconds of simulation time. The mobility model was the Random Waypoint model. Five source-destination pairs are randomly chosen to establish end-to-end connections. Since there was no data traffic, the link check/prediction was activated periodically in an interval time of 50 ms (sampling granularity). We studied the performance of LOTAR, LOTAR1 (LOTAR without global handoff), and LOTAR2 (LOTAR without local or global handoff).

Fig. 4.22 indicates the results of average end-to-end connectivity. As expected, LOTAR exhibits very high end-to-end connectivity, while LOTAR2 has much lower end-to-end connectivity than LOTAR and LOTAR1. LOTAR1 has very similar end-to-end connectivity to LOTAR, meaning that local flow handoff is good enough to keep high end-to-end connectivity in our simulated networks.

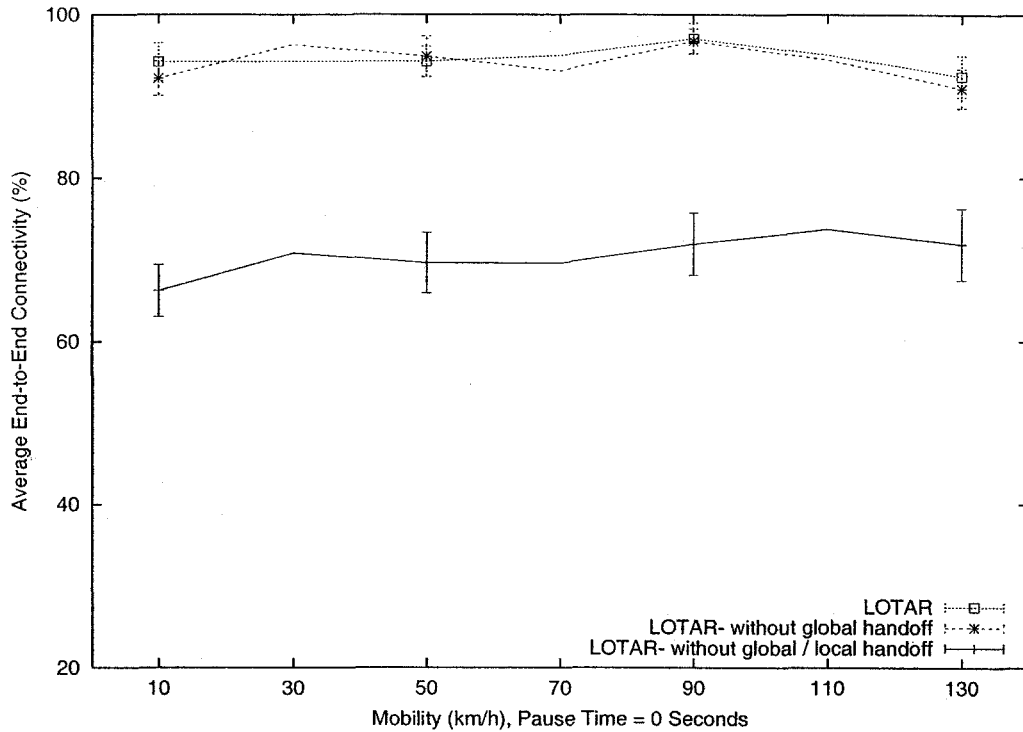


Figure 4.22: Average end-to-end connectivity

Fig. 4.23 shows the results of path length ratios. Very surprisingly, LOTAR and LOTAR1 have smaller path length ratios than LOTAR2. We carefully checked the results and found that local handoff improves the path on the whole. However, in prior experiments, LOTAR had a higher end-to-end delay than LAR. This is largely because local flow handoff contends for radio channels with data transmission and some flow handoff might not be successful due to packet losses. Fig. 4.23 illustrates the results under ideal conditions: local flow handoff is always successful, there is no data traffic, and flow handoff is almost on-time (because 50 ms sampling granularity is very small). The promising results present more opportunities to improve routing performance if we use different technologies such as separate signaling for flow handoff.

Again, LOTAR and LOTAR1 have very similar path length ratios. This demonstrates that global flow handoff has little influence in our simulated networks.

Fig. 4.24 shows the results for control overhead. Control overhead for LOTAR2 is significantly higher than that for LOTAR and LOTAR1. This is because LOTAR2 has no handoff mechanisms and lower end-to-end connectivity. Therefore, more route discoveries are initiated, resulting in high control overhead in LOTAR2. LOTAR has

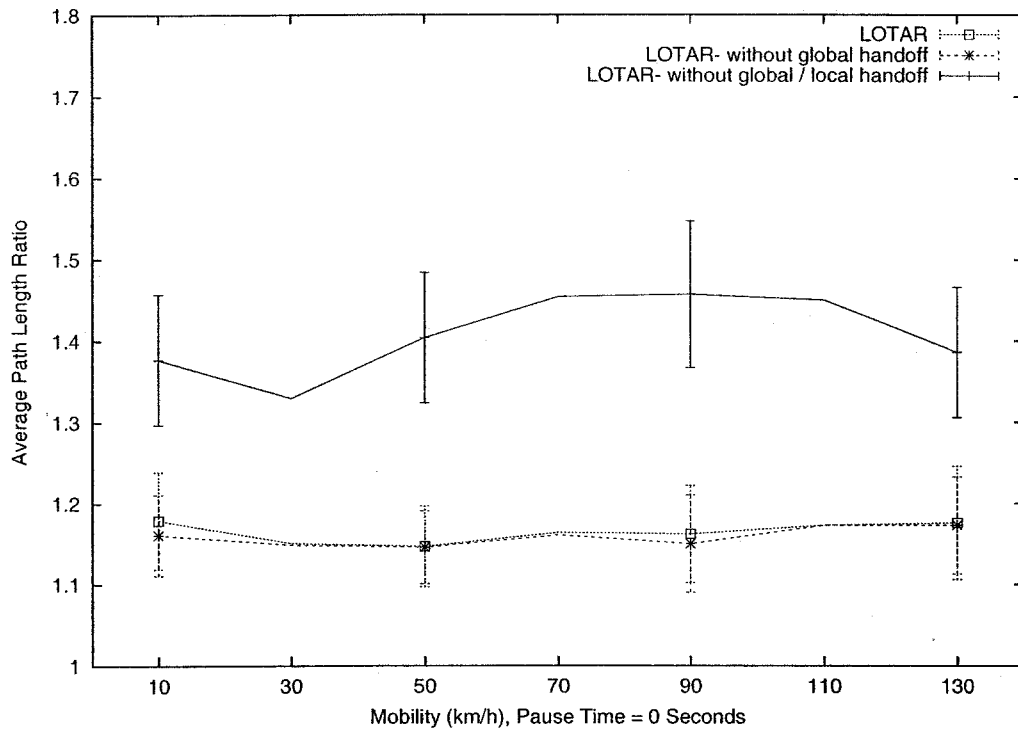


Figure 4.23: Path length ratio

slightly higher control overhead than LOTAR1 because LOTAR includes a global handoff mechanism which is demonstrated to be unnecessary if all local handoffs are successful and in time, as seen in the results in Fig. 4.22 and Fig. 4.23.

In conclusion, local flow handoff is effective enough to maintain good performance in terms of end-to-end connectivity, path length ratio, and control overhead in the simulated network. Global flow handoff seems to have little influence in the simulated network. The results shown in this section indicate the best performance of flow handoff.

## 4.7 Conclusions

Location information can be utilised to assist in routing [KV98, KK00, SL02, SG99]. In this chapter, we propose a routing scheme which uses location information to reduce route query flooding, search for efficient partial routes, and help flow handoff. This routing scheme should work with a concrete prediction model for a particular mobility model. We investigate its performance in various mobility models and its

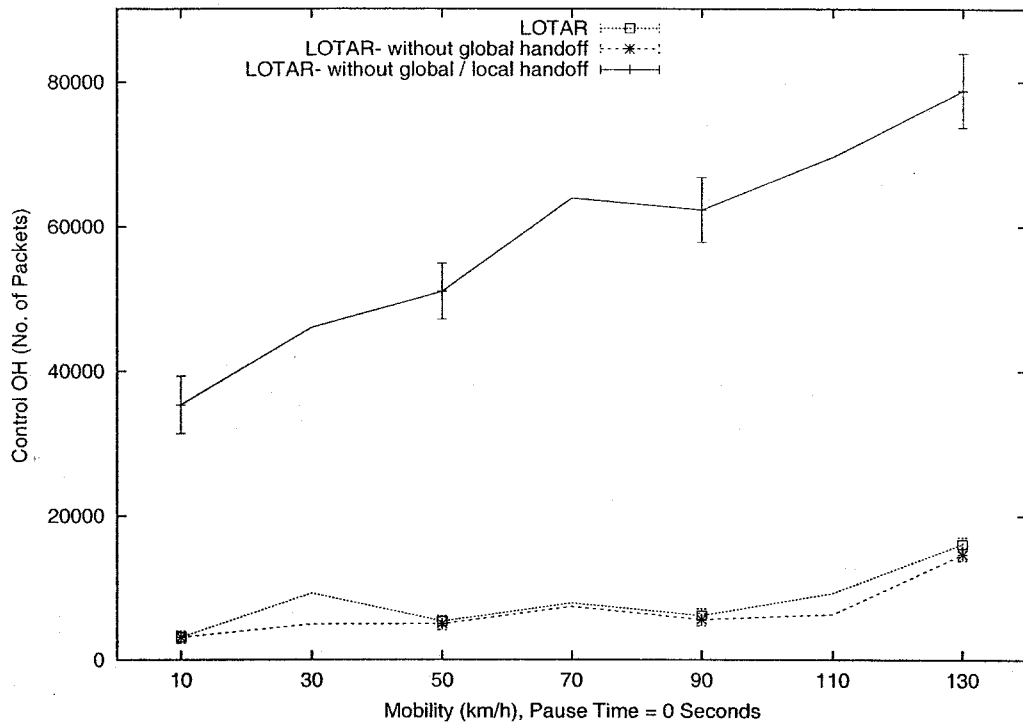


Figure 4.24: Average control overhead in packets

performance with inaccurate location information. The simulation study shows that our routing scheme can maintain very high packet delivery ratios even with high-speed mobile nodes. The average end-to-end delay and control overhead are also acceptable. Furthermore, we studied the benefits of flow handoff in an ideal situation. The results show that local flow handoff is effective enough to keep high end-to-end connectivity, small path length ratios, and small control overhead in the simulated networks. The good performance in the ideal situation provides us with great motivation to investigate efficient flow handoff techniques in mobile ad hoc networks.

# Chapter 5

## On-Demand Multipath Routing

### 5.1 Introduction

The absence of a fixed infrastructure forces the mobile hosts in MANETs to cooperate with each other for message transmissions. To form such a cooperative self-configurable environment, every mobile host is willing to relay messages for others to their ultimate destinations. Due to the limited power energy in every mobile host, it is very important to distribute routing tasks fairly.

However, few proposed routing protocols for MANETs take fairness into account. They tend to take the shortest path as the main route selection principle and therefore place a heavy burden on the hosts along the shortest path from a source to a destination. As a result, heavily loaded hosts may deplete power quickly, leading to network partitions and failure of application sessions. It is quite agonizing for end-users to experience rapid power depletion or to feel isolated from others in a cooperative environment.

Multipath routing is aimed at establishing multiple paths between a source-destination pair. In wired networks, multipath routing [ST92, GG96, Max93] can reduce end-to-end delay, increase network throughput, and provide better network load balancing. However, these advantages are not obvious in MANETs because it requires more hosts to be responsible for the routing tasks and the traffic flows along different paths may interfere with each other due to the broadcast feature of radio transmission. In addition, it is not easy to search effectively for multiple paths with little cost if network topology changes frequently.

As described in Chapter 2, proactive routing protocols try to maintain routes to

all possible destinations, regardless of whether or not the routes are needed. This category of routing protocols must periodically send control messages in order to maintain correct route information. In contrast, reactive (or on-demand) routing methods initiate the route discovery on demand of data traffic. Routes are needed only to desired destinations. This routing approach can dramatically reduce routing overhead when a network is relatively static and the active traffic is light. In this method, each node has little or no topology information. Without a complete and accurate knowledge of the topology, how to find node-disjoint or edge-disjoint multiple paths efficiently is difficult. All problems discussed in this chapter are based on the on-demand routing method.

We have not addressed the multipath routing problems with proactive methods. In proactive methods, a node periodically sends control messages to maintain update routing information even if the routes are not needed, wasting bandwidth when data traffic is light. However, proactive methods usually provide users with partial or complete topology information, making the calculation of node-disjoint multiple paths relatively easier. Nevertheless, proactive multipath routing methods deserve investigation if we can effectively control the routing overhead.

In this chapter, we introduce new criteria to measure the quality of multiple paths and present two on-demand approaches to effectively search for good multiple paths based on these criteria. We performed simulation studies to investigate network performance using the proposed methods. The motivation is to explore the benefits and present the difficulties of deploying on-demand multipath routing in shared-channel wireless mobile networks. The rest of this chapter is organized as follows. Section 5.2 introduces related work. In Section 5.3, we discuss the interference of traffic flows along different paths and its influence on the performance of multipath routing in MANETs. Section 5.4 introduces our on-demand multipath calculation algorithms. In Section 5.5, we introduce how to use the obtained multiple paths. Section 5.6 describes the simulation model. We present our performance results in Section 5.7 and conclude this chapter in Section 5.8.



## 5.2 Related Work

Multipath routing is not a new concept. Its use can be found in early circuit-switched and packet-switched networks [ST92, GG96, Max93]. It has been an effective mechanism in wired networks to balance network loads, increase throughput, and provide fault tolerance. However, the work on wired networks cannot be directly applied to MANETs because nearby links are no longer physically independent and the network topology can change frequently.

Some protocols for MANETs, such as Dynamic Source Routing (DSR) [BJM01] and the Temporally Ordered Routing Algorithm (TORA) [PC97], use multiple paths. In DSR, a source node acquires multiple paths to a destination by flooding route request messages, which are answered by the nodes that know how to reach the destination. As we will demonstrate later, this method has very little chance of finding disjoint multiple paths. TORA provides multiple paths by maintaining a “destination-oriented” directed acyclic graph from the source. However, TORA does not include criteria to evaluate the quality of the multiple paths. Furthermore, TORA did not perform well in the simulation studies in [BMJHJ98, DCYS98], partly because the benefits of multiple paths are greatly compromised by the large control overhead in maintaining multiple paths. On the whole, multipath routing is utilised as a backup or auxiliary method, and its advantages have not been fully explored in these particular protocols.

In [ND99], Nasipuri and Das proved that the use of multiple paths can keep correct end-to-end transmission longer than a single path. In other words, the frequency of searching for new routes is much lower if a node keeps multiple paths to a destination. To our knowledge, [ND99] is the first in-depth study on the performance benefits of multipath routing in MANETs. However, they did not measure the quality of the multiple paths and did not study the performance improvement using multipath routing in terms of other performance metrics such as network load balancing, end-to-end delay, and power consumption. Their performance study was based on theoretical analysis, in which it is difficult to consider the influence of nodes’ arbitrary movements and the interference of radio transmissions, both of which are distinguishing features of wireless mobile networks.

Perlman et al. [PHST00] demonstrate that multipath routing can balance network loads [PHST00]. They proposed a *diversity injection* method to find more node-disjoint paths compared to DSR, and studied its performance based on a hybrid proactive/reactive Zone Routing Protocol [HP00]. However, their work is based on multiple channel wireless networks, which are contention-free but may not be available in some application scenarios.

Almost simultaneously, Lee et al. [LG01] and Wu and Harms [WH01] separately proposed the same approach to search effectively for multiple node-disjoint paths. The basic idea is to selectively broadcast those route request messages that might include more disjoint information. As we will demonstrate in this chapter, this method is likely to have large control overhead. In this chapter, we introduce this method, and an improved method [WH01-2], to find node-disjoint multiple paths.

### 5.3 Multiple Path Selection Criteria

If we assume all mobile hosts' radio transmission ranges are the same, then a MANET can be modeled as an undirected graph  $G = (V, E)$ , where  $V$  is a set of  $|V|$  nodes and  $E$  is a set of  $|E|$  undirected links connecting nodes in  $V$ . Each node has a unique identifier and represents a mobile host with a wireless communication range of  $R$ . There is an undirected link  $(i, j)$  connecting two nodes  $i$  and  $j$  when the two nodes are within each other's transmission range.

Due to the broadcast feature of radio transmission, nearby radio transmissions may interfere with each other, and two nearby nodes must contend for a radio channel. We define a metric, correlation factor, to estimate the interference between two nearby traffic flows.

**Definition 1:** The correlation factor ( $\eta$ ) of two node-disjoint paths is defined as the number of links connecting the two paths. If there is no link ( $\eta=0$ ) between two node-disjoint paths, we say that the two node-disjoint paths are unrelated. Otherwise, the two node-disjoint paths are  $\eta$ -related. The total correlation factor for a set of multiple paths is defined as the sum of the correlation factor of each pair of paths.

As shown in Fig. 5.1(a) and (b), the source node  $S$  sends Constant Bit Rate (CBR)

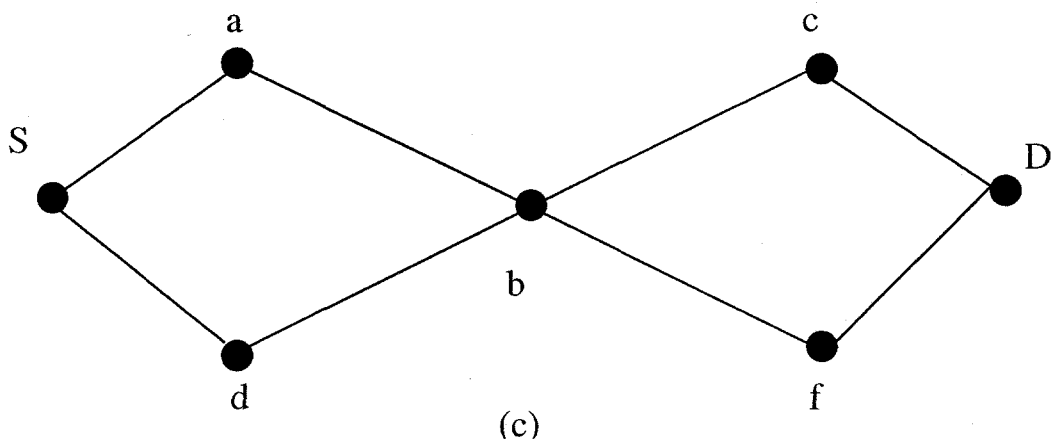
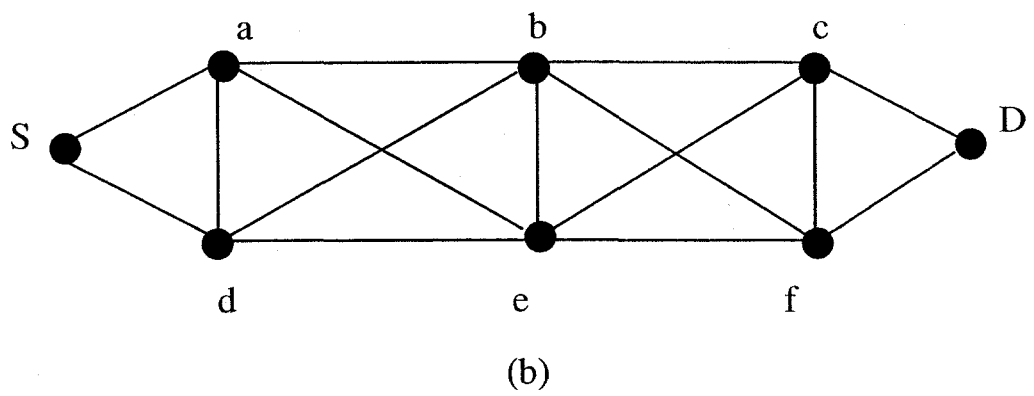
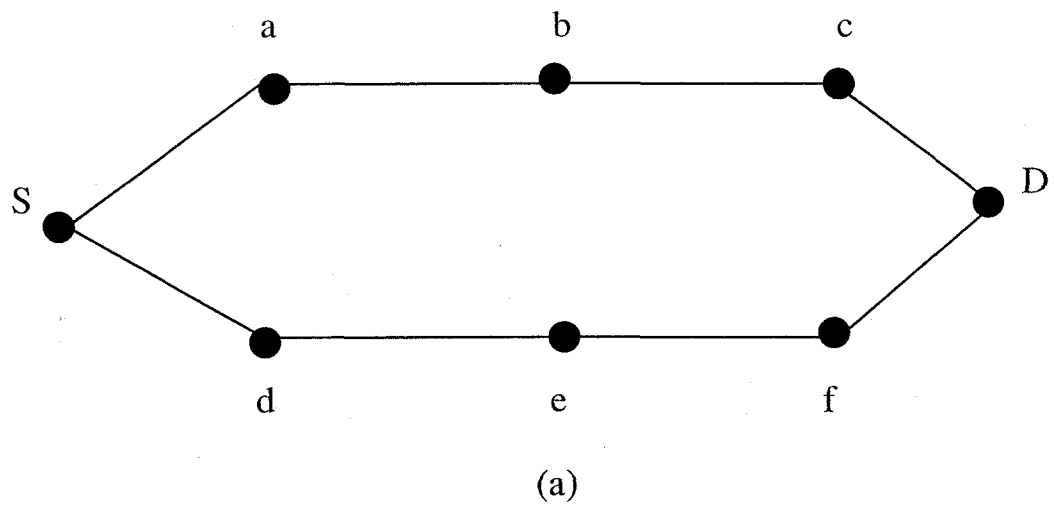


Figure 5.1: Different initial topologies

traffic to the destination node D using two node-disjoint paths between S and D, which are  $S \rightarrow a \rightarrow b \rightarrow c \rightarrow D$  and  $S \rightarrow d \rightarrow e \rightarrow f \rightarrow D$ . In Fig. 5.1(a), the two node-disjoint paths are unrelated. But in Fig. 5.1(b), the two node-disjoint paths are 7-related. In order to study the influence of the correlation factor, we initially selected static topologies. In our simulation, we changed the initial positions of the nodes to obtain different correlation factors for the two node-disjoint paths. We used a shared channel model in which all hosts use the same radio spectrum and compete for the radio channel. For example, in Fig. 5.1(b), if S is sending messages to a, then node d cannot send messages to node e since the transmissions will collide at node a.

Fig. 5.2 shows the result of the average end-to-end delay along these two node-disjoint paths for different correlation factors. The larger the correlation factor, the larger the average end-to-end delay for both paths. This is because two paths with a larger correlation factor have more chances of interfering with each other's transmissions due to the broadcast feature of radio propagation. In addition, with the increase in the correlation factor, the difference between the average end-to-end delay along the two paths is also increased even if the two paths have the same length and the same traffic load. The reason is that the traffic flows from node S to the different paths are not transmitted simultaneously. As a result, the data traffic sent a little bit ahead will have more chances of capturing the transmission channel, and the lagged data traffic will have to defer its transmission. The larger the correlation factor, the larger the influence on the difference in end-to-end delay.

In addition, we sent the same traffic load from S to D in the static topology shown in Fig. 5.1(c), using two link-disjoint but not node-disjoint paths,  $S \rightarrow a \rightarrow b \rightarrow c \rightarrow D$  and  $S \rightarrow d \rightarrow b \rightarrow f \rightarrow D$ . The average end-to-end delay was 21.3 ms along the path  $S \rightarrow a \rightarrow b \rightarrow c \rightarrow D$  and 30.5 ms along the path  $S \rightarrow d \rightarrow b \rightarrow f \rightarrow D$ . Compared with the results shown in Fig. 5.2, we see that the traffic flows along these two paths dramatically interfere with each other, since the flows along the different paths compete for the transmission channel in the common node b. Therefore, if the paths are link-disjoint but not node-disjoint, good performance cannot be guaranteed.

Path length is also an important factor in multipath routing. A longer path will increase the end-to-end delay and waste more bandwidth. It also requires routing services from more hosts. When we use multiple paths between a source-destination

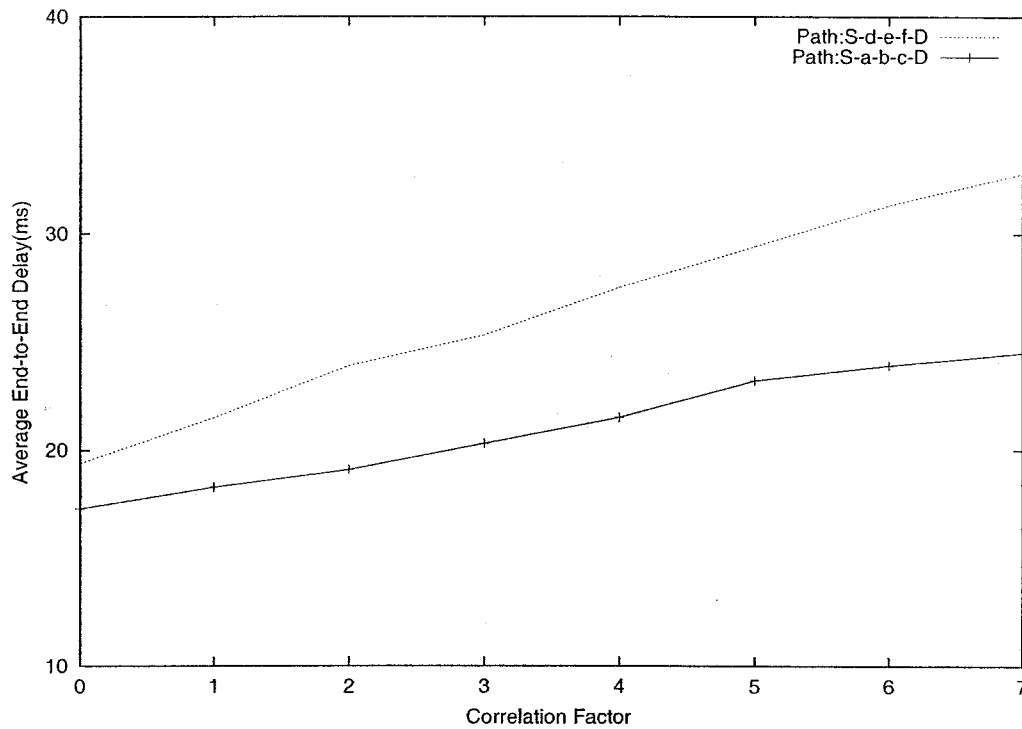


Figure 5.2: The impact of correlation factor on end-to-end delay in a static network

pair, the difference in the end-to-end delay among the multiple paths requires more buffer space in the destination to deal with the disordered data packets. Although this may not be a serious problem considering that the radio transmission rate is relatively slow compared with that of high-speed wireline networks, we should not ignore the influence of the differences in path length.

Based on the above observation, our path selection criteria in MANETs include properties of (1) node-disjoint, (2) small length differences between the primary (shortest) path and the alternative paths, and (3) small correlation factors between any two of the multiple paths. Since the path correlation factor is always changing when the nodes are moving, maintaining the property of small correlation factors will be costly. However, we can use this criterion at the initial path selection. We measure the quality of multiple paths in terms of the above criteria.

## 5.4 On-Demand Multipath Calculation

Finding node-disjoint multiple paths is not an easy task when the whole topology is unknown. The Dynamic Source Routing (DSR) [MBJJ99, BJM01] protocol finds multiple paths, but does not take the property of node-disjoint or link-disjoint into account. In this section, we first describe how DSR finds multiple paths and illustrate why the multiple paths obtained by DSR are not diverse. We then introduce three different on-demand multipath calculation approaches: the *diversity injection* method, the *selective broadcast* method, and the *heuristic redirection* method, all of which improve DSR in the ability of finding more node-disjoint paths.

### 5.4.1 Multipath Discovery in DSR

In DSR, if a source node does not know a route to a destination, it will initiate a route discovery by flooding a Route REQuest (RREQ) message. The RREQ message carries the sequence of hops it passed through in the message header. When a node receives an RREQ, if it is the first time that this node receives this RREQ message, the node will broadcast it again. Otherwise, the node will drop this RREQ packet. Once an RREQ message reaches the destination node, the destination node will reply with a Route REPLY (RREP) packet to the source, using the reverse path contained in the RREQ packet. When the RREP packet traverses backward to the source, the source and all traversed nodes will know a route to the destination. Since the destination replies to all the RREQ messages, multiple paths will be created between the source and the destination.

The method of broadcasting RREQ in DSR can effectively save bandwidth. However, it greatly reduces the possibility of finding multiple node-disjoint paths because it quenches the diversity of the multiple paths—the obtained multiple paths usually have some common nodes. In our simulation, the chance of finding node-disjoint multiple paths using DSR is almost zero. The reason is that later-received RREQ packets, which may include node-disjoint paths, are dropped at internal nodes. For example, in Fig. 5.1(b), if S broadcasts an RREQ, nodes a and d will receive and re-broadcast it. If we assume that node d transmits a little bit ahead of node a, nodes b and e will receive the RREQ packet from node d and drop the later RREQ

packet from node a. Finally, the destination D will receive only two paths having a common node d. We will explain this phenomenon further in Section 5.4.4.

### 5.4.2 The Diversity Injection Method

Perlman et al. [PHST00] proposed a new method, called *diversity injection* method, to improve DSR's ability to search for node-disjoint multiple paths. In the method, any internal node broadcasts only the first-received same route request messages as in DSR. However, instead of dropping all later-received route request messages as in DSR, the diversity injection method caches these messages. When a route reply message is received, the remaining path back to the source is replaced with "the shortest of the least selected cached paths that does not create a reply loop" [PHST00]. The simulation study shows that the diversity injection method increases the chance for the source node to obtain node-disjoint multiple paths [PHST00].

### 5.4.3 The Selective Broadcast Method

In [WH01], we propose a multipath calculation method based on the selective broadcast of route request messages. In this approach, we combine the path selection criteria in Section 5.3 with the path calculation method in DSR. When a route request message is initiated by a source node to search for paths to a destination, besides the information required for DSR such as the destination address and the sequence number, the route request message includes a parameter  $d$ , which indicates the permitted maximal length difference between the primary (shortest) path and the alternative paths. This parameter corresponds to path selection criterion (2) in Section 5.3.

As stated above, the broadcast mechanism for RREQ messages in DSR has little chance of finding multiple node-disjoint paths. Thus, we modify the route query flooding method as follows.

When a node receives a route query message, if it is the first time that it receives this query message or the path included in this query message is node-disjoint with the paths included in previously cached same route query messages, the node will cache it and broadcast it again. Otherwise, the node will discard this query message. In this way, any internal node caches and broadcasts all received node-disjoint paths. This method will increase the flooding cost, but it also increases the diversity of

the query messages. When the destination receives a query message, a route reply is sent back to the source if the query message carries a source route that is node-disjoint from the existing routes and the length difference between this route and the primary (shortest) route is less than  $d$ . The primary route is usually taken by the first query message arriving at the destination. The reply message carries the entire path information between the source and the destination.

The parameter  $d$  is used to limit the length difference between the shortest path and the alternative paths. A larger value of  $d$  may provide a source node with more node-disjoint paths to a destination, but a larger length difference among multiple paths requires a larger buffer in the destination to handle disordered packets. In addition, longer paths may waste more bandwidth and have a larger end-to-end delay.

Every node has a neighbourhood table to record its neighbours. The contents of the neighbourhood table are refreshed by any received control and data messages. If a neighbour has not been refreshed for a timeout value, it is deemed to be obsolete and is erased from the table.

When a reply message traverses from the destination back to the source node, it piggybacks the neighbourhood information along the path. The source node will calculate the path correlation factor using this neighbourhood information. The maximal correlation factor is used by the source to select the proper paths. The source will select the multiple paths whose total correlation factor is less than the maximal correlation factor.

In the worst case, the communication complexity of the *selective broadcast* method in terms of the number of message transmissions is  $O(L*N + L*W)$ , where  $L$  is the maximal number of node-disjoint paths from the source to the destination,  $N$  is the total number of mobile hosts, and  $W$  is the maximal permitted path length limited by the Time-To-Live (TTL) field in a packet. The first term,  $L*N$ , represents the maximal number of message transmissions for the route request broadcast. The second term,  $L*W$ , represents the maximal number of message transmissions for the RREP packets traversing back to the source node.



#### 5.4.4 The Heuristic Redirection Method

In order to reduce the control overhead of the selective broadcast method, we propose a heuristic method [WH01-2, WH02], in which an intermediate node does not re-broadcast the same RREQ message twice, but re-directs the RREP messages according to an heuristic algorithm.

**Definition 2** A *path*  $P$  between node  $v_1$  and node  $v_n$  is an ordered sequence of distinct nodes  $\langle v_1, \dots, v_n \rangle$  such that  $(v_{i-1}, v_i)$  is a link for all  $2 \leq i \leq n$ . Node  $v_i$  is called an *internal node* if  $1 < i < n$ . If no path exists between node  $v_1$  and node  $v_n$ , then node  $v_1$  and node  $v_n$  are *partitioned*.

**Definition 3** A *spanning tree* rooted at node  $S$  ( $S \in V$ ) in a graph  $G = \langle V, E \rangle$  is a subgraph of  $G$ , which is a tree with root  $S$  and includes all nodes in  $V$ . A subtree of a spanning tree is called a *primary subtree* if this subtree's root's direct father is the root of the spanning tree. The *primary subtree nodes* in a spanning tree with root  $S$  are a set of nodes whose direct father is node  $S$  in the spanning tree.

**Lemma 1** Using the route discovery method in DSR, if two paths  $P_1 = \langle S, v_1, \dots, v_n, D \rangle$  and  $P_2 = \langle S, u_1, \dots, u_m, D \rangle$  received by node  $D$  have  $k$  ( $k < m$  and  $k < n$ ) common nodes excluding node  $S$  and  $D$ , then  $v_1 = u_1, v_2 = u_2, \dots$ , and  $v_k = u_k$ .

*Proof:* Assume that the route discovery method of DSR is used. Suppose, to the contrary, that two paths  $P_1 = \langle S, v_1, \dots, v_n, D \rangle$  and  $P_2 = \langle S, u_1, \dots, u_m, D \rangle$  received by node  $D$  have  $k$  ( $k < m$  and  $k < n$ ) common nodes (excluding node  $S$  and  $D$ ), but there is an  $i \leq k$  such that  $v_i \neq u_i$ . Then there must be a  $j > k$  where  $v_j = u_j$ . Otherwise, the total number of common nodes could be less than  $k$ . Thus, node  $j$  will receive at least two RREQ packets: one was previously transmitted from  $v_i$  while the other was from  $u_i$ . However, according to the message broadcast method in DSR, node  $j$  will drop at least one of the packets because a node will not broadcast the same RREQ message twice. Therefore, only one path, either  $P_1$  or  $P_2$ , could arrive at node  $D$ . Node  $D$  cannot receive both  $P_1$  and  $P_2$ . This is a contradiction.  $\diamond$

From Lemma 1, two paths received by a destination are node-disjoint if and only if the first hops of the two paths are different. Furthermore, the union of all paths received by the destination will be a part of a spanning tree rooted at the source node

if we do not consider the last hop of these paths. This phenomenon provides us with useful information to search for diverse multiple paths. Note that Lemma 1 is true for any node that is reachable from the source.

**Lemma 2** For a spanning tree rooted at node S, if node  $v$  and node  $u$  belong to different primary subtrees and there is a direct link between node  $v$  and node  $u$ , then there are at least two node-disjoint paths between node  $v$  (or node  $u$ ) and node S.

*Proof:* The proof is straightforward. Assume that node  $v$  belongs to primary subtree  $B_1$  and node  $u$  belongs to primary subtree  $B_2$ . Then one path is from node  $v$  (or node  $u$ ) to node S along the tree links in  $B_1$  (or  $B_2$ ). The other path is from node  $v$  (or node  $u$ ) to node  $u$  (or node  $v$ ) and then to node S along the tree links in  $B_2$  (or  $B_1$ ). The two paths are node-disjoint since node  $u$  and node  $v$  belong to different primary subtrees of the spanning tree. Note that two different primary subtrees of a spanning tree have no common nodes.  $\diamond$

Lemma 1 and Lemma 2 are used to search heuristically for node-disjoint paths. In our method, the broadcast method of RREQ is very similar to that in DSR except that the later-received RREQ packets are cached instead of dropped in the intermediate nodes. An RREP packet in our method includes a label *isRedirection* to indicate whether the RREP packet should be redirected when traversing back to the source node. Due to Lemma 1, to check whether two paths received by the destination are node-disjoint, we only need to see whether their first hops are the same. The first hops of the paths included in the RREQ packets are also the primary subtree nodes in the spanning tree rooted at the source node. When the destination node D receives an RREQ packet, if the path  $P$  included in this RREQ packet is node-disjoint with all paths included in previously received RREQ packets, then an RREP packet is sent to node S, using the reverse path of  $P$ , and the label *isRedirection* is set to FALSE. Otherwise, an RREP packet is sent back with the label *isRedirection* set to TRUE.

When an intermediate node receives an RREP packet, it utilises the algorithm in Fig. 5.3 to check the next hop to forward the RREP packet. The main idea here is to redirect those RREP packets whose *isRedirection* label is TRUE. The reason is that an RREP packet with a TRUE *isRedirection* label includes a path having some common nodes with a previously sent RREP packet, which is labeled as *beforeRrep* for the convenience of later description. The intermediate node checks whether it has

```

if (the label isRedirection is set to FALSE in the RREP) or
  (there are no cached RREQ packets) {
  Forward the RREP packet to the prior hop along the path included in the RREP packet;
  Return.
}
else{
  Get S= {All cached RREQs that include paths whose first hop is different with the first hop of path P, where P is the
  reverse path included in the RREP};
  if (S is not empty) {
    repeat {
      Get the cached RREQ with shortest route in S. If several cached routes in S have same shortest length,
      then select one of them randomly;
      Replace the remaining forward path back to the source in the RREP packet with the path in the RREQ;
      if (the new RREP does not include a loop) {
        Forward the new RREP packet using the new path;
        Set the label isRedirection in the new RREP to FALSE;
        Return;}
      else Remove the RREQ from S;
    }
    until (all RREQ in S are checked or a proper RREQ in S, that can construct a new RREP without a loop, is
    searched )
  }
  /* In the following case, S is empty or S does not include proper RREQ packets to satisfy the above requirement */
  Get S2= {all cached RREQ - S};
  repeat{
    Search for a RREQ with shortest route in S2. If several cached routes have same shortest length, then select one
    of them randomly;
    Replace the remaining forward path back to the source in the RREP packet with the path in the RREQ;
    if (the new RREP does not include a loop) {
      Forward the new RREP packet using the new path;
      Return. }
    else Remove from the RREQ from S2;
  }
  until (a proper RREQ in S2, that can construct a new RREP without a loop, is searched)
  /* The proper RREQ should be in S2 since S2 includes a path, which is the same as the one in the RREP. */
}

```

Figure 5.3: The algorithm for forwarding the RREP packets

cached a path to the source node, which is node-disjoint with the remaining hops included in the RREP packet. Based on Lemma 2, redirecting the RREP packet to such a cached path will forward the RREP to a different primary subtree, where it is highly possible to find a path that is node-disjoint with the path included in the *beforeRrep*. Once the RREP packet is redirected, the label *isRedirection* in this packet is set to FALSE. Since the algorithm still uses source routing to forward the RREP, loops can be easily removed and the RREP packet will finally arrive at the source node.

In order to help the source nodes select good node-disjoint paths, we combine

the path selection criteria in Section 5.3 with the above path calculation method (Fig. 5.3). Every node has a neighbourhood table to record its neighbours. The maintenance of the neighbourhood table is the same as that in the selective broadcast method. When the reply message traverses from the destination to the source node, it piggybacks the neighbourhood information along the path. The source node will calculate the path correlation factor using the neighbourhood information. Two parameters are used in the selection of node-disjoint paths:  $d$ , which indicates the permitted maximal length difference between the primary (shortest) path and the alternative paths, and  $f$ , which is the permitted maximal total correlation factor among the selected multiple paths.

In the worst case, the communication complexity of the *heuristic redirection* method in terms of the number of message transmissions is  $O(N + M * W)$ , where  $N$  is the total number of mobile hosts,  $M$  is the number of paths received by the destination limited by the number of the destination's neighbouring nodes, and  $W$  is the maximum permitted path length limited by the Time-To-Live (TTL) field in a packet. The first term,  $N$ , represents the number of message transmissions for one global flooding. The second term,  $M * W$ , represents the maximal number of message transmissions for all RREP packets traversing back to the source node.

#### 5.4.5 The Ability to Find Multiple Node-Disjoint Paths

Because some route request and route reply messages may be lost due to the unreliability of radio transmission, our multipath calculation methods cannot guarantee finding all node-disjoint paths. However, we demonstrate through simulation that our methods can find most node-disjoint paths. We compared the ability to find multiple node-disjoint paths using the diversity injection method [PHST00], DSR, and our methods separately. The main difference between the diversity injection method and the heuristic redirection method is that the diversity injection method uses a different criterion for redirecting the RREP packets. As we will show below, the diversity injection method may redirect the RREP packets in a way that reduces the number of node-disjoint multiple paths found.

We randomly chose 200 source-destination pairs and calculated the maximal number of node-disjointed paths between each source-destination pair, using an off-line

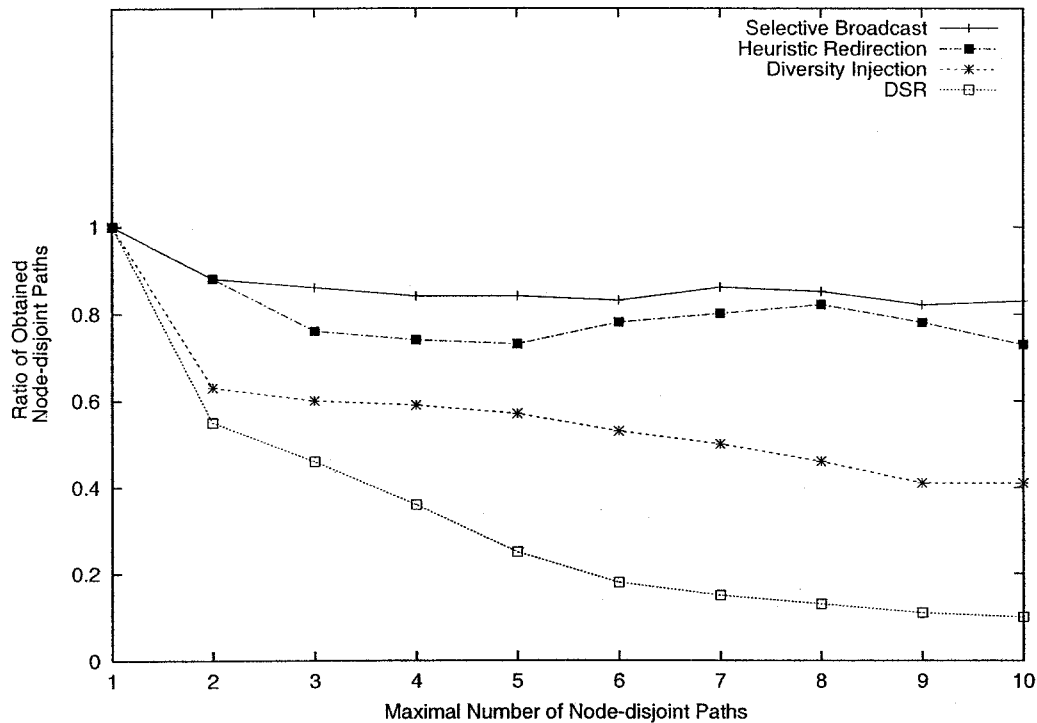


Figure 5.4: The ability to find multiple node-disjoint paths

algorithm with a knowledge of the entire topology. In the meantime, we separately used the diversity injection method, DSR, and our methods to search for node-disjoint paths. Fig. 5.4 shows the ratio of the number of obtained node-disjoint paths using different searching methods to the maximal number of node-disjoint paths using the off-line algorithm. The result shows that both the selective broadcast method and the heuristic redirection method can find most node-disjoint paths and more than the diversity injection method, while DSR has little success in finding node-disjoint paths.

Note that the above results are obtained in a static network since the network usually keeps relatively unchanged during the short time period of routing discovery. In the situation that the network topology changes dramatically even during one routing discovery period, all routing protocols may not work effectively, and the differences among the three methods will be trivial.

## 5.5 Multipath Routing

There are several ways to use the obtained multiple paths. In [ND99] and [BJM01], the multiple paths are not used simultaneously. The data packets are transmitted along one path. Other paths are kept as backup paths in case the used one fails. When all possible paths are broken, a new multipath discovery procedure is initiated.

Our approach of using the multiple paths is different. In order to balance network loads, we use the multiple paths simultaneously as in dispersity routing [Max93], which disperses the data traffic along different paths. Dispersity routing can be divided into redundant and non-redundant routing. In redundant dispersity routing, only some of the multiple paths are used to transfer data, while the others are used to transfer redundant information such as error-correcting codes. In contrast, in non-redundant dispersity routing, all paths are used to transmit data simultaneously. We use non-redundant dispersity routing and choose a path with a probability inversely proportional to the length of the path. If a path fails, an error message is sent back to the source node and the traffic on that path will be transferred to other paths that are still alive. When all paths are broken, a new multiple path discovery is initiated.

## 5.6 Simulation Model

We used the same simulation model described in Chapter 3 to study the performance of multipath and unipath routing. The multipath routing methods include our proposed methods and the diversity injection method. Note that only the method of searching for node-disjoint paths (diversity injection) from [PHST00] is used. The underlying radio channel model and the way to use the multiple paths are different from those that are used in [PHST00]. The unipath routing method studied here is a simplified DSR, which does not include such optimisations as promiscuous learning of source routes, leveraging the route cache, piggybacking on route discoveries, etc. [BJM01]. The reason for excluding these mechanisms from the simulation is that our main focus is on the performance difference when dispatching the traffic along single path versus multiple paths.

In order to investigate whether or not multipath routing can benefit energy con-

sumption, we used the Agilent WaveLan [Luc00] model to calculate power consumption. The power consumption in doze, receive, and transmit modes is approximately 50 mw, 900 mw, and 1425 mw respectively. We assume that a mobile host will be in doze mode immediately after transmitting or receiving a packet and can be revoked from the doze mode immediately before transmitting or receiving a packet. We also assume that a mobile host passively receives any heard packets even if they are not for the mobile host. It is the network layer that decides how to process (drop/relay/accept) the packets.

In the simulation, 50 mobile nodes moved in a 1500 metre x 500 metre rectangular region for 900 seconds of simulation time. Initial locations of the nodes were obtained using a uniform distribution. We assume that each node moves independently with the same average speed. All nodes had the same transmission range of 250 metres. The mobility model is the Random Waypoint model (Chapter 3). In the simulation, the minimal speed was 5 m/s and the maximal speed was 10 m/s. We changed the pause time from 0 seconds to 900 seconds to investigate the performance influence of different mobilities. A pause time of 0 seconds presents continuous motion, and a pause time of 900 seconds corresponds to no motion.

The simulated traffic was Constant Bit Rate (CBR) traffic. 15 source nodes and 15 destination nodes were chosen randomly with uniform probabilities. The interval time to send packets was 250 ms. The size of all data packets was set to 512 bytes. The maximal total correlation factor was set to 15. The maximal number of multiple paths was 4. The maximal length difference between the shortest path and the alternative paths was 3, and the maximal path length was 9.

## 5.7 Simulation Results

### 5.7.1 Performance Metrics

We compared the performance of unipath routing with the various multipath routing methods. We evaluated the performance according to the following metrics:

- *Control overhead*: The control overhead is defined as the total number of routing control packets normalized by the total number of received data packets.

- *Bandwidth cost for data:* The bandwidth cost for data is defined as the total number of data packets transmitted at all mobile hosts normalized by the total number of received data packets.
- *Average end-to-end delay:* The end-to-end-delay is averaged over all surviving data packets from the sources to the destinations. It includes queuing delay and propagation delay.
- *Load balancing:* We use a graph  $G=(V, E)$  to represent the network, where  $V$  is the node set and  $E$  is the link set. We define a state function  $f : V \rightarrow I$  where  $I$  is the set of positive integers.  $f(v)$  represents the number of data packets forwarded at node  $v$ . Let  $\text{CoV}(f) = \text{standard deviation of } f / \text{mean of } f$ . We use  $\text{CoV}(f)$  as a metric to evaluate the load balancing. The CoV can be used as a measure of the variance of a certain metric and it is to be preferred to standard deviation because the CoV is a unit-less quantity. The smaller the  $\text{CoV}(f)$ , the better the load balancing.
- *Energy balancing:* As above, we use  $\text{CoV}(g)$  to evaluate the energy balancing, where  $g(v)$  represents the energy consumption at each node. The energy consumption in a node is calculated as the sum of  $T_i * P_i$  ( $i = 0, 1, 2$ ), where  $T_i$  represents the time spent in the three different modes (doze, receive, and transmit) and  $P_i$  represents the power consumption in the corresponding modes.
- *Average energy consumption:* The energy consumption is averaged over all nodes in the network.

### 5.7.2 Comparison of Multipath Routing Performance with Unipath Routing Performance

Fig. 5.5 shows how the total number of route discovery phases varies with the mobility. The frequency of route discovery for multipath routing is less than that for the unipath routing. This result is consistent with the theoretical analysis in [ND99]. The frequency of route discoveries for our multipath routing methods (selective broadcast and heuristic redirection) is less than that for the diversity injection method since our methods find more node-disjoint paths, as shown in Fig. 5.4.



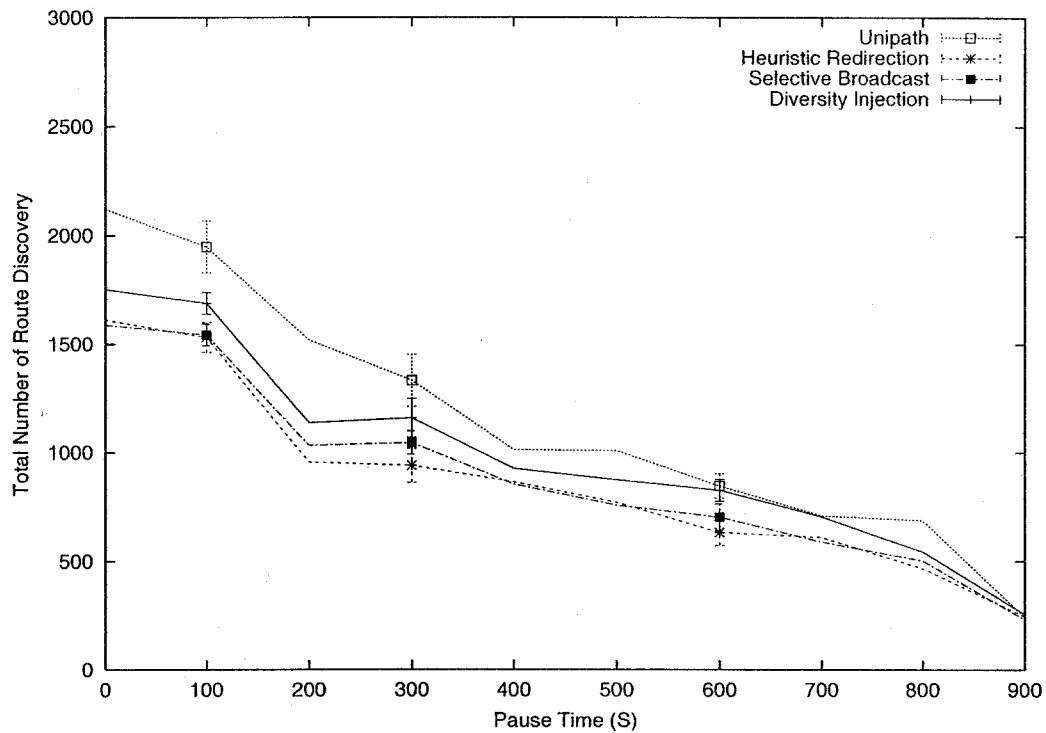


Figure 5.5: The frequency of route discovery for different methods

Because every route discovery needs approximately the same control overhead in the heuristic redirection, the diversity injection, and the unipath routing methods, the reduction of route discovery frequency will reduce the total control overhead. This is the reason that the heuristic redirection method has the smallest control overhead, as shown in Fig. 5.6. However, the selective broadcast method exhibits the highest control overhead because each route discovery requires more control message transmissions, resulting in the highest total control overhead even if it has fewer route discoveries than unipath routing.

Fig. 5.7 shows the total bandwidth cost for data transmission. This tends to be the smallest for unipath routing because unipath routing uses the shortest path from a source to a destination. The alternative paths in multipath routing are usually sub-optimal and thus cost more bandwidth. An interesting phenomenon is that the bandwidth cost for data transmission in the diversity injection method tends to be higher than in our methods at low speed. However, the variability in the results indicates that these differences may not be significant.

Fig. 5.8 shows the average end-to-end delay. The end-to-end delay includes the

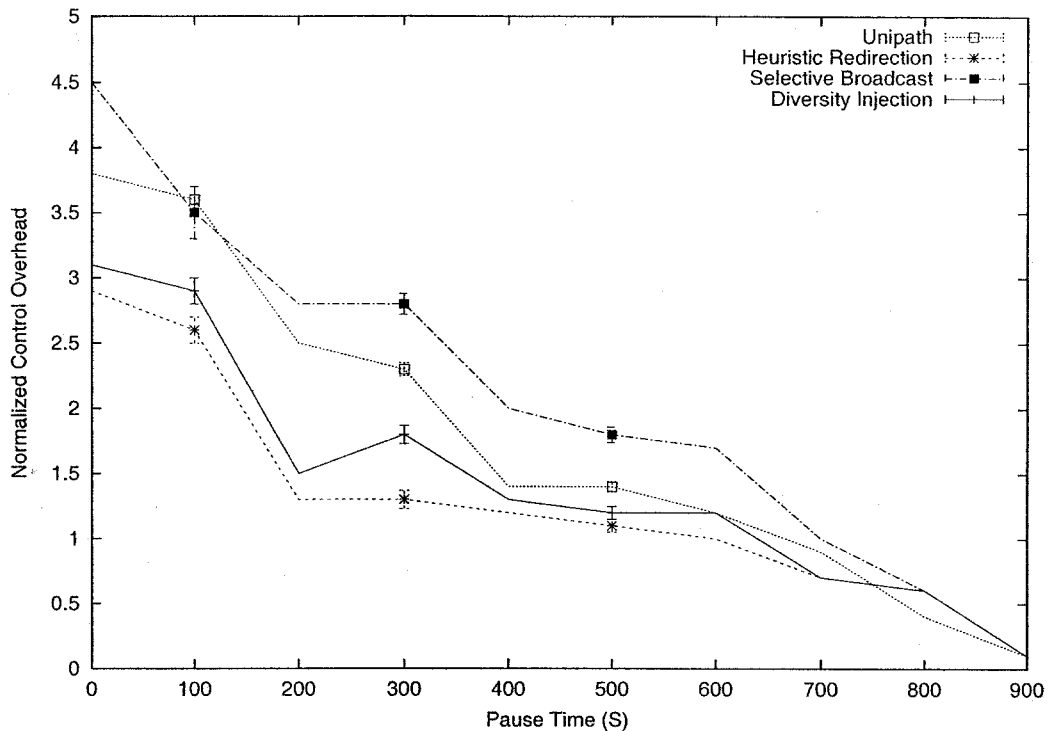


Figure 5.6: Normalized control overhead for different methods

queuing delay in every host and the propagation delay from the source to the destination. Multipath routing will reduce the queuing delay because the traffic is distributed along different paths. On the other hand, it will increase the propagation delay because some data packets may be forwarded along sub-optimal paths. The unipath routing method has higher average end-to-end delay than our multipath routing methods, demonstrating that our multipath routing methods can distribute network loads and reduce the end-to-end delay, but the improvement may be limited at a high mobility (when pause time is below 300 seconds in the simulation). With a decrease in pause time, the average end-to-end delay for both multipath routing and unipath routing increases because the network topology changes more frequently. More route discoveries will be initiated, and thus the queuing delay of the data packets in the source nodes increases, resulting in an increase in the average end-to-end delay.

At low speed, however, the diversity injection method shows a larger end-to-end delay than the unipath routing method. As described above, the diversity injection method uses the shortest cached route that has been used the least number of times to re-direct RREP packets. This does not mean that the diversity injection method

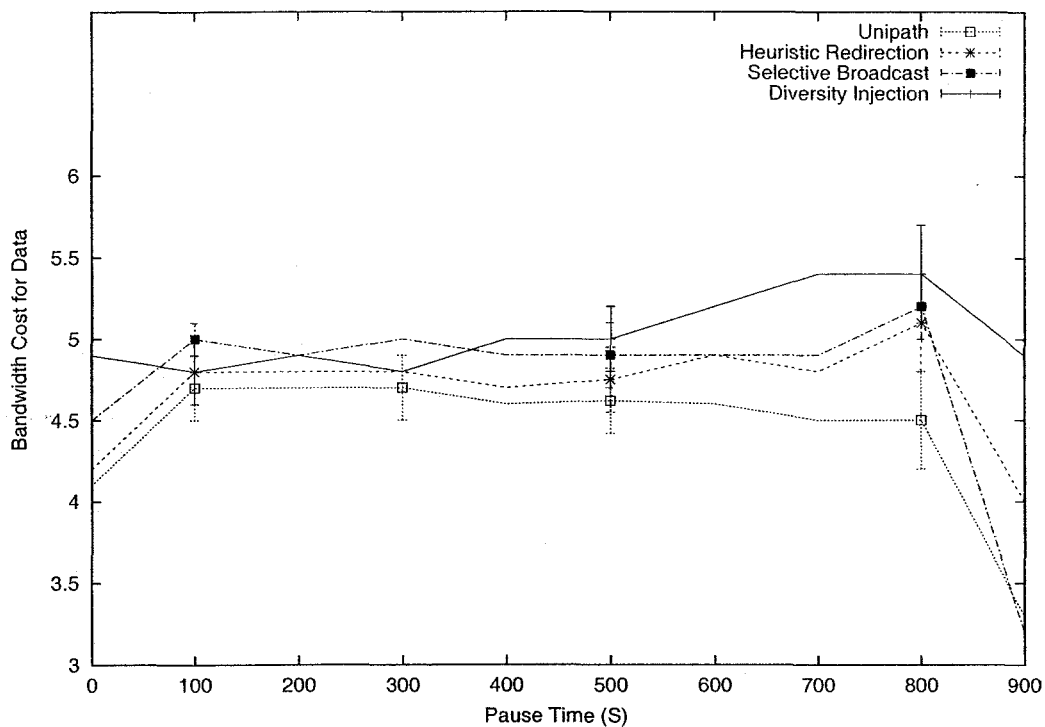


Figure 5.7: Bandwidth cost for data for different methods

will ultimately obtain shorter node-disjoint paths than our methods. The possible multiple redirecting in the diversity injection method may result in longer paths. In contrast, the heuristic redirection method re-directs an RREP packet only once when a proper direction is found; and the selective broadcast method provides the most possible node-disjoint paths to the destination. Thus no re-direction of RREP packets is needed. By tracing the simulation, we observed that those RREP packets that happen to be re-directed to longer paths are likelier to include node-disjoint paths in the diversity injection method. In other words, the diversity injection method finds fewer and longer node-disjoint paths. The improvement in end-to-end delay due to dispersing traffic cannot compensate for the degradation of end-to-end delay when using longer paths, demonstrating that multipath routing does not improve end-to-end delay in all scenarios. This is an important lesson for deploying multipath routing in MANETs.

Fig. 5.9 illustrates the results relating to load balancing. The CoV of network load is the highest for the unipath routing method because unipath routing always uses the shortest paths between the sources and the destinations. This has the effect

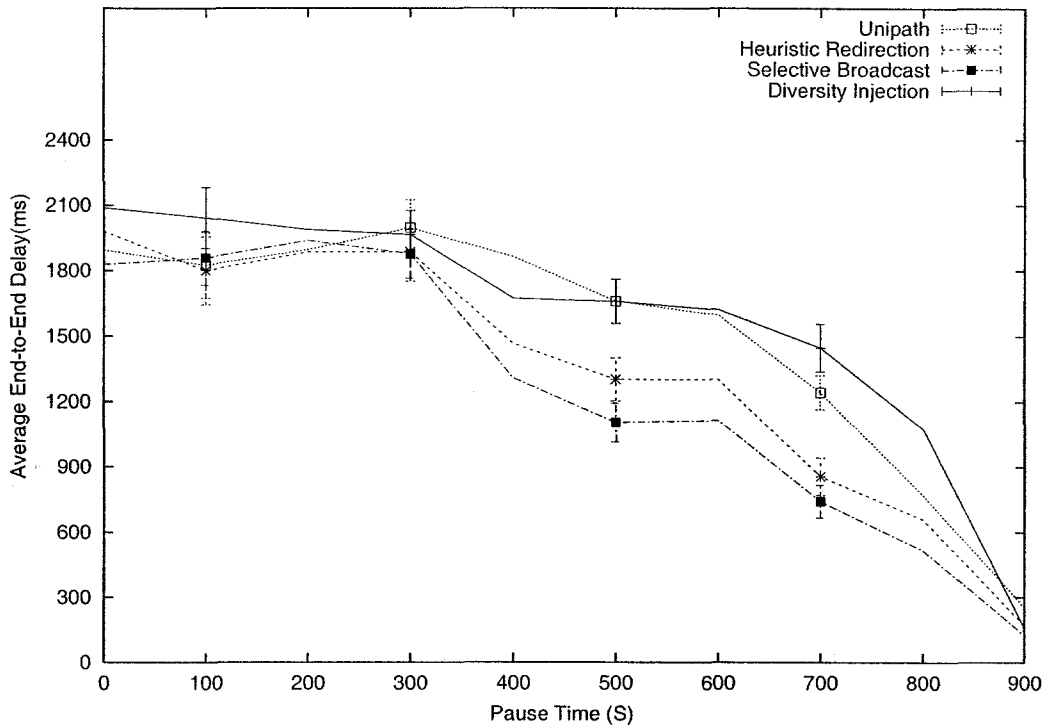


Figure 5.8: Average end-to-end delay for different methods

of assigning more duties to the nodes along the shortest paths. On the other hand, multipath routing methods can distribute the network traffic along different paths. As pause time decreases, the CoV of network load also decreases for the unipath routing method and the multipath routing methods, demonstrating that an increase in mobility can result in better load balancing. This suggests that “Hot spots” are likely to be removed as mobility increases. Our methods are better than the diversity injection method in terms of load balancing because they can find more and shorter node-disjoint paths.

Based on the NCR WaveLan model for energy consumption, Fig. 5.10 shows the results of energy balancing. The CoV of energy consumption for the unipath routing method and the diversity injection method is higher than that for our methods, demonstrating that our methods can assign the routing tasks more fairly. However, note that the scale of the y-axis is much smaller than the one in Fig. 5.9. The improvement in energy balancing is modest. This is because the nodes, even when they have no routing tasks, have to passively listen to neighbouring nodes’ radio transmission, which inevitably consumes battery energy.

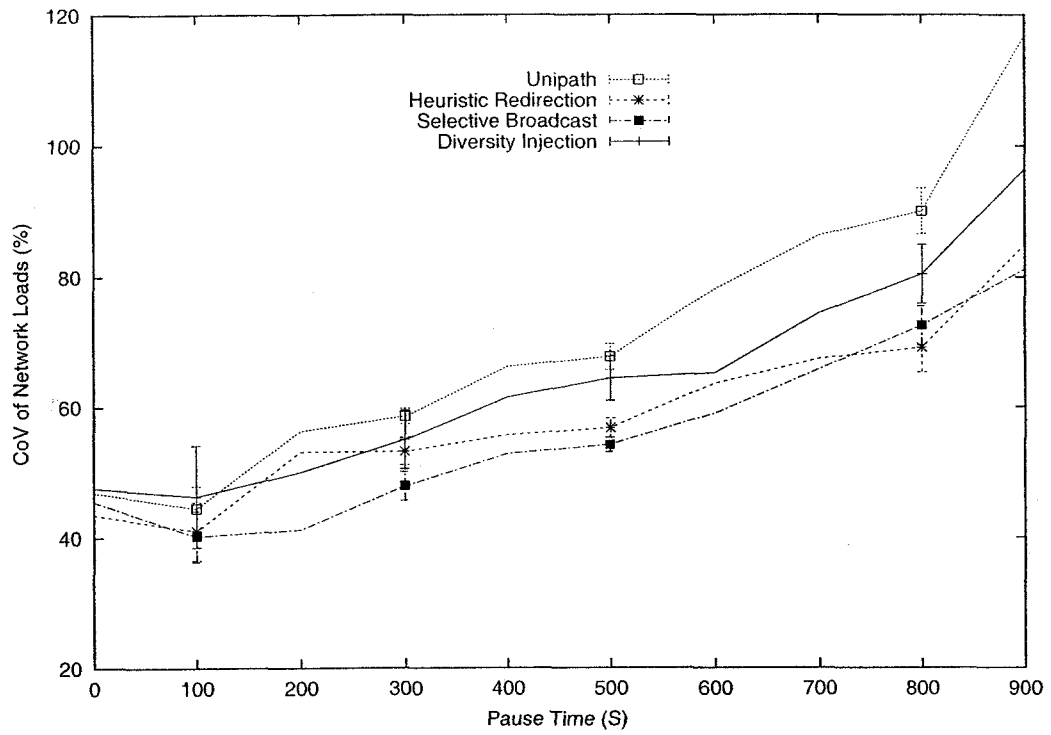


Figure 5.9: The CoV of network load for different methods

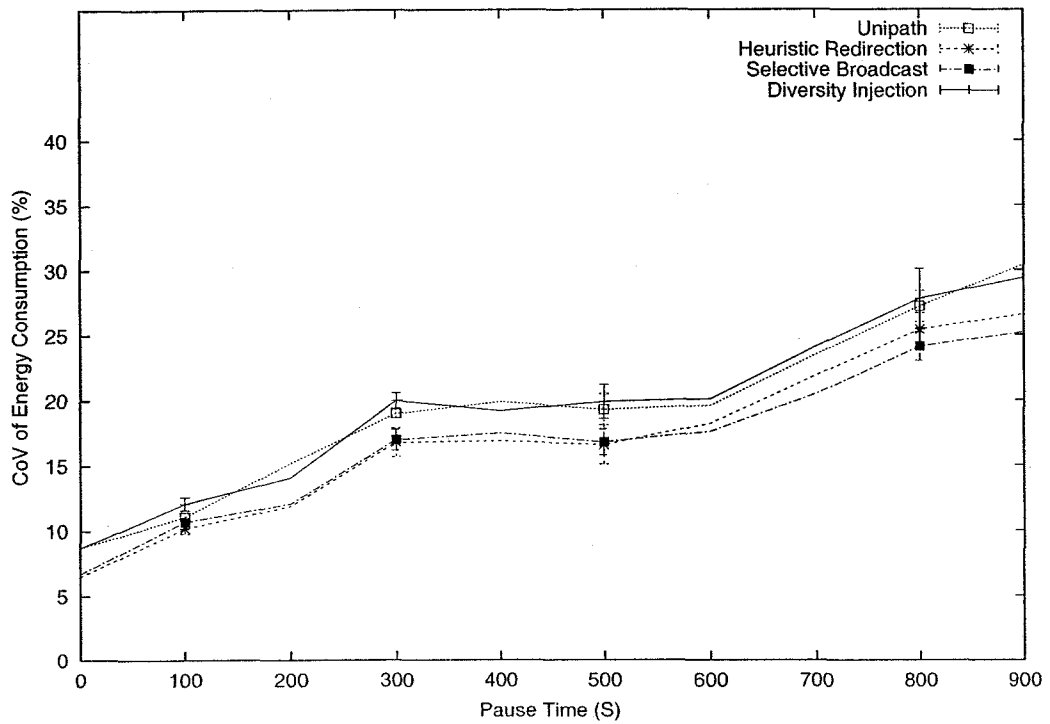


Figure 5.10: The CoV of energy consumption for different methods

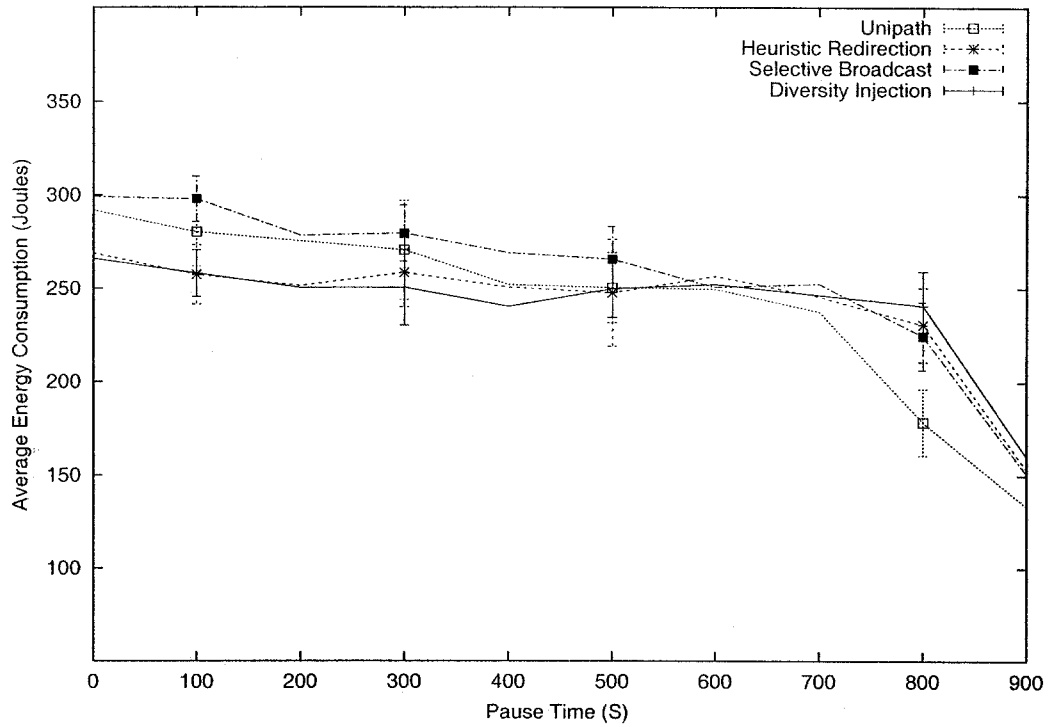


Figure 5.11: Average energy consumption for different methods

Fig. 5.11 shows that the heuristic redirection method and the diversity injection method have less average energy consumption than unipath routing when mobile speed is high (the *pause time* is less than 400 seconds in the simulation). At low speed, however, the average energy consumption in the unipath routing method is less than that in the multipath routing methods. The battery energy of a network node is consumed mainly through forwarding control packets and data packets. Multipath routing may increase the energy consumption due to the transmission of data messages because some data packets traverse sub-optimal paths, but it will decrease the energy consumption due to the transmission of control messages if proper strategies are adopted as in the heuristic redirection method. When mobile speed is high, the energy expensed on routing control is too large to compensate for the energy saved on data transmission along the optimal paths in unipath routing. This is why energy consumption is higher in unipath routing when mobile speed is high. Nevertheless, the selective broadcast method exhibits consistently high energy consumption because of its high control overhead.

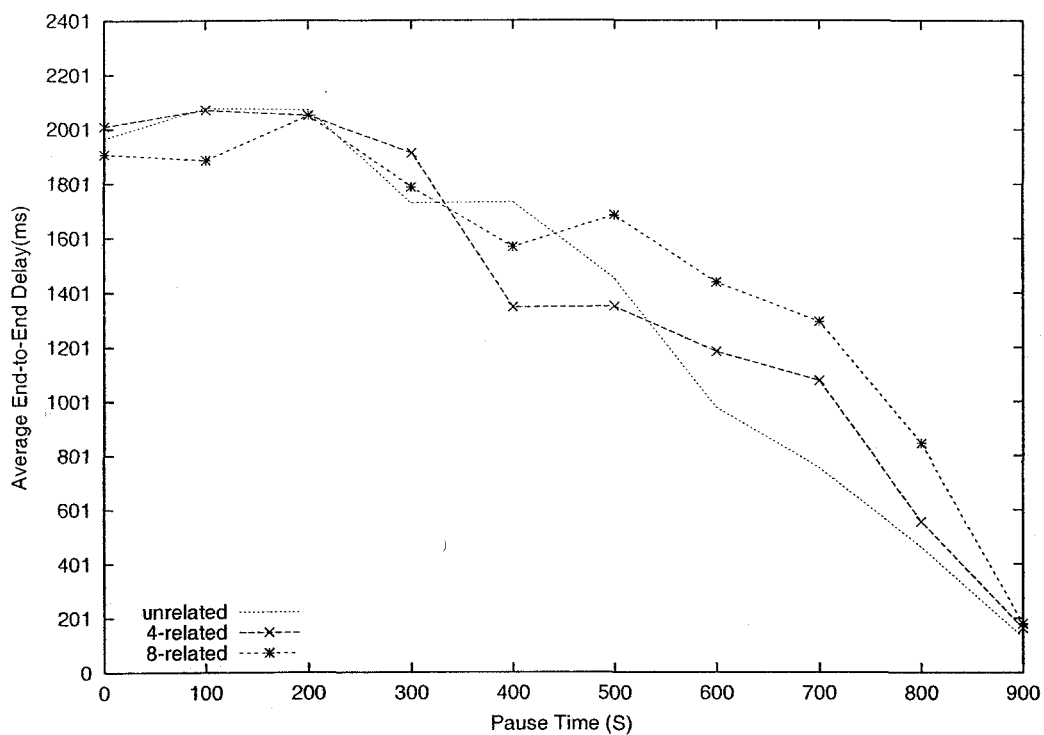


Figure 5.12: End-to-end delay with different correlation factors

### 5.7.3 Comparison of Multipath Routing Performance With Various Correlation Factors

In this section, we discuss how the initial selection of the multiple paths with different correlation factors can influence routing performance. In this experiment, we utilised the selective broadcast multipath routing approach and selected two paths. The simulation results show that when mobile speed is fast, selecting multiple paths with different correlation factors has little influence on average end-to-end delay. This is because the correlation factors will change quickly with the nodes' mobility. However, as shown in Fig. 5.12, when the mobile speed is slow (the pause time is longer than 600 seconds in the simulation), selecting multiple paths with smaller correlation factors has smaller average end-to-end delay.

Selecting multiple paths with various correlation factors does not influence the routing performance in terms of control overhead, bandwidth cost for data transmission, and load balancing. For example, the CoV of network loads with different correlation factors shows no significant differences (Fig. 5.13).

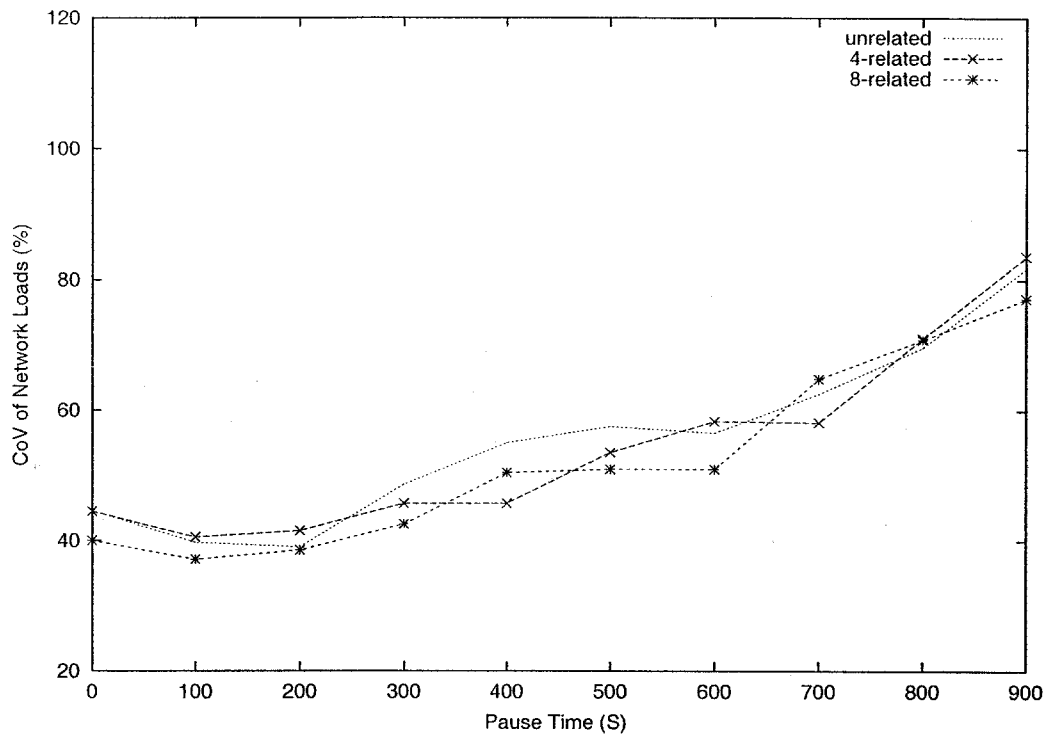


Figure 5.13: The CoV of network loads with different correlation factors

## 5.8 Conclusions

In a shared channel wireless mobile network, it is not easy to exploit the advantages of multipath routing due to the transmission interference and lack of topology information. In this chapter, multipath selection criteria were introduced, two new on-demand multipath calculation methods were proposed, and the performance of different multipath routing strategies was studied. Simulation results show that

- Multipath routing methods can reduce the frequency of route discovery, but the control overhead must be carefully controlled when deploying multipath routing.
- Multipath routing methods can provide some improvement in end-to-end delay in a shared channel MANET if suitable short multiple node-disjoint paths are utilised.
- Network load can be distributed more evenly in multipath routing. This feature is important to distribute the routing tasks fairly among mobile nodes and can



prevent node from failure since a node with heavy use is likely to deplete its energy quickly. Mobility can also contribute to effective network load balancing.

- The heuristic redirection multipath routing method can provide fair energy consumption among network nodes and reduce total energy consumption when mobile speed is high.
- The initial selection of the multiple paths with different correlation factors can influence the average end-to-end delay when mobile speed is low, but it has little effect on other performance metrics.

# Chapter 6

## Load-Sensitive Routing

### 6.1 Introduction

Distributing routing tasks fairly has eminent advantages, such as reducing the possibility of power depletion and queuing delay in hosts with heavy load. In Chapter 5, we investigated the benefit of multipath routing in terms of load balancing. In a very dynamic environment, however, obtaining and maintaining good quality multiple paths are still challenging problems. In this chapter, we propose another way to avoid loading mobile nodes heavily.

Although many routing protocols have been proposed for MANETs, few of them take load information into account. Pearlman et al. [PHST00] show that the multipath routing approach can balance network loads, but they do not use network load information in their method. In [DPR00], Das et al. conclude that using congestion-related metrics such as queue length can improve the routing performance in MANETs, but they do not point out how to utilise this information. In [HZ01], Hassanein and Zhou utilise load information as the main path selection criterion for routing in MANETs. They define the network load in a node as the total number of routes passing through the node and its neighbouring nodes. However, this load metric may not be accurate since the traffic along different paths may not be the same. In their method, each node exchanges load information with its neighbours periodically, and each intermediate node broadcasts all received route request messages. In [LG01-2], Lee and Gerla propose a load-aware routing method. They define the network load of a node as the number of packets queued in the output interface of the node. They utilise a route adaptation strategy to search for better paths.

Similar to the methods in [HZ01, LG01-2], our proposed Load-Sensitive Routing (LSR) protocol [WH01-3] also utilises network load information as the main path selection criterion. In LSR, the network load in a node includes not only the local load, but also the load in the node’s neighbouring nodes. This metric is more accurate than the one used in [LG01-2] due to the broadcast feature of radio propagation, which creates contention. Obtaining network load information in LSR does not require periodic exchanges of load information among neighbouring nodes and is suitable for any existing routing protocol. We also provide a unified scheme to utilise load information and to tune the performance of LSR. Unlike the methods in [HZ01] and [LG01-2], LSR does not require that destination nodes wait for all possible routes. Instead, LSR uses a re-direction method to effectively find better paths (similar to our method in Section 5.4.4). This non-blocking property makes source nodes respond quickly to a call for connection without losing the chance to obtain the best path. Based on the initial status of an active path, LSR can search for better paths dynamically if the active path becomes congested during data transmission.

The rest of the chapter is organised as follows: in Section 6.2, we introduce the Load-Sensitive Routing (LSR) method. Section 6.3 describes the simulation model. We present the performance results of LSR in Section 6.5, and conclude this chapter in Section 6.6.

## 6.2 Load-Sensitive Routing (LSR)

### 6.2.1 Background

In Chapter 5, we demonstrated that traffic flows along nearby paths can interfere with each other and that the interference influences the routing performance. Due to the broadcast feature of radio transmission, a node is affected by the loads of its neighbours. Therefore, the network load in a mobile host depends on the traffic passing through this host as well as the traffic in the neighbouring hosts.

**Definition 1:** Traffic load in a mobile host is defined as the sum of the number of packets queued in the output interface of the mobile host and the number of packets queued in the output interfaces of the host’s neighbouring hosts.

A host can easily obtain the traffic load information if this host listens promis-

Table 6.1: Packet header for LSR

Field	Description
LL	the number of packet queued in the current node
PL	total path load so far
PV	standard deviation of path load so far

Table 6.2: Load information maintained by LSR

Information	Description
TL	sum of all LL(i)
ID(i)	the node id of the i-th neighbour
LL(i)	local load of the i-th neighbour

cuously to neighbouring nodes' transmissions. In our Load-Sensitive Routing (LSR) method, every packet transmitted includes the Local Load ( $LL$ ) of the node and path load information, which includes total Path Load ( $PL$ ) and the standard deviation of path load ( $PV$ ) for the path so far. Table 6.1 shows the information included in the packet header. The local load ( $LL$ ) is the number of packets that are queued in the current host. As shown in Table 6.2, every host maintains a value of Traffic Load ( $TL$ ). The value of the  $TL$  in a node is calculated as the sum of the  $LL$  of this node and the newest  $LLs$  received from neighbouring nodes. Obsolete  $LL$  values will be removed after a timeout period. The total Path Load ( $PL$ ) is calculated as the total sum of  $TL$  in all hosts along the path. The standard deviation of path load, which is denoted as  $PV$ , is defined as the standard deviation of  $TL$  in every host along the path. The total Path Load ( $PL$ ) represents the total traffic along a path and nearby nodes, while the standard deviation of path load ( $PV$ ) represents whether the traffic is distributed evenly.

**Remark 1** For a given path  $P = \langle v_1, \dots, v_n \rangle$ , if we assume that  $PL(P)$  denotes the total path load of  $P$  and  $TL(v_i)$  denotes the traffic load in node  $v_i$  ( $1 \leq i \leq n$ ), then

$$PL(P) = \sum_{i=1}^n TL(v_i) \quad (1)$$

**Remark 2** For a given path  $P = \langle v_1, \dots, v_n \rangle$ , if we assume that  $PV(P)$  denotes the standard deviation of traffic load in the nodes along the path  $P$  and  $TL(v_i)$  denotes

the traffic load in node  $v_i$  ( $1 \leq i \leq n$ ), then

$$PV(P) = \sqrt{\frac{\sum_{i=1}^n (TL(v_i))^2 - n * (ML)^2}{n - 1}}$$

$$\text{where } ML = \frac{PL(P)}{n} \quad (2)$$

**Remark 3** Using Formula (2), we can calculate  $PV(P)$  incrementally from  $PL$  and the hop count so far, and we do not need to keep every value of  $TL(v_i)$  in the packet header. For example, when  $v_n$  receives a packet from  $v_{n-1}$ , if we denote  $P_1 = \langle v_1, \dots, v_{n-1} \rangle$ , then this packet should carry the values of  $PL(P_1)$  and  $PV(P_1)$  as described above. Let  $P = \langle v_1, \dots, v_n \rangle$ , then  $PL(P) = PL(P_1) + TL(v_n)$ . From Formula (2), we can solve the equation to obtain  $\sum_{i=1}^{n-1} (TL(v_i))^2$  first, then calculate  $\sum_{i=1}^n (TL(v_i))^2$ , and finally get  $PV(P)$ . The detailed steps are as follows.

Step 1: Let

$$Y_1 = \sum_{i=1}^{n-1} (TL(v_i))^2$$

$$ML_1 = \frac{PL(P_1)}{n - 1}$$

Solve the following equation to get  $Y_1$ :

$$PV(P_1) = \sqrt{\frac{Y_1 - (n - 1) * (ML_1)^2}{n - 2}}$$

Note that  $PV(P_1)$  and  $PL(P_1)$  are carried by the received packet (see Table 6.1).

Step 2: Calculate  $PV(P)$ .

$$Y = Y_1 + (TL(v_n))^2$$

$$ML = \frac{PL(P_1) + TL(v_n)}{n}$$

$$PV(P) = \sqrt{\frac{Y - n * (ML)^2}{n - 1}}$$

A path comparison function is used to select a better path.

**Definition 2:** A path comparison function  $f(A, B)$ , where  $A, B$  are two paths, is defined as:

$$f(A, B) = \begin{cases} -1 & \text{if } (PL(A) - PL(B) > \alpha) \vee \\ & ((|PL(A) - PL(B)| \leq \alpha) \wedge (PV(A) - PV(B) > \beta)), \\ 0 & \text{if } (|PL(A) - PL(B)| \leq \alpha) \wedge (|PV(A) - PV(B)| \leq \beta), \\ 1 & \text{otherwise.} \end{cases}$$

In the definition of  $f$ , we use the total path load ( $PL$ ) as the main comparison criterion and the standard deviation of path load ( $PV$ ) as an auxiliary criterion. If the total path load of path  $A$  is at least  $\alpha$  larger than that of path  $B$ , or the difference in total path load of path  $A$  and  $B$  is not larger than  $\alpha$  but the standard deviation of the path load of path  $A$  is at least  $\beta$  larger than  $B$ , then we think that path  $B$  is better than path  $A$ .  $f(A, B) = -1$  represents this case. When  $f(A, B) = 0$ , we consider that path  $B$  and path  $A$  are almost the same in terms of path load information.  $f(A, B) = 1$  indicates that path  $A$  is better than path  $B$ . The two parameters  $\alpha$  and  $\beta$  are used to make the comparison flexible.

Based on different requirements, we use two sets of path comparison functions: *strict* and *loose*. In the strict set, the parameters  $\alpha$  and  $\beta$  are set very small, while in the loose set,  $\alpha$  and  $\beta$  are relatively larger.

## 6.2.2 Overview of LSR

### Route Discovery

Like DSR [MBJJ99, BJM01], LSR is an on-demand routing protocol. Unlike [HZ01] and [LG01-2], we use a re-direction method similar to the one that we developed in the *heuristic redirection* method (Chapter 5) to forward Route REPLY (RREP) messages. This method enables the source node to obtain better paths without an increase in flooding cost and waiting delay in the destination node.

In LSR, if a source node does not know a route to a destination, it will initiate a route discovery by flooding a Route REQuest (RREQ) message. The RREQ message carries the source and destination addresses, a sequence number which is initialized to 0 and increased by 1 for the next route discovery, the sequence of hops which it passed through, and the local load and path load information. The path load information

includes two values,  $PL$  and  $PV$ , and is calculated according to Formulas (1) and (2). When a node receives an RREQ, if it is the first time that the node receives this RREQ message, this node will broadcast it again. Otherwise, the node will cache this RREQ packet but not broadcast it. Two RREQ packets are assumed to be the same if they include the same source and destination addresses and the same sequence number.

Once an RREQ message reaches the destination node, the destination node will reply with a Route REPLY (RREP) packet to the source, using the reverse path contained in the RREQ packet. Since the destination does not wait for all possible routes, the source node can quickly obtain the route information and can quickly respond to calls for connections. Similar to DSR, LSR lets the destination reply to every received RREQ message in order to provide tolerance for broken links on the return trip and more routing choices for the source node. The RREP packet will recalculate the path load information when it traverses back to the source node. When a node receives an RREP packet, the remaining path in the RREP to the source node is replaced with the cached RREQ that has the best path load information and does not form a loop. The best path is chosen using a strict path comparison function  $f$ . If several best paths exist, one of them is chosen at random.

When the source node first receives an RREP packet, it uses the path included in this RREP packet to forward packets immediately. In the meantime, the source node will record the path load information included in this RREP packet. If a later-received RREP packet provides better path load information, then the source node transfers the traffic to the newly obtained path and updates the path load information accordingly.

In the route discovery phase, we use a strict path comparison function to compare the load information of two paths. Since the parameters  $\alpha$  and  $\beta$  are set to small values in a strict path comparison function, small differences between two paths in terms of path load and standard deviation of path load can distinguish the paths, and thus a better path can be chosen easily.

## Route Adaptation and Maintenance

In the LOTAR protocol discussed in Chapter 4, we dynamically select a more robust path before the used one breaks. The criterion of route adaptation in LOTAR is based on link lifetime. In LSR, we adapt the active routes in a different context by using network load information. When a used path becomes congested, LSR tries to search for a lightly loaded path. Note that the source node continues to send data traffic along the congested path until a better path is found. In [LG01-2], Lee and Gerla also utilise a route adaptation method. The criterion for route adaptation in [LG01-2] is based on fixed threshold values. Instead, our route adaptation method is based on the initial status and current status of an active path.

When a data packet is sent from the source to the destination, the current path load is calculated. The initial path load information obtained by the first data packet along a selected path will be recorded in the destination node. When the current path load information is much worse than the initial path load information, network congestion may have occurred and a new better path should be found. In order to check whether the current path has become much worse, a loose path comparison function is used. The parameters  $\alpha$  and  $\beta$ , in this case, are threshold values for the needs of route adaptation. As we will show later, if the parameters  $\alpha$  and  $\beta$  are too small, too much unnecessary route adaptation will be initiated.

If the current path becomes much worse than its initial status, which is calculated based on the first data packet, the destination node will initiate a route adaptation. It broadcasts a Route REQuest message to search for a route to the source node. This RREQ message will calculate the path load, using the same method as in the route discovery stage. It also carries the path load information obtained from the latest data packet received at the destination. Once the source node receives an RREQ, if the path load calculated by this RREQ is much better than the path load of the currently-used path, which is included in the RREQ, the source node transfers its data traffic to the path included in this RREQ packet. Note that the source node does not send back an RREP packet, and the source node uses the same loose path comparison function to check whether the path received in the RREQ packet is much better than the current one.



When an established path breaks due to mobility, an error message is sent back to the source node and the source node activates a route discovery, as described above.

Since a strict path comparison function is used in the route discovery phase while a loose one is used in the route adaptation phase, for the convenience of later discussion and without loss of generality, we call the path comparison function used in the route discovery phase a strict path comparison function and the one used in the route adaptation phase a loose path comparison function even if we may deliberately set its parameters to small values for the purpose of a simulation study.

### 6.3 Simulation Model

We used the same simulation model described in Chapter 3 to study the performance of LSR. We compared the performance of LSR and Dynamic Source Routing (DSR), which does not take network load information into account. We studied how the path comparison functions influence protocol performance. We also evaluated the performance of a variant of LSR, called LSR-2, whose promiscuous listening mode is disabled. Like [LG01-2], LSR-2 defines the network load in a node as the total number of packets queued in the node's output interface only, because a node cannot know its neighbouring nodes' local load information if the promiscuous listening mode is disabled.

In the simulation, 50 mobile nodes moved in a 1500 metre x 500 metre rectangular region for 900 seconds of simulation time. All nodes had the same transmission range of 250 metres and each node moved independently with the same average speed. The mobility model was the Random Waypoint model (Chapter 3). The minimal speed was 1 m/s and the maximal speed was 20 m/s. The simulated traffic was Constant Bit Rate (CBR). The size of all data packets was 512 bytes. We select a larger buffer size to permit a node to have a relatively heavy load without buffer overflow. In this simulation, the buffer size was 128 packets.

We evaluated the performance in terms of packet delivery ratio, average end-to-end delay, and normalized control head (See Chapter 3).

## 6.4 The Influence of Path Comparison Functions

The path comparison functions in LSR provide us with a unified way to select paths using different load information. Two metrics, total path load and standard deviation of path load, are used in the path comparison functions. Since a strict path comparison function is used in the route discovery phase, setting the parameters in the strict path comparison function to large values makes no sense (otherwise, the path selection method will be the same as in DSR). Therefore, we mainly consider how to set the parameters in the loose path comparison function for route adaptation.

We studied several schemes to use load information. These schemes include (1) using the total path load information only, (2) using the standard deviation of path load information only, (3) adapting routes frequently, (4) disabling route adaptation, and (5) properly adapting routes with the help of both total path load and standard deviation of path load information.

As shown in Table 6.3, five groups of values were studied. In Group A, the path comparison functions select a better path based only on the total path load information since  $\beta$  is set to a very large number and plays almost no role in the path comparison. In Group B, the path comparison functions select a better path based only on the standard deviation of path load since  $\alpha$  is very large. In Group C, the parameters in the loose path comparison function are set very small. In this case, a small difference in the traffic load information may result in route adaptation. The parameters in the loose path comparison function are set to very large values in Group D. In this case, LSR may not detect congestion during data transmission. In other words, many fewer route adaptations will be activated. In Group E, the parameters  $\alpha$  and  $\beta$  in the strict path comparison function are set to small values, but they are set to relatively larger values in the loose path comparison function.

In the simulated traffic model, 15 source-destination pairs sent CBR traffic with an interval time of 250 ms, and 10 randomly chosen nodes periodically broadcast background traffic to their neighbouring nodes with an interval time uniformly distributed from 50 ms to 100 ms.

Fig. 6.1 shows the results for packet delivery ratios. Group C has the lowest packet delivery ratio. In Group C, too much route adaptation will be aroused unnecessarily.

Table 6.3: Path comparison functions with different parameters

Group	Strict $f$		Loose $f$	
	$\alpha$	$\beta$	$\alpha$	$\beta$
A	5	3000	200	3000
B	3000	2	3000	10
C	5	2	5	2
D	5	2	3000	3000
E	5	2	200	10

A large amount of control traffic will cause network congestion and result in more packet losses. As shown in Fig. 6.1, although Group E has the highest packet delivery ratio, the performance difference among Groups A, B, D, and E is not significant. The packet delivery ratio for Group B is lower than that for Groups A, D, and E, although the difference may not be significant. This indicates that the total path load information is more important than the standard deviation of path load information, since the role of total path load is almost disabled in Group B due to the fact that a large number is set for the parameter  $\alpha$  in both the strict and loose path comparison functions. A path with a large standard deviation in load may be a good path. For example, there may exist a situation in which most nodes along a path are lightly-weighted with only a very small number of the nodes having heavy loads. This is the main reason that we define the total path load information as the main path selection criterion in the path comparison function.

Fig. 6.2 shows the results for average end-to-end delay. As expected, Group C has the largest average end-to-end delay. In Group C, unnecessary route adaptation may switch the traffic to longer paths and more control packets will contend for the transmission channel. At relatively lower speeds (pause times longer than 300 seconds in the simulation), Group A presents average end-to-end delays similar to those in Group D and E and has better performance than Group B, demonstrating that the average end-to-end delay depends mainly on the total path load.

Fig. 6.3 shows the results for control overhead. Group C has the highest control overhead because it has many route adaptation operations. The control overhead of Groups A, B, and E is not significantly different, although Group B has the highest control overhead among them. This is because the numbers of route adaptations

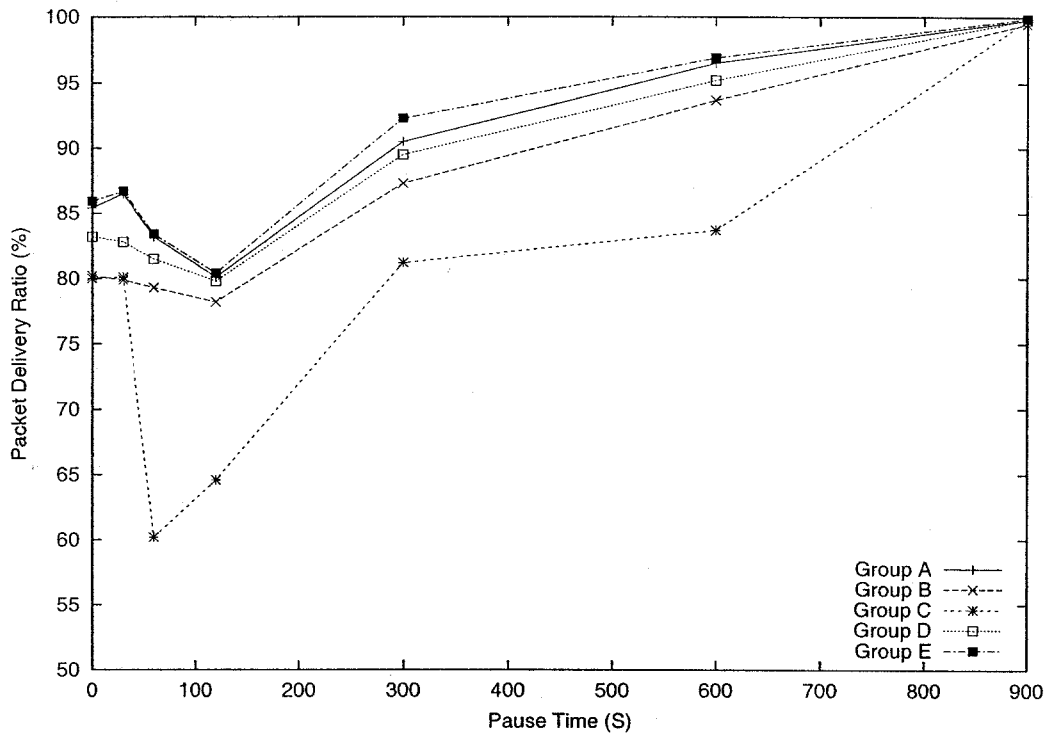


Figure 6.1: Packet delivery ratio with different parameters in  $f$

are not significantly different among these scenarios, although they activate route adaptation according to different criteria. Group D has the smallest control overhead since it has very few route adaptations.

Based on the above observations, two main principles can be applied to setting the parameters in the path comparison functions. First, the parameters  $\alpha$  and  $\beta$  in the strict path comparison function should always be set to small values, but they should not be set to small values in the loose path comparison function. Second, choosing different values can adjust the path selection criterion. For example, setting a very large value for  $\alpha$  in the strict and loose path comparison functions will disable the role of total path load information, and setting a very large value for  $\beta$  in the strict and loose path comparison functions will disable the role of path load deviation information.

Another observation is that the packet delivery ratio and the average end-to-end delay of Group E (with route adaptation) are better than those of Group D (with very little route adaptation), but the control overhead of Group E is worse than that of group D at high mobility. However, these differences are not great with our simulated

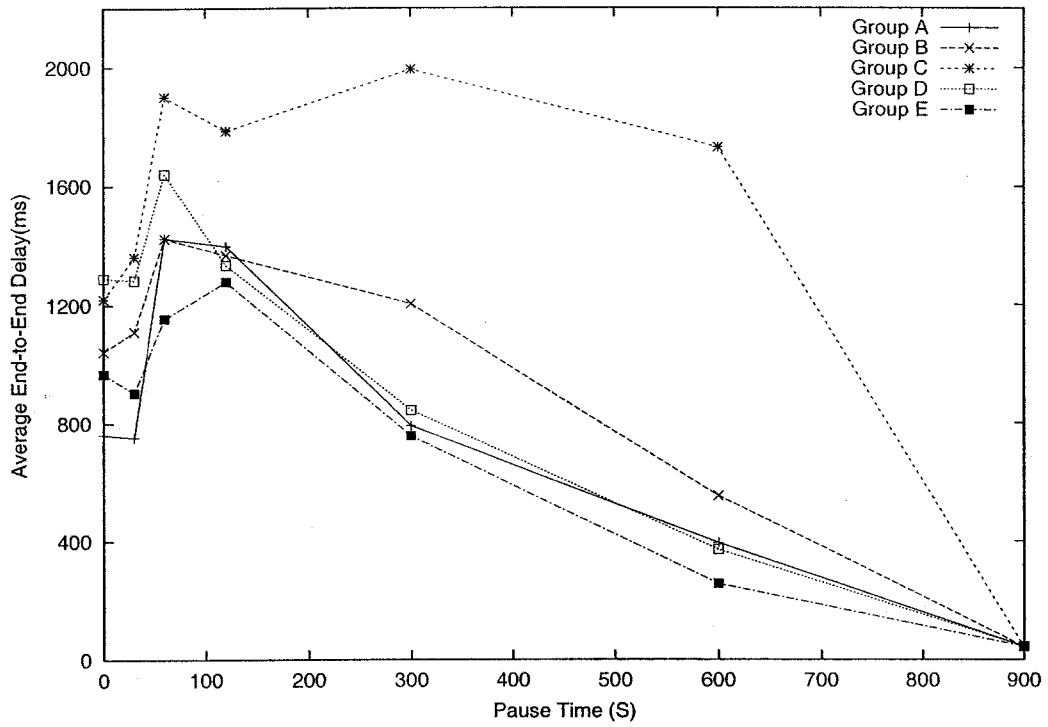


Figure 6.2: Average end-to-end delay with different parameters in  $f$

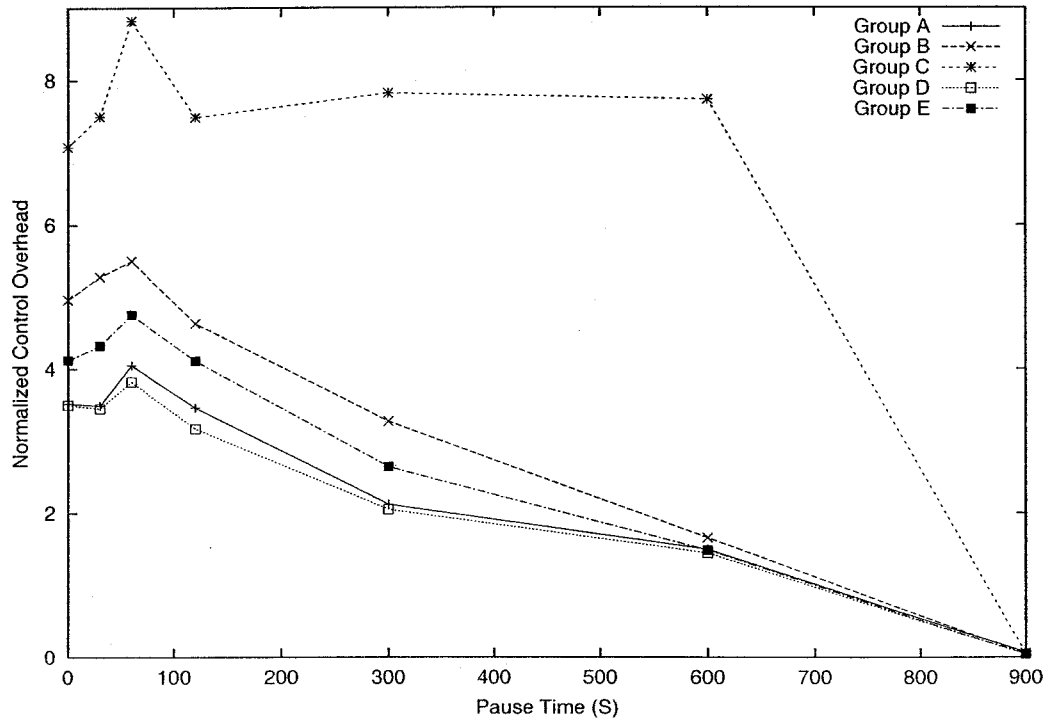


Figure 6.3: Control Overhead with different parameters in  $f$

traffic.

## 6.5 Simulation Results

As described above, the parameters  $\alpha$  and  $\beta$  in the strict path comparison function should be set to small values, but they should not be set to small values in the loose path comparison function. According to this principle, we set the parameters  $\alpha$  and  $\beta$  to 5 and 2 respectively in the strict path comparison function and to 200 and 10 respectively in the loose path comparison function. Three types of traffic loads were simulated. In the first scenario, 10 source-destination pairs sent CBR traffic with an interval time of 200 ms, and 10 randomly chosen nodes periodically broadcast background traffic to their neighbouring nodes with an interval time uniformly distributed from 50 ms to 100 ms. In this scenario, the connections had a medium traffic load, but the randomly chosen nodes had a heavy traffic burden and were likely to become “hot spots” in the network. In the second scenario, 10 source-destination pairs sent CBR traffic with an interval time of 250 ms. The traffic was light in this case. In the third scenario, 15 source-destination pairs sent CBR traffic with an interval time of 100 ms, generating heavy traffic along the established paths.

### 6.5.1 Packet Delivery Ratios

Figures 6.4, 6.5, and 6.6 show the packet delivery ratios for LSR and DSR for the above different traffic loads. In all simulated scenarios, the packet delivery ratio of LSR is better than that of DSR. There are two reasons for this phenomenon. First, LSR uses the strict path comparison function to choose a path with light traffic, reducing packet losses caused by buffer overflow in the congested nodes. Second, DSR aggressively uses the route cache mechanism [DPR00], which may use stale and incorrect route information and results in packet losses. In the light load scenario, however, the improvement in the packet delivery ratio of LSR is relatively small compared with the results in scenarios with “hot spots” and heavy load. This is because the traffic is light, and there is little chance to form congested nodes. The advantage of LSR using lightly loaded routes is not very obvious in this case. In both “hot spot” and heavy load scenarios, the packet delivery ratio for LSR is much better

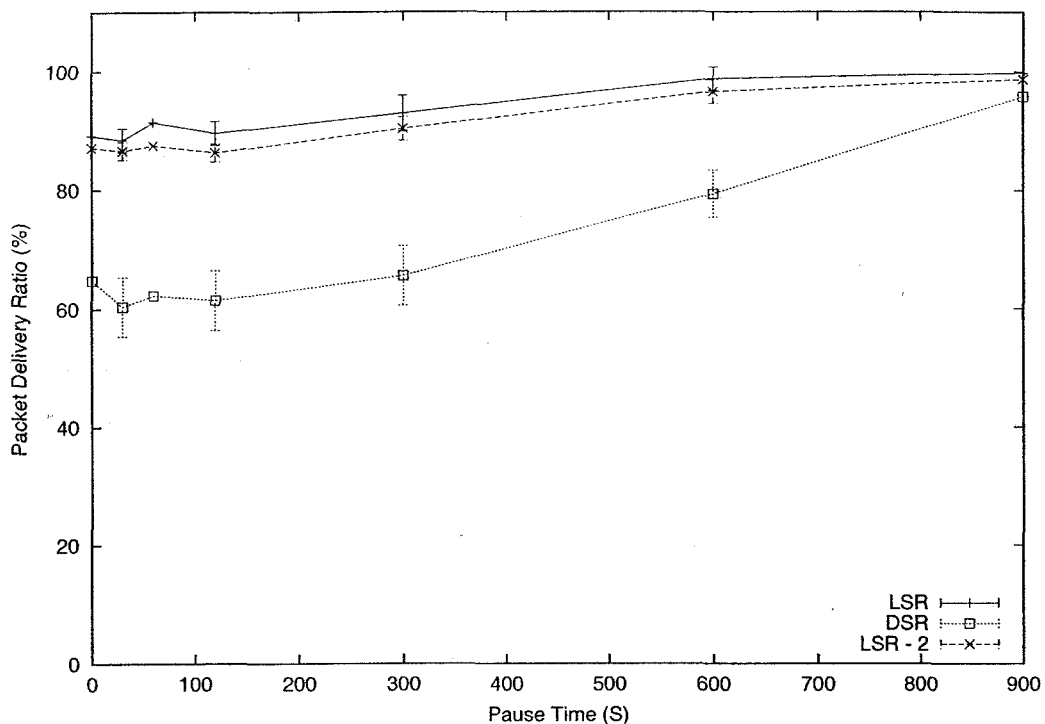


Figure 6.4: Packet delivery ratio (10 sources with background traffic scenario)

than for DSR. The reason is straightforward since LSR will keep a path away from the heavily loaded nodes whenever a lightweight path exists. In contrast, DSR does not consider load information in path selection and always chooses the shortest path, which may include congested nodes.

With the increase in mobility (the decrease in pause time), the packet delivery ratios for both LSR and DSR are decreased because paths are more likely to break with high mobility. A very interesting phenomenon is that the packet delivery ratio for LSR is more consistent and resilient to mobility. The route adaptation mechanism in LSR may contribute to this phenomenon. Using route adaptation, data traffic is likely to be transferred to a new path before the used one fails.

Under light traffic, LSR and LSR-2 are not obviously different. When the network includes some “hot spots” or when the end-to-end traffic is heavy, however, LSR outperforms LSR-2 because LSR chooses lightweight paths, and the neighbouring nodes along the paths are also likely to have light traffic. In contrast, LSR-2 can choose a lightly loaded path, whose neighbouring nodes, however, may have heavy traffic. In this case, the heavily loaded neighbouring nodes will contend for the

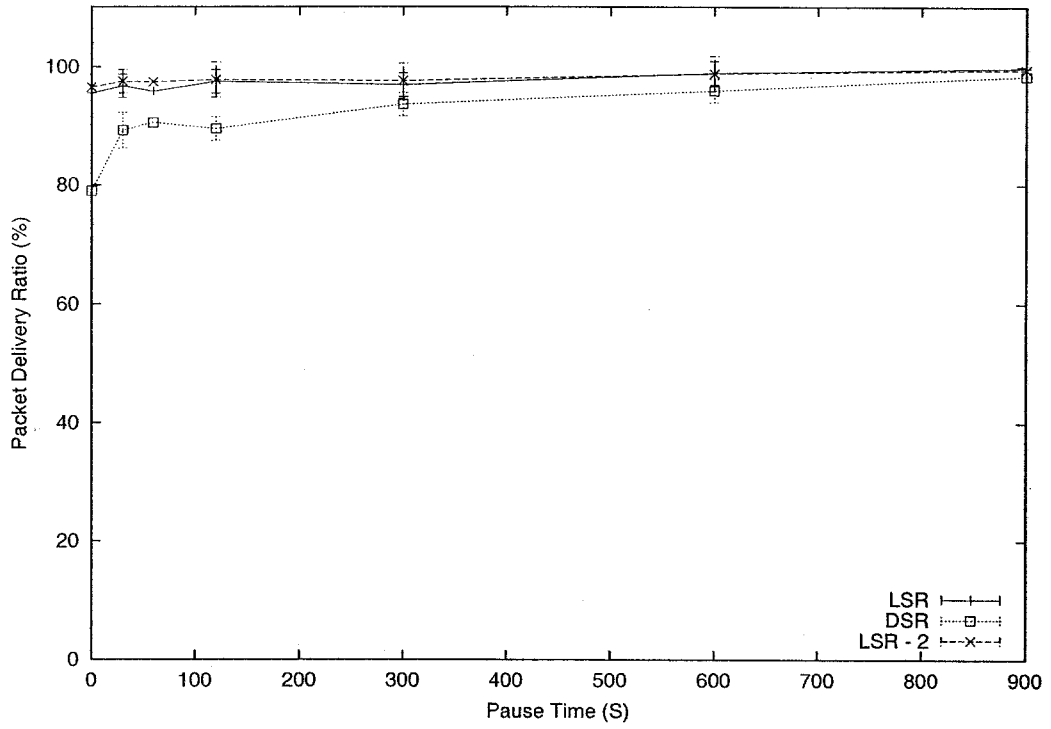


Figure 6.5: Packet delivery ratio (10 sources with light traffic scenario)

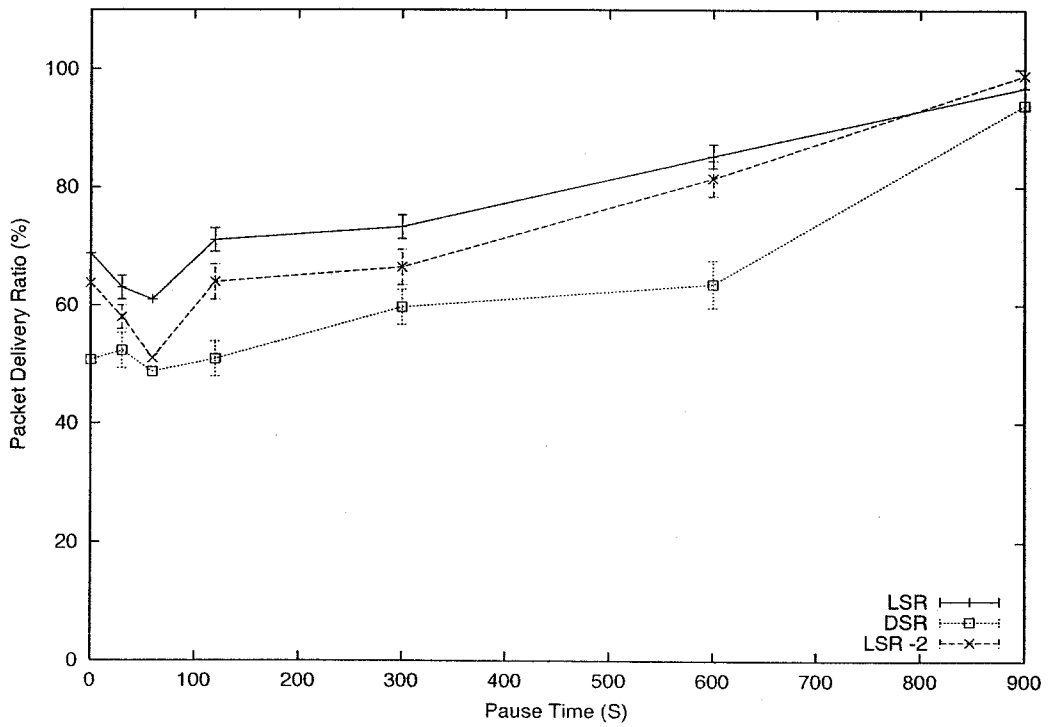


Figure 6.6: Packet delivery ratio (15 sources with heavy traffic scenario)



transmission channel and cause network congestion.

### 6.5.2 Average End-to-End Delay

Figures 6.7, 6.8, and 6.9 show the average end-to-end delay results for LSR and DSR under different traffic loads. In the “hot spot” scenario, the average end-to-end delay of LSR is better than that of DSR because LSR chooses a path with light traffic loads and thus reduces the queuing delay in the network interfaces. The end-to-end delay will be increased if a path includes nodes having heavy background traffic. This situation may occur in DSR but is less likely to happen in LSR since LSR has mechanisms to choose lightweight paths and can adjust traffic dynamically from a congested path to a less-loaded one.

When traffic load is light, however, LSR and DSR show little difference. The reason is that light network traffic does not cause network congestion, and network load information has little meaning in this context. Another strange phenomenon is that LSR’s end-to-end delay is slightly larger than DSR’s in the heavy load scenario. Although LSR can dynamically search for a lightweight path, the nodes along the newly established path will quickly become congested since the end-to-end traffic is always heavy. In this extremely bad situation, LSR does not help too much and may, in fact, make things worse. A newly found path in LSR usually has more hops, and the end-to-end delay will be increased. DSR, however, keeps transmitting messages along the shortest path at the cost of more packet losses in the heavy load scenario.

In both the “hot spot” and heavy load scenarios, LSR performs better than LSR-2 in terms of average end-to-end delay. The reason is straightforward: LSR-2 may choose a path whose neighbouring nodes are heavily loaded, and the transmission interference will increase the end-to-end delay. This phenomenon is also described in Chapter 5.

### 6.5.3 Control Overhead

Figures 6.10, 6.11, and 6.12 show the control overhead results. At high mobility, DSR has smaller control overhead than LSR, but there is little difference in the control overhead at low mobility. DSR uses route caching and allows intermediate nodes to respond to route requests. This method will quench the broadcast of route request

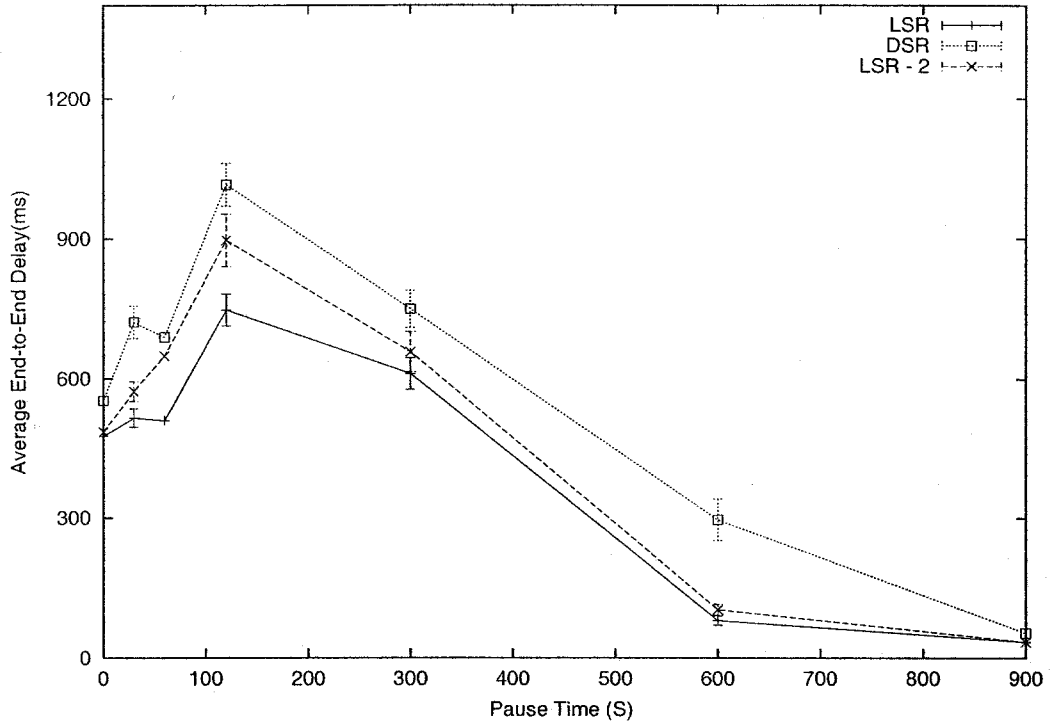


Figure 6.7: Average end-to-end delay (10 sources with background traffic scenario)

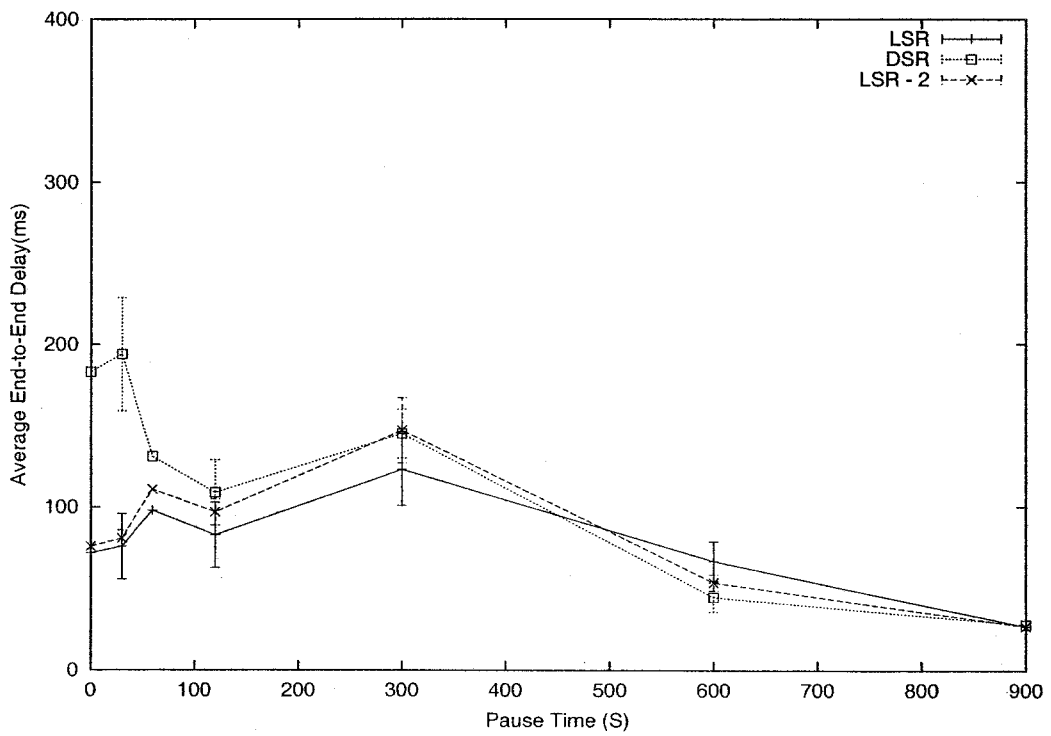


Figure 6.8: Average end-to-end delay (10 sources with light traffic scenario)

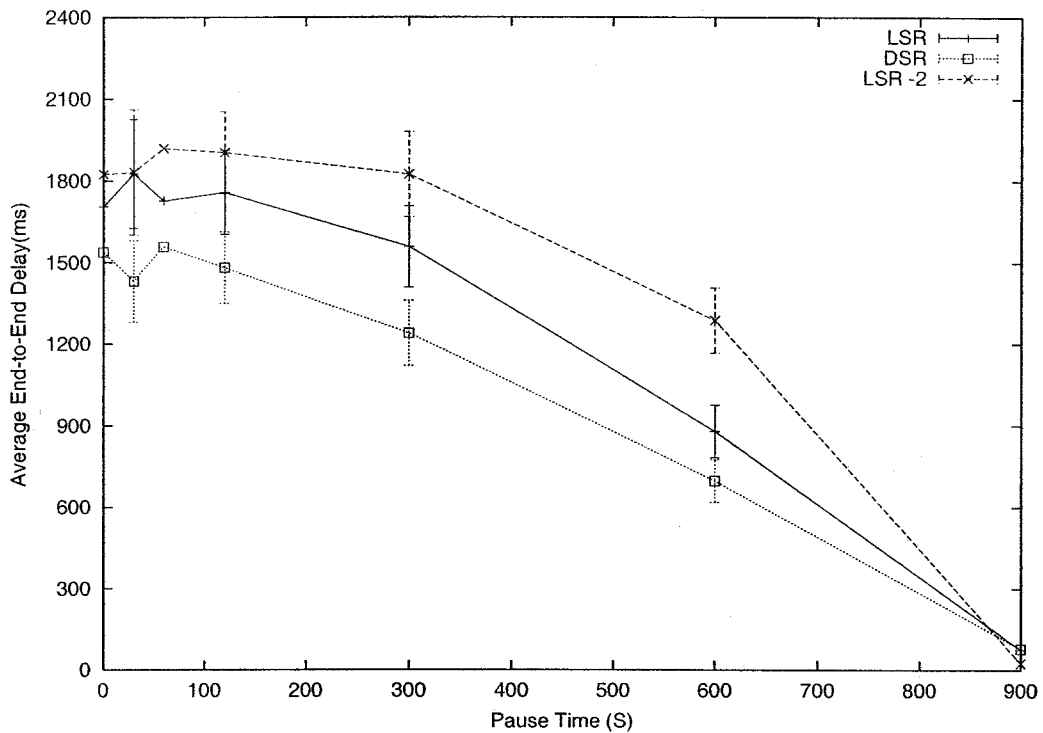


Figure 6.9: Average end-to-end delay (15 sources with heavy traffic scenario)

messages earlier and thus reduces control overhead. The disadvantage of route caching is that more route replies will be sent out and obsolete route information may be used. From the simulation results, however, we can see that the route-caching strategy in DSR reduces the control overhead at high mobility because more route requests will be promoted, and most route requests can be quenched and replied to the source by intermediate nodes. On the other hand, LSR does not allow intermediate nodes to reply to route requests. In addition, in LSR, the route adaptation mechanism to search dynamically for better paths during data transmission will also increase control overhead. Nevertheless, we think that the control overhead of LSR is acceptable since LSR can maintain a higher packet delivery ratio than DSR. The higher control overhead in LSR does not influence the performance too much, particularly for the packet delivery ratio.

LSR and LSR-2 have similar control overheads because enabling the promiscuous listening mode in LSR does not require additional transmissions of control packets.

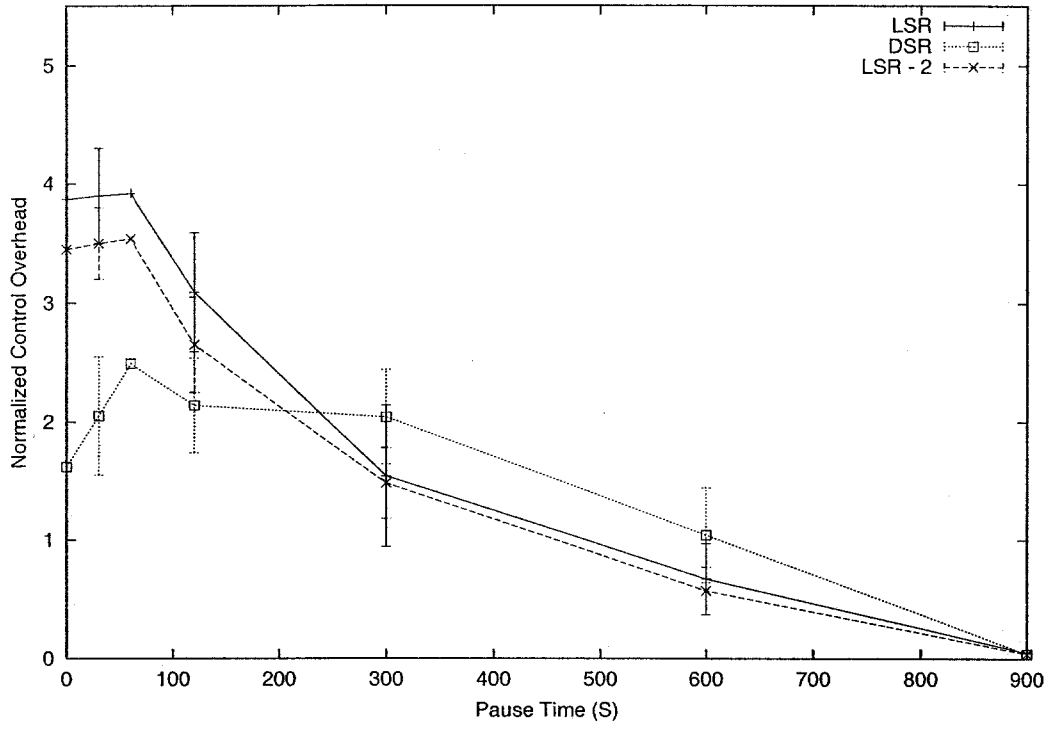


Figure 6.10: Control overhead (10 sources with background traffic scenario)

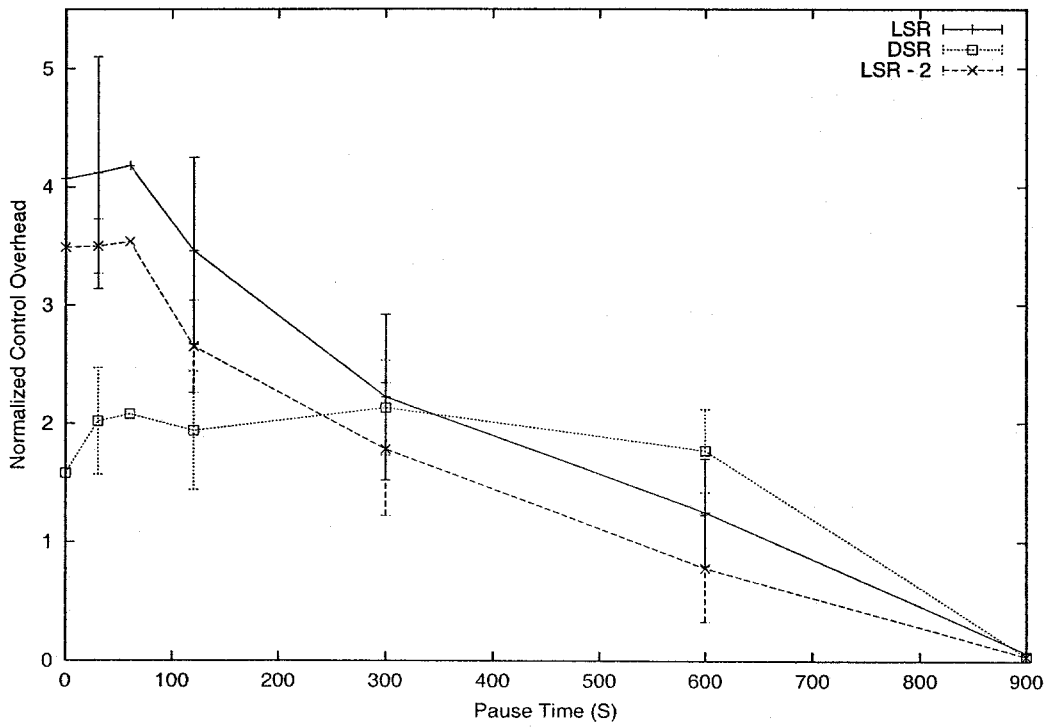


Figure 6.11: Control overhead (10 sources with light traffic scenario)

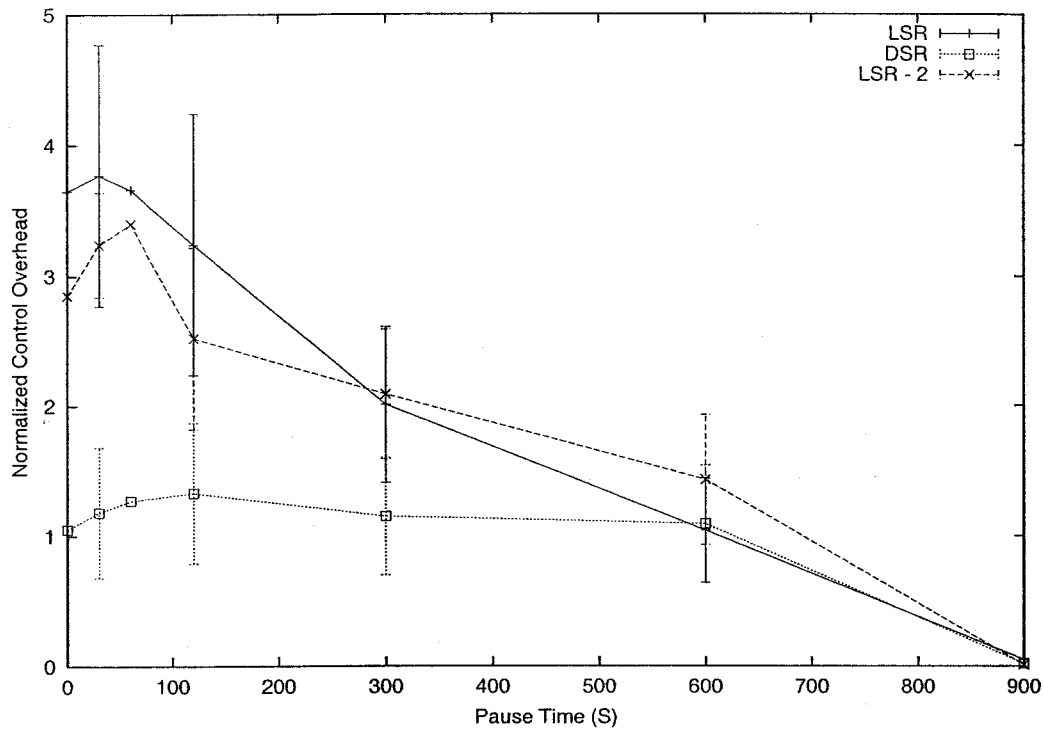


Figure 6.12: Control overhead (15 sources with heavy traffic scenario)

## 6.6 Conclusions

In this chapter, we presented a Load-Sensitive Routing (LSR) method for MANETs. Our method utilises network load information to assist in routing and adaptively selects lightweight paths during data transmission. We performed a simulation study on LSR. Compared with DSR, the results show that with the help of load information, LSR can improve performance in terms of packet delivery ratio and average end-to-end delay when a network includes some congested nodes. Furthermore, the control overhead is acceptable since LSR can maintain a high packet delivery ratio even at high mobility and it has no increase in control overhead at low mobility. We defined the path comparison functions to provide a unified scheme to tune the performance of LSR. We also studied the influence on performance of the path comparison functions in LSR and discussed the principles of choosing proper path comparison functions.

# Chapter 7

## Profile-Based Routing

### 7.1 Introduction

Wireless Mobile Ad hoc NETWORKS (MANETs) provide end-users with a flexible and low-cost way to access and exchange information. The promising application market for MANETs may be compromised, however, by the difficulty of providing services in MANETs. First, because of mobility, wireless links among mobile hosts can change very quickly, resulting in dynamic changes in network topology, message forwarding routes, and available bandwidth. Second, MANETs are power-constrained because mobile hosts usually rely on battery power. The implementation of routing and other services must be power-efficient, which implies that excessive costs for control information are unacceptable. Third, radio transmission is subject to the effects of multiple access, fading, noise, interference, etc. Finally, a MANET may be large, with thousands of mobile hosts, making network control difficult. Much work has been done to address these difficulties in MANETs. Most of the work, however, addresses these complex problems in general settings, that is, the proposed solutions are supposed to be suitable in all application scenarios.

We have observed that trying to address general problems is the main reason that currently proposed routing, multicasting, and other protocols are complicated and that no obvious winners seem to have become de facto standards for MANETs. On the other hand, more and more applications will soon create a huge market for MANETs. In reality, the behaviours of end users in most application environments are predictable or controllable. To mention a few examples, the movements of buses in the transportation system of a city are quite predictable. The behaviour of a

battalion can be controlled by its commander. And, in a sensor network for animal tracking, we can predict that a species of animals should roam more frequently around a particular area because of their habitat. Utilising the certainties in a particular application environment can simplify the implementation of a MANET. In industry, the ability to provide competitive services in a particular application environment is, without a doubt, profitable.

Instead of addressing general problems in MANETs, we suggest a different direction for the commercial evolution of MANETs [WHE01]. We utilise the information concerning end users' behaviours, or profiles, to simplify the implementation of MANETs and enhance network performance. To better understand the design principle of profile-based protocols, let us imagine the task of changing a flat car tire. A lug wrench usually provided with the car makes the job of unscrewing the nuts easy, although the same work might be done using a universal wrench if one has enough strength (it is even impossible to use a universal wrench to unscrew the nuts for some car models). The lug wrench is designed for a fixed nut size and has a long handle. Obviously, the profile information, that is, the fixed nut size and limited strength of most users, is considered when designing lug wrenches. We can expect it to work better than any generic wrench for this specific application.

The rest of the chapter is organized as follows. Section 7.2 introduces prior work. In Section 7.3, we introduce various architectures by which MANETs can be deployed. To demonstrate how the users' profile information can be used to facilitate network implementation, we propose a profile-based routing strategy for a realistic city transportation system and study its network performance through simulation in Section 7.4. Finally, we conclude this chapter in Section 7.5.

## 7.2 Prior Work

A lot of work (for example, [BMJHJ98, DCYS98, DPR00, LGT99, LSHGB00]) has been done to study the performance of routing protocols for MANETs. The common feature of these studies is that they assume that mobile hosts move in a random fashion and the protocols are supposed to work in any application environment. We observe that one protocol can out-perform another and vice versa, depending on the

features of an ad hoc network, such as connectivity, mobile speed, traffic patterns, etc. When the advantages of a protocol are stated, they usually implicitly include the prerequisite of the network characteristics. It is highly possible that none of the general-purpose protocols could be best for all MANETs.

A simulation study was performed for several routing protocols under different “realistic mobility scenarios” in [JLHMD99]. These included a conference scenario in which the speaker has low mobility and the audiences are rather static; an event scenario which involves a group of reporters in a political or sports event or stock brokers at a stock exchange; and a disaster scenario in which a rescue team works in a disaster area. To our knowledge, this is the first attempt to investigate routing problems in specific realistic environments. The motivation in [JLHMD99], however, is to understand how generic protocols would behave in an environment more realistic than the random scenarios, instead of exploiting the features of the environment as in a profile-based protocol. Their results may be helpful to understand the performance of a generic protocol when deployed in a particular application environment. Although they point out that it is necessary to have routing protocols specifically tuned to the characteristics of an ad hoc network, we argue that tuning a general-purpose protocol to work most efficiently in a particular environment may be more difficult and less efficient than designing a dedicated profile-based protocol due to the complexities of the general purpose protocol.

Tang and Baker [TB00] studied the users’ behaviours (activity, mobility, and traffic characteristics) in a local area wireless network in Stanford University. This work is valuable to understand the way users exploit a mobile network. In order to provide profile-based services in MANETs, understanding the users’ behaviours in a specified application environment is important. It is also an essential step to obtain useful profile information and to design the functionality of the profile agent. We believe that more work should be done to study users’ behaviours in particular realistic application scenarios.

Some protocols [KV98, Toh97] are proposed for general applications but seek help from a form of profile information, such as the location information in the GPS system, the associativity among mobile users, etc. Since the original design goal of these protocols is to support general cases, they still need to be modified to work



more efficiently in a specified application environment.

Several projects [MJKLD00, BH00, ZS00] are underway to provide profile-based services. For example, the CarNet system [MJKLD00] uses GPS in each car to obtain consistent location information; the Role-Based Multicast protocol [BH00] is dedicated to broadcasting accident information in a highway system; the Content Based Multicast (CBM) protocol [ZS00] dynamically chooses multicast receivers according to the content of messages and the potential interest of the messages to the receivers. These projects are all oriented to specific application environments and utilise users' profile information in the network layer protocols.

### 7.3 Node Architecture

In this section, we give a brief overview of some existing cost-effective technologies that form the basic building blocks of a general purpose ad hoc network. Subsequently, we discuss some possible architectural modifications to support profile-aware protocols that are particularly optimised to work in specific application environments similar to the public transit system.

Our discussion targets mobile end users running application programs that require the standard protocol stack. The ad hoc network forms a self-organized domain in which users communicate with each other, with occasional access to the Internet via some mobile hosts in the network designated as gateways. In the layered architecture, the application layer includes application programs and a range of application protocols that enable the interoperation of popular applications; the transport layer provides functionality for correct end-to-end transmission; the network layer provides mechanisms to search for paths from a source to a destination; and the physical layer (including physical links and Medium Access Control) is responsible for transmitting information between two directly-connected hosts.

This layered architecture provides us with a convenient and effective way of implementing and managing the system. Layering is a form of information hiding. For example, to develop the communication network for a city transportation system, application software developers need not know any details about lower layers. Meanwhile, network engineers, who are responsible for the implementation of the transport,

network, and physical layers, need not know the functionality of the application programs. Since how to develop the application software is beyond the scope of this discussion, we will simply describe how the lower three layers are implemented in the layered architecture.

First, no modification is needed for the transport layer. Second, since no industrial standard for the network protocols exists for MANETs, we can choose from currently proposed routing protocols such as Dynamic Source Routing (DSR) [MBJJ99, BJM01] or Ad hoc On-Demand Distance Vector Routing (AODV) [PR99, PRD01]. Third, in the physical layer, we can install a laptop computer with a WaveLan wireless card in every bus and use IEEE 802.11 as the Medium Access Control (MAC) protocol. The Industrial, Scientific, and Medical (ISM) band can be utilised for radio transmission.

Needless to say, the above network implementation is quick and feasible. However, the implementation is less efficient since the chosen routing protocol is supposed to work for general applications. Without modifications to accommodate the particular application environment, a generic routing protocol may work in an inefficient way. Our work in the next section shows that significant improvement can be obtained by using our newly proposed profile-aware protocol.

The system inefficiency is caused mainly by the generic network layer protocol we adopted. Although the principle of layering provides an effective method for system development and management, information hiding may lead to poor performance, especially in MANETs. Most network layer protocols proposed for MANETs operate with little information about users' behaviours, such as mobility, application environment, and users' requirements. Therefore, these protocols are generally complicated in order to deal with all possible cases, which may never or seldom occur in a particular application. In addition, most protocols do not provide a mechanism to utilise the users' profile information to simplify routing strategy and to enhance routing performance.

Based on the above observations, we next discuss some architectural changes to support profile-aware protocols better. Fig. 7.1 illustrates the new architecture. We use a database system, called the *profile database*, to store the profile information. The profile database includes information about users' behaviours. According to different application scenarios, the profile information may include the users' mobility pattern,

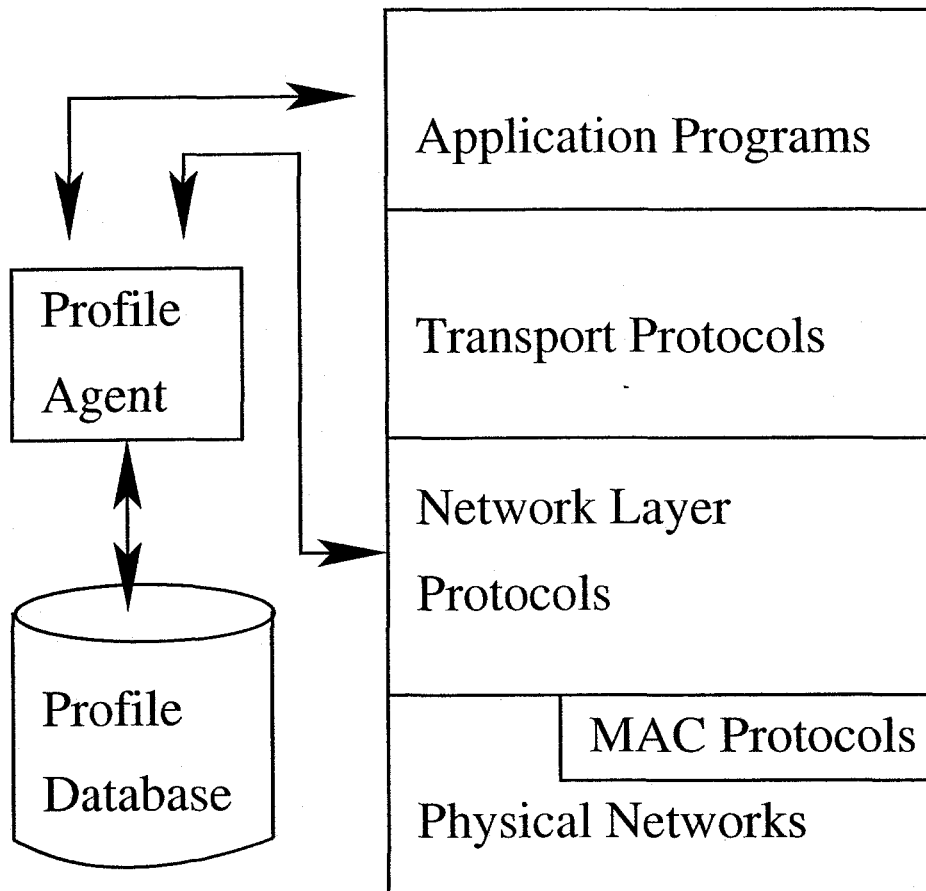


Figure 7.1: A profile-based node architecture

job schedule, possible communication requirements, mostly-used resources, etc. The profile database can be implemented in a distributed or centralised way, depending on the particular application environment.

Obviously, obtaining and managing the profile information may result in additional costs. Since the profile information is usually necessary for the application software, however, we argue that re-using the profile information at the network layer will not increase system costs dramatically. As in the case for the example described in Section 7.4, the profile database, which includes the timetable of a city transport system, can be manually installed in each mobile host before the mobile host joins the communication system. No or little information exchange is needed to manage the profile database in this special case. Note that the profile information needed by the application software and network layer protocols may be different, and thus it is a sound practice to design two profile databases separately for application programs and network layer protocols. In this case, it is necessary to be careful to maintain the consistency of the two profile databases if some common information is included.

Every mobile host also uses a *profile agent*. The basic functionality of the profile agent includes (1) searching the profile database for useful profile information, (2) automatically collecting newer profile information, (3) accepting pre-defined or manually-inputted profile information, and (4) updating the profile database if necessary.

The network layer protocols can be simplified and their performance can be improved if they properly use the profile information. For example, in the case of a dig team working in a tunnel, the routing strategy can be made basically on a line topology; for a public transportation system of a city, the routing decision can be made easily with the help of a bus timetable; in a tourist guide system, the tour schedule and some context information such as a tourist's interests could be used in designing the network layer protocol.

## 7.4 An Example: a City Transportation Wireless Communication System

As an example, we developed a realistic city transportation wireless communication system to demonstrate how to implement profile-based system architecture. We assumed that each bus was equipped with a radio communication device. Multi-hop radio transmissions are necessary since the radio transmission range is not large enough to cover the whole city.

A timetable and a city map are the main pieces of profile information utilised in the system. The profile database, which includes the above information, can be manually installed in every bus in the communication system. The profile agent, which is software installed in each bus, can calculate all buses' approximate locations based on the information (timetable and city map) from the local profile database. With the help of the location information, we propose a *directional geographical forwarding* strategy to implement our profile-based routing. According to the design principle of profile-based protocols, the proposed method is dedicated to this specific application environment, and we do not expect it to work well in all MANETs.

### 7.4.1 Profile-Based Routing

When a source node wants to communicate with a destination, it first gets the approximate location information of other nodes through its profile agent. Based on the location information, it can calculate a possible route to reach the destination, based on the shortest path algorithm. Because the buses may not be exactly on time and power depletion is possible in some nodes, this calculated path might not be a feasible one from the source to the destination. Nevertheless, the calculated path points out a good direction to forward messages to the destination. We call the calculated path a directional path.

The source node broadcasts data messages along this probable direction. Each data packet carries the directional path in its header. Once a non-destination node receives a data packet for the first time, if the location of this node is near enough to the directional path, it will broadcast this data packet again. Otherwise, it simply discards this data packet. Note that a node can check whether it is near to the

directional path based on the information obtained by its own profile agent and a threshold distance value. To prevent unlimited forwarding (looping), each node keeps a cache to record the received data packets' unique IDs and does not broadcast the same data packet twice. The data packet's unique ID includes the source ID, the destination ID, and the sequence number. The destination node will finally receive the data messages if the directional path and its nearby area include a connected path and the destination. If the source node cannot calculate a directional path based on the information from its profile agent, the source and the destination are most likely to be partitioned. In this case, the source node discards this message.

Fig. 7.2 shows an example of the above routing algorithm. Based on the location information, the source node S calculates a directional path to the destination node D before it broadcasts the data packets. Any nodes that are near the directional path (within the shadowed area) will re-broadcast received data. This strategy is tolerant of the location inaccuracy caused by possible inaccurate profile information. When a bus is late or early, it is still likely that the bus is in the shadowed area.

Compared with DSR described in Section 2.3.1, which uses no profile information and therefore includes complex route discovery and maintenance mechanisms, PBR does not need to flood control messages to search for routes with the help of the mobility profile information. In addition, PBR does not require each mobile host to keep the state of routes since the directional path in each data packet header and local profile information can determine how to handle the data packet (broadcast it or drop it). Therefore, PBR does not require a route maintenance mechanism.

## 7.4.2 Simulation Model

We compared the performance of our Profile-Based Routing (PBR) method with Dynamic Source Routing (DSR) [MBJJ99, BJM01], a generic routing protocol, in a realistic city transport system. The simulation model was built on GloMoSim [ZBG98] described in Chapter 3. We applied our profile-based routing method to a realistic wireless communication system based on Edmonton's downtown Transit System (ETS). The ETS map can be found at [ETS01]. For simplicity and without loss of generality, our simulation dealt with a 1500 metre x 1400 metre rectangular area, which extends north from 98 Avenue to 104 Avenue and east from 111 Street to

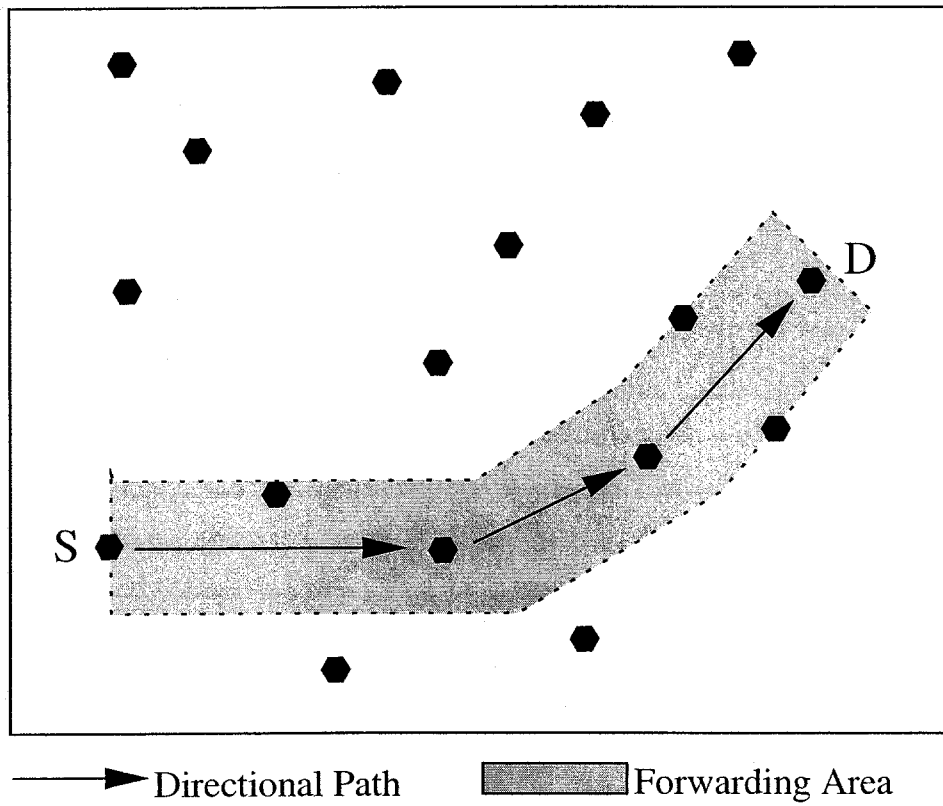


Figure 7.2: An example of profile-based routing.

100 Street. We assume that a bus will not take part in the communication once it is outside the simulated area. Fifty buses that run routinely in the simulated area were selected. We compared the performance of PBR and DSR in this realistic application environment. We built a trace-file for the simulation from the routine schedules of the fifty buses and the ETS map [ETS01].

In our simulation, the channel capacity of all mobile hosts was set to the same value: 2 Mbps. We used the Distributed Coordination Function (DCF) of IEEE 802.11 for wireless LANs as the MAC layer protocol. A free space propagation model with a threshold cutoff was used as the channel model. In the radio model, capture effects were taken into account (Chapter 3). We assume that all nodes have the same transmission range. We changed the value of the radio transmission range to study the influence of network connectivity.

The selected buses moved according to the schedules for a simulation time of 1 hour. In the simulation, we assumed that every bus ran strictly according to the timetable, and we also studied the performance of PBR if buses were out of schedule. Our proposed profile-based routing method permits time inaccuracy since the data messages are flooded along a limited directional region. The threshold value, which presents the nearby region along the directional path, was set to 300 metres. This value was set larger than the maximal width of the streets and can cover any crossroad in the shown area.

The simulated traffic was Constant Bit Rate (CBR). Different source and destination nodes were chosen randomly with uniform probabilities. The interval time to send packets was 250 ms. The size of all data packets was set to 512 bytes. All traffic was generated, and the statistical data were collected after a warm-up time of 100 seconds in order to give the nodes sufficient time to finish the initialisation process.

We evaluated the performance in terms of packet delivery ratio, average end-to-end delay, and normalized control head (See Chapter 3).



### 7.4.3 Simulation Results

#### Average end-to-end delay

Fig. 7.3 shows the results of average end-to-end delay of PBR and DSR with different traffic loads and different radio transmission ranges. When the radio transmission range is 300 metres, network partitions occur frequently. But when the radio transmission range is 500 metres, the network keeps connected most of the time. In both cases, the average end-to-end delay for PBR is much smaller than that for DSR. The reason is twofold. First, in DSR, a source node must search for a path to a destination before data transmissions if there is no route in its route cache. In contrast, PBR sends data packets immediately based on the calculation of the directional path. Second, if there is no path from a source to a destination when network partitions occur, DSR will buffer the data packet and continue to flood Route REQuest (RREQ) messages until the network becomes connected. This strategy may unnecessarily congest the network and enlarge the end-to-end delay. PBR, however, will not flood any data and control messages when network partitions happen, at the risk of potential packet losses if the profile information is not accurate and a feasible path to the destination might actually exist.

PBR has a relatively consistent average end-to-end delay. This is because the end-to-end delay in PBR depends mainly on the length of the directional path. In contrast, the end-to-end delay in DSR is more sensitive to the network topology and data traffic because DSR has no way to know about existing network partitions, and all data packets must be buffered until a path is found.

#### Control overhead

Fig. 7.4 shows the results of control overhead for PBR and DSR. When the radio transmission range is 300 metres, the control overhead for DSR is much larger than that for PBR. This is because DSR floods a large number of control packets into the network when network partitions occur. However, when the radio transmission range is 500 metres, the control overhead for DSR is only a little higher than that for PBR because DSR can quickly search for a new path if the radio transmission range is large. In this case, the control overhead for both DSR and PBR depends

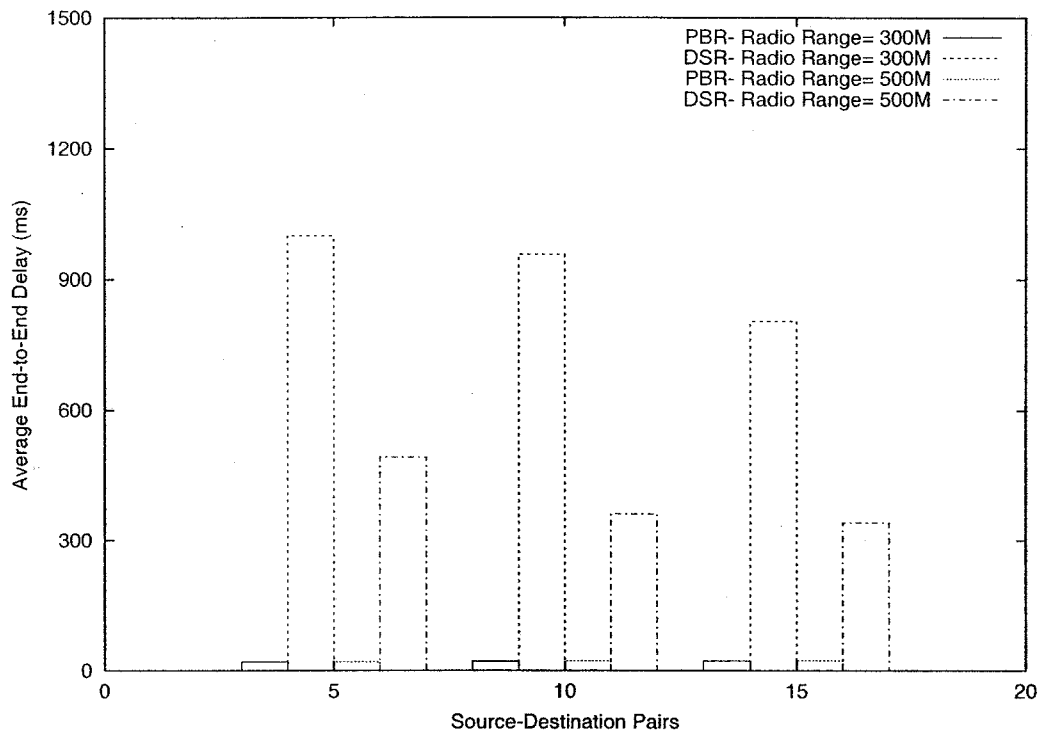


Figure 7.3: Average end-to-end delay for PBR and DSR

mainly on the control information included in each data packet. In DSR, the control information in each data packet is the source route; in PBR, it is the directional path. They should be approximately the same since both the source route in DSR and the directional path in PBR are likely to be the shortest paths from the source to the destination. As a result, DSR has slightly larger total control overhead because it requires extra cost for route discovery and maintenance.

### Packet delivery ratio

Fig. 7.5 shows the results for packet delivery ratios. Both PBR and DSR exhibit higher packet delivery ratios with the radio transmission range of 500 metres than with the radio transmission range of 300 metres because the network becomes partitioned more frequently at a smaller radio transmission range.

When the radio transmission range is 500 metres, DSR and PBR have very similar packet delivery ratios. This is because both protocols can quickly find feasible paths, and the control overhead is similar for DSR and PBR when the radio transmission range is 500 metres, as shown in Fig. 7.4. In other words, the control overhead has a

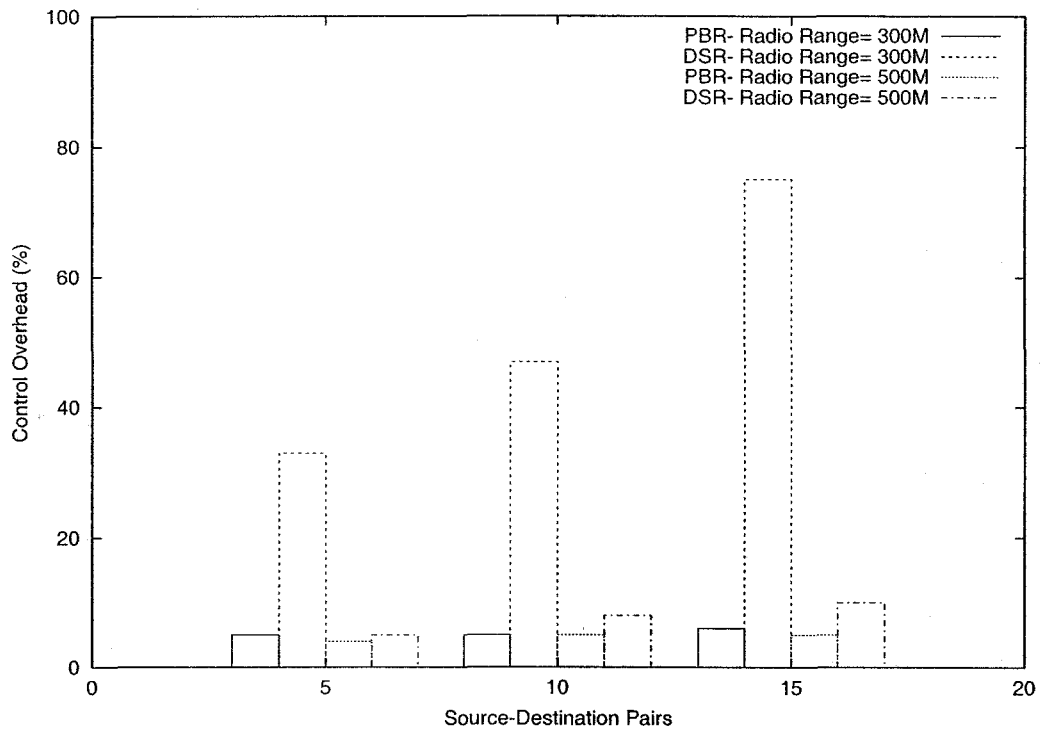


Figure 7.4: Control overhead for PBR and DSR

similar influence on data transmission.

When the radio transmission range is 300 metres, however, DSR and PBR have different packet delivery ratios, although the difference is not very large. When the CBR traffic includes 10 source-destination pairs, DSR and PBR exhibit very similar packet delivery ratios. When the traffic is lighter (5 source-destination pairs), DSR has a slightly higher packet delivery ratio than PBR. But when the traffic is heavier (15 source-destination pairs), the packet delivery ratio for PBR is slightly higher than that for DSR. When network partitions occur, DSR will buffer the data packets and delay the transmission until a path is found. This strategy may cause bursty data transmission if the network gets connected after a period of partition. When the traffic is light, this strategy may deliver more data packets. But when the traffic is heavy, too many bursty transmissions will cause losses of data packets. This is the main reason that DSR and PBR have different packet delivery ratios when the radio transmission range is small.

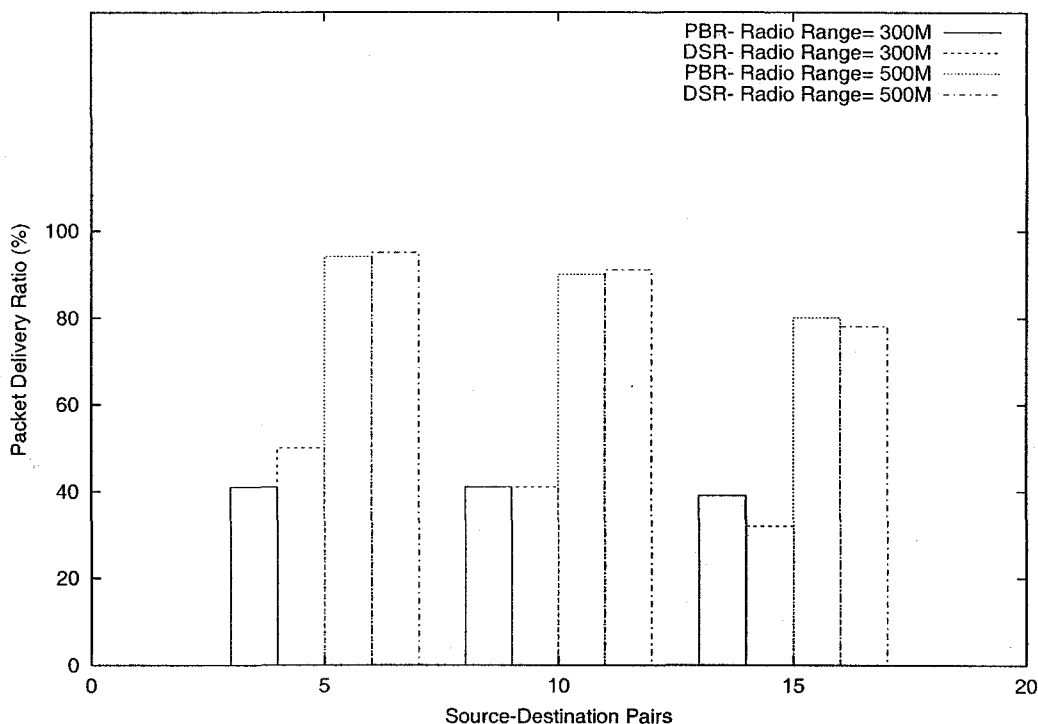


Figure 7.5: Packet delivery ratio for PBR and DSR

### The influence of profile inaccuracy

We investigated protocol performance with inaccurate location information. The location inaccuracy is introduced by allowing a difference between a node's actual location and its calculated location according to the profile information—the time schedule and the city map in this case. Note that the location inaccuracy was introduced at each bus station. In the simulation, the distance between a node's actual location and its calculated location (at each bus station) was uniformly distributed between  $(-d/2, d/2)$ , where  $d$  is the maximal location inaccuracy in metres. In this particular application, the location errors are introduced along the directions of the streets, which is a simple and reasonable assumption in this specific environment. We have not investigated profile errors such as in the case where buses are out of schedule and may arbitrarily distribute themselves in the city area. In this case, we suggest that the profile information is useless. The radio transmission range is set to 500 metres, and 10 CBR sessions are selected in this experiment.

Introducing location inaccuracy has no influence on the average end-to-end de-

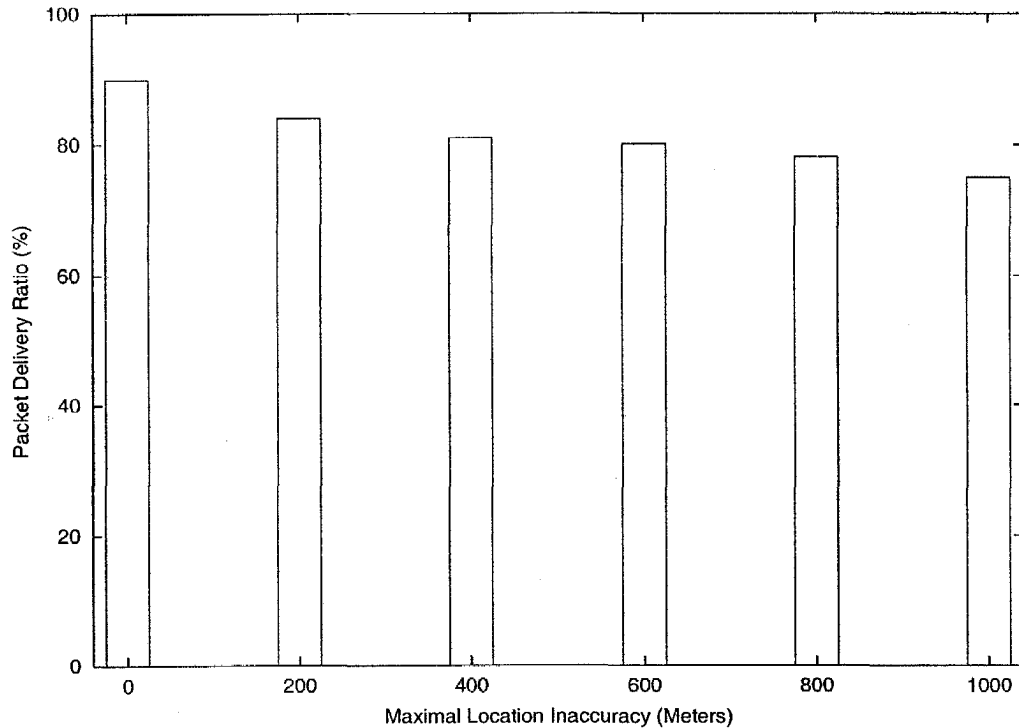


Figure 7.6: Packet delivery ratio with location inaccuracy

lay and control overhead since the former depends mainly on the actual hops from the source to the destination and the latter mainly on the threshold distance, which presents the shadowed area, as shown in Fig. 7.2. However, the location inaccuracy affects the packet delivery ratio. Fig. 7.6 shows the results of packet delivery ratios of PBR with different location inaccuracies. PBR has a lower packet delivery ratio at a larger location inaccuracy. Nevertheless, PBR is able to tolerate location inaccuracy quite well. The main reason is that the directions of the location inaccuracy are limited along the directions of streets, which are approximately coincident with the calculated *directional paths*. Data packets can finally reach the destination if the streets have a sufficient number of nodes (buses) to connect the source and the destination.

In conclusion, the profile-based routing protocol can provide a similar packet delivery ratio and have smaller average end-to-end delay and smaller control overhead compared to a generic protocol, DSR, in the ETS wireless communication system. Furthermore, it is resilient to location inaccuracy.

What we demonstrate here is the feasibility that a profile-based protocol can

outperform a generic protocol in a particular application area. Again, note that we do not expect a profile-based protocol to work well in all MANETs.

## 7.5 Conclusions

Future ad hoc networks are expected to provide services comparable in quality to the data services targeted by the third generation wireless cellular networks [BZM01]. If such expectations are met, future ad hoc networks can be relied on to provide access to a truly global networking infrastructure at a much lower user cost. Such expectations, however, appear to be most reasonable in environments where users' profiles can be efficiently predicted.

Motivated by the above prospect, we presented a different direction for the research and implementation of MANETs, which is aimed at specific application environments. As a concrete example, we considered a broad and useful class of restricted environments where mobility profiles were constrained by scheduling disciplines similar to these encountered in a typical public transit system. We proposed a simple and efficient routing protocol for such special environments. The protocol was tested on a collection of 50 bus routes, selected from an existing transit system, where each operational bus acted as a mobile host. Our results show that significant performance improvements can be attained as compared to the use of a generic ad hoc routing protocol.

While the competition of proposing and studying generic routing protocols for MANETs will continue fiercely, more attention should be paid to design profile-based network layer protocols. As shown in this chapter, utilizing users' profile information can simplify network layer design and improve the performance of network layer protocols. With the rapid growth in the commercial market for MANETs, these profile-based protocols can be standardized for specified types of application environments. Instead of dreaming of becoming "panacea" protocols, the profile-based protocols will prosper and provide competitive services for end-users in particular application environments.

# Chapter 8

## Conclusions and Future Research

### 8.1 Conclusions

In this dissertation, we investigated the routing problems in MANETs. As a fundamental support for other services and applications in MANETs, routing is an unavoidable task facing researchers and engineers. Routing in MANETs is extremely challenging because of frequent network topology changes, scarce power and bandwidth resources, and the unreliability of radio transmission. The main contributions and achievements of this dissertation include proposing and evaluating effective mechanisms to cope with these routing difficulties in MANETs. We addressed the routing problem from the following different perspectives.

First, we utilised users' mobility profiles to enhance routing performance. In our proposed L<sub>O</sub>cation Trace Aided Routing (LOTAR) protocol, location information is used to limit the route discovery scope, search quickly for a feasible path if links break, and support dynamic flow handoffs. LOTAR provides dynamic local and global handoff mechanisms that work together with particular link prediction methods. We performed simulation studies to investigate its performance with different mobility models, as well as its performance with inaccurate location information. We also presented the performance advantages of flow handoffs in an ideal scenario, demonstrating a promising direction for the development of dynamic soft handoff mechanisms.

Second, in order to distribute the routing tasks among mobile hosts evenly, we proposed two routing methods—multipath routing and load sensitive routing—to keep a node from heavy duties. For the multipath routing method, we proposed

two on-demand multipath routing protocols to effectively search for node-disjoint paths without whole topology information. Compared with other multipath protocols, our approaches can find good quality node-disjoint paths. Simulation studies show that our heuristic redirection multipath routing protocol can reduce route discovery frequency, balance network load and power consumption, and reduce end-to-end delay. In load sensitive routing, we utilised network load information as a main route selection criterion and provided a unified scheme to tune network performance. The proposed protocol is non-blocking and can dynamically adjust the routing paths according to network load information. Compared with Dynamic Source Routing (DSR), load sensitive routing offers a higher packet delivery ratio and lower average end-to-end delay. Furthermore, these benefits are gained without an increase in control overhead at low mobility.

Third, in order to simplify routing implementation, we proposed a profile-based routing scheme that utilises the specific characteristics of particular application scenarios where end-users' behaviours are quite predictable or controllable. Through a realistic example, we demonstrated that the users' profile information in a specific application environment is helpful to simplify routing decisions and improve routing efficiency. We pointed out this feasible commercial evolution of MANETs and expect that more research will be done along this direction.

## 8.2 Future Research

### 8.2.1 Supporting Link Asymmetry

Most routing protocols for MANETs assume that all wireless links are symmetric (node A and node B are within each other's transmission range). This assumption, however, does not reflect real life scenarios because different nodes may have different power levels and thus different transmission ranges. It is very common that a realistic MANET may include asymmetric links (node A is within node B's transmission range, but not vice versa). Link asymmetry will greatly reduce the number of feasible paths obtained by the source node for most on-demand routing protocols since a route reply message cannot be sent back to the source node if the path from the source to the destination includes asymmetric links.



However, the direct influence of link asymmetry is on the MAC layer protocols. Most MAC layer protocols, such as IEEE 802.11 described in Chapter 3, depend on explicit acknowledgement from the receiver to guarantee correct transmission. The sender will assume that the link to the receiver breaks if it fails to receive acknowledgement from the receiver after several retransmissions. This is not true if the link from the sender to the receiver is unidirectional. In this case, a link breakage notification will be sent up to the network layer protocols.

Based on the above observation, in order to support link asymmetry in the network layer protocols, we need to address the problem in the MAC layer first. Alternatively, expanding routing protocols to include the MAC layer mechanisms to handle asymmetric links might be a feasible solution. How to design effective mechanisms to support asymmetric links will be future research.

### 8.2.2 Multicasting

Group communication is important to applications characterized by the close collaboration of teams. Multicasting can efficiently support group communication and will be a main service for MANETs due to special application environments such as battlefield searching and disaster recovery. However, multicasting in a multi-hop wireless environment is much more difficult than in other kinds of networks due to frequent topology changes. How to design efficient multicast routing protocols deserves further study.

Traditional multicast protocols cannot be used directly in MANETs since they have no mechanisms to deal with topology changes. Lee et al. [LSHGB00] demonstrated that multicasting with mesh topology has better performance than with tree topology, especially when the nodes' mobile speeds in MANETs are fast. The main reason is that the mesh topology can provide redundant paths for multicast routing while the tree structure is too sensitive to mobility. Further work on multicasting must address how to make a multicast protocol resilient to mobility.

### 8.2.3 QoS Support

Quality of Service (QoS) support in MANETs is very difficult because the available resources change dynamically with the nodes' mobility. The traditional meaning of

QoS support that some performance metrics must be guaranteed once a request is accepted is no longer true in MANETs [WH01-4]. In some sense, we cannot guarantee any strict performance constraints because of the dynamic network topology. In a MANET, after a service is accepted, the network will try its best to satisfy the performance requirements, e.g., bandwidth and end-to-end delay constraints.

We have observed that some fundamental mechanisms in our proposed routing protocols will benefit QoS support in MANETs. The flow handoff mechanisms in LOTAR (Chapter 4) can maintain end-to-end connectivity for a long time, and the load balancing mechanisms in the multipath routing method (Chapter 5) and in the load sensitive routing method (Chapter 6) can fairly distribute routing tasks and reduce the possibility of network congestion. In the future, we will design new QoS routing schemes that utilise the functionality provided in our routing protocols to satisfy flexible QoS requirements.

#### 8.2.4 Middleware

Profile-based routing protocols utilise the mobile users' profile information to improve routing performance. The mobile users' profile information includes users' movement schedules, communication patterns, the processing capacity of devices, most-used resources, etc. This kind of information represents approximate current networks status. Nevertheless, this kind of information is dynamically changing, and an efficient routing protocol should automatically adjust its routing strategy according to current network status.

Middleware is a layer of software that provides enabling services that allow multiple processes running on one or more machines to interact across a network. This software provides services such as identification, authentication, directories, security, transaction monitoring, etc. The middleware method is effective in providing development and runtime support to context-sensitive applications [YK01]. Although mobile users' profile information might be easily obtained in some specific application environments (Chapter 7), the middleware method provides a more general scheme to obtain and manage mobile users' profile information. How to use this approach to achieve context-sensitive routing deserves further study.

### 8.2.5 Security

Although security is a major concern for users in MANETs, providing security in MANETs is particularly difficult to achieve because of the lack of centralised control, the vulnerability of wireless links, the dynamically changing topology, and the absence of a certification authority [HBC01]. In MANETs, new types of security breaches may exist, making the problem more complex. For example, how to prevent selfishness (a node does not provide routing services for others in order to save power) is not an easy task.

Currently, although a lot of routing protocols are proposed for MANETs, none of them discusses the security problem in detail [WG01]. This is mainly because security in MANETs is a young area and not very well understood. As a basic step, a comprehensive study of the possible threats in MANETs is necessary. To deal with such threats, how to design efficient mechanisms to secure MANETs and how to make tradeoffs among security, system cost, and performance will be future topics.

# Bibliography

- [ABB96] A. Acharya, A. Bakre, and B.R. Badrinath, "IP Multicast Extension for Mobile Internetworking," *Proceedings of IEEE INFOCOM 1996*, San Francisco, CA, March 1996, pp. 67-74.
- [ACH98] C. Aurrecochea, A. T. Campbell, and L. Hauw, "A Survey of QoS Architectures," *ACM/Springer Verlag Multimedia Systems Journal*, Special Issue on QoS Architecture, Vol. 6, No. 3, May 1998, pp. 138-151.
- [ACLZ99] G.S. Ahn, A. T. Campbell, S.B. Lee, and X. Zhang, "INSIGNIA," *Internet Draft*, draft-ietf-manet-insignia-01.txt, October 1999.
- [BCS94] R. Braden, D. Clark, and S. Shenker, "Integrated Services in the Internet Architecture- an Overview," *IETF RFC1663*, June 1994.
- [BDSZ94] V. Bharghavan, A. Demers, S. Shenker, and L. Zhang, "MACAW: A Media Access Protocol for Wireless LANs," *Proceedings of ACM SIGCOMM 1994*, London, UK, August 1994, pp. 212-225.
- [Bell57] R. E. Bellman, *Dynamic Programming*, Princeton, NJ, Princeton University Press, 1957.
- [BFC93] T. Ballardie, P. Francis, J. Crowcroft, "Core Based Trees (CBT): An architecture for Scalable Inter-Domain Multicast Routing." *Proceedings of ACM SIGCOMM 1993*, San Francisco, CA, August 1993, pp. 85-95.
- [BH00] L. Breisemeister and G. Hommel, "Role-Based Multicast in Highly Mobile but Sparsely Connected Ad Hoc Networks," *Proceedings of IEEE/ACM MobiHoc 2000*, Boston, MA, August 2000, pp. 45-50.
- [BJM01] J. Broch, D. Johnson, and D. Maltz, "The Dynamic Source Routing Protocol for Mobile Ad hoc Networks," *Internet Draft*, draft-ietf-manet-dsr-05.txt, March 2001.
- [Blake98] S. Blake, "An Architecture for Differentiated Services," *IETF RFC2475*, December 1998.
- [BMJHJ98] J. Broch, D.A. Maltz, D.B. Johnson, Y.C. Hu, and J. Jetcheva, "A Performance Comparison of Multi-Hop Wireless Ad Hoc Network Routing Protocols," *Proceedings of MobiCom 1998*, Dallas, TX, October 1998, pp. 85-97.
- [BMT98] R. Bagrodia, R. Meyer, M. Takai, Y. Chen, X. Zeng, J. Martin, B. Park, H. Song, "Parsec: A Parallel Simulation Environment for Complex Systems," *IEEE Computer*, Vol. 31, No. 10, October 1998, pp. 77-85.

- [BZM01] Q. Bi, G. I. Zysman, and H. Menkes, "Wireless Mobile Communications at the Start of the 21st Century," *IEEE Communications Magazine*, Vol. 39, No. 1, January 2001, P.110-116.
- [BZBHJ97] R. Braden, L. Zhang, S. Berson, S. Herzog, and S. Jamin, "Resource reSerVation Protocol (RSVP)- Version 1 Functional Specification," *IETF RFC 2205*, September 1997.
- [CF98] D. D. Clark and W. Fang, "Explicit Allocation of Best-effort Packet Delivery Service," *IEEE/ACM Transactions on Networking*. Vol. 6, No. 4, August 1998, pp. 362-373.
- [CG98] T.W. Chen and M. Gerla, "Global State Routing: A New Routing Scheme for Ad-hoc Wireless Networks," *Proceedings of IEEE ICC'98*, Atlanta, GA, June 1998, pp. 171-175.
- [CGZ98] C.C. Chiang, M. Gerla, and L. Zhang, "Forwarding Group Multicast Protocol (FGMP) for Multihop, Mobile Wireless Networks." *ACM-Baltzer Journal of Cluster Computing: Special Issue on Mobile Computing*, vol.1, no. 2, 1998, pp. 187-196.
- [Chen98] T.W. Chen , "Efficient Routing and Quality of Service Support for Ad Hoc Wireless Networks," University of California, Los Angeles, Ph.D Dissertation, 1998.
- [Chiang98] C.C. Chiang, "Wireless Network Multicast," University of California, Los Angeles, Ph.D Dissertation, 1998.
- [CN98] S. Chen and K. Nahrstedt, "An Overview of Quality-of-Service Routing for the Next Generation High-Speed Networks: Problems and Solutions," *IEEE Network: Special Issue on Transmission and Distribution of Digital Video*, Vol. 12, No. 6, November 1998, pp. 64-79.
- [CN99] S. Chen and K. Nahrstedt, "Distributed Quality-of-Service Routing in Ad-Hoc Networks," *IEEE Journal on Special Areas in Communications*, Vol. 17, No. 8, August 1999, pp. 2580-2592.
- [DC90] S. E. Deering and D. R. Cheriton, "Multicast Routing in Datagram Internetworks and Extended LANs." *ACM Transactions on Computer Systems*, Vol. 8, No. 2, May 1990, pp. 85-110.
- [DCYS98] S.R. Das, R. Castaneda, J. Yan, and R. Sengupta, "Comparative Performance Evaluation of Routing Protocols for Mobile Ad Hoc Network," *Proceedings of the 7th Int. Conf. On Computer Communications and Networks (IC3N)*, Lafayette, LA, October 1998, pp. 153-161.
- [Dee89] S. Deering, "Host Extension for IP Multicasting," *IETF RFC 1112*, Aug. 1989.
- [DEF97] S. Deering, D. Estrin, D. Farinacci, V. Jacobson, A. Helmy, and L. Wei, "Protocol Independent Multicast Version 2, Dense Mode Specification," *Internet Draft*, 1997.
- [DPR00] S.R. Das, C.E. Perkins, and E.M. Royer, "Performance Comparison of Two On-Demand Routing Protocols for Ad-Hoc Networks," *Proceedings of IEEE INFOCOM 2000*, Tel Aviv, Israel, March 2000, pp. 3-12.

- [ETS01] ETS Web Site, <http://www.gov.edmonton.ab.ca/transit>
- [FG95] C.L. Fullmer and J.J. Garcia-Luna-Aceves, "Floor Acquisition Multiple Access (FAMA) for Packet-radio Networks," *Proceedings of ACM SIGCOMM 1995*, Cambridge, MA, 1995, pp. 262-273.
- [GAPK01] T. Goff, N. Abu-Ghazaleh, D. S. Phatak and R. Kahvecioglu, "Pre-emptive Routing in Ad Hoc Networks," *Proceedings of Mobicom 2001*, July 2001, Rome, pp. 43-52.
- [GBH97] R. Guerin, S. Blake, and S. Herzog, "Aggregating RSVP-based QoS Requests," *Internet draft*, draft-guerin-aggreg-RSVP-00.txt, Nov. 1997.
- [GG96] P. Georgatsos and D. Griffin, "A Management System for Load Balancing Through Adaptive Routing in Multiservice ATM Networks," *Proceedings of IEEE INFOCOM 1996*, San Francisco, CA, March 1996, pp. 863-870.
- [GM99] P. Gupta and N. McKeown, "Packet classification on multiple fields," *Proceedings of ACM SIGCOMM 1999*, Cambridge, MA, Aug. 1999, pp. 135-145.
- [GM99-2] J.J. Garcia-Luna-Aceves and E.L. Madruga, "The Core-Assisted Mesh Protocol," *IEEE Journal on Selected Areas in Communications*, Vol. 17, No. 8, August 1999, pp. 1380-1394.
- [GO97] R. Guerin and A. Orda, "QoS Routing in Networks with Inaccurate Information: Theory and Algorithms," *Proceedings of IEEE INFOCOM 1997*, Japan, April 1997, pp. 75-83.
- [GPLC98] M. Gerla, G. Pei, S.J. Lee, and C.C. Chiang, "On-Demand Multicast Routing Protocol (ODMRP) for Ad Hoc Networks," *Internet Draft*, Nov. 1998.
- [GTB99] M. Gerla, K. Tang, and R. Bagrodia, "TCP Performance in Wireless Multi-hop Networks," *IEEE Workshop on Mobile Computing Systems and Applications (WMCSA)*, New Orleans, LA, Feb. 1999.
- [HBC01] J.P. Hubaux, L. Buttyan, S. Capkun, "The Quest for Security in Mobile Ad Hoc Networks," *Proceedings of IEEE/ACM MobiHoc 2001*, California, Oct. 2001.
- [HHSWW99] A. Harter, A. Hopper, P. Steggles, A. Ward, and P. Webster, "The Anatomy of a Context-Aware Application," *Proceedings of MobiCom 1999*, Seattle, WA, August 1999, pp. 59-68.
- [HP00] Z.J. Haas and M.R. Pearlman, "The Zone Routing Protocol (ZRP) for Ad-hoc Networks," *Internet Draft*, draft-ietf-manet-zone-zrp-03.txt, March 2000.
- [HZ01] H.S. Hassanein and A. Zhou, "Routing with Load Balancing in Wireless Ad Hoc Networks," *Proceedings of ACM International Workshop on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, Rome, Italy, July 2001.
- [ICPGC99] A. Iwata, C.C. Chiang, G. Pei, M. Gerla and T.W. Chen, "Scalable routing strategies for ad hoc wireless networks," *IEEE Journal on Selected Areas in Communications*, Aug. 1999, pp. 1369-1379.

- [IEEE97] IEEE Computer Society LAN MAN Standards Committee, "*Wireless Lan Medium Access Protocol (MAC) and Physical Layer (PHY) Specification*," IEEE Std 802.11-1997, the Institute of Electrical and Electronics Engineers, New York, NY, 1997.
- [IN98] J. Ibanez and K. Nichols, "Preliminary Simulation Evaluation of an Assured Service," *Internet Draft*, draft-ibanez-diffserv-assured-eval-00.txt, August 1998.
- [JLHMD99] P. Johansson, T. Larsson, N. Hedman, B. Mielczarek, and M. Degermark, "Scenario-Based Performance Analysis of Routing Protocols for Mobile Ad-Hoc Networks," *Proceedings of MobiCom 1999*, Seattle, WA, Aug. 1999, pp. 195-206.
- [Johnson99] D. B. Johnson, "Validation of Wireless and Mobile Network and Simulation," *Proceedings of the DARPA/NIST Workshop on Validation of Large-Scale Network Models and Simulation* Fairfax, VA, May 1999.
- [Kes97] S. Keshav, *An Engineering Approach to Computer Networking: ATM Networks, the Internet, and the Telephone Network*, Addison-Wesley professional computing series, 1997.
- [KK00] B. Karp and H. T. Kung, "GPSR: Greedy Perimeter Stateless Routing for Wireless Networks," *Proceedings of MobiCom 2000*, Boston, August 2000, pp. 243-254.
- [KV98] Y.B. Ko and N.H. Vaidya, "Location-Aided Routing (LAR) in Mobile Ad Hoc Networks," *Proceedings of MobiCom 1998*, Dallas Texas, October 1998, pp. 66-75.
- [KV00] Y.B. Ko and N.H. Vaidya, "Anycasting and Geocasting in mobile ad hoc networks," *Tech. Rep. 00-015, Dept. of Computer Science, Texas A&M University*, June 2000.
- [LBS97] S. Lu, V. Bharghavan, and R. Srikant, "Fair Scheduling in Wireless Packet Networks," *Proceedings of ACM SIGCOMM 1997*, Cannes, France, Sept. 1997, pp. 63-74.
- [LG01] S.J. Lee and M. Gerla, "SMR: Split Multipath Routing with Maximally Disjoint Paths in Ad hoc Networks," *Proceedings of ICC 2001*, Helsinki, Finland, June 2001.
- [LG01-2] S.J. Lee and M. Gerla, "Dynamic Load-Aware Routing in Ad hoc Networks," *Proceedings of ICC 2001*, Helsinki, Finland, June 2001.
- [LGT99] S.J. Lee, M. Gerla, and C.K. Toh, "A Simulation Study of Table-Driven and On-Demand Routing Protocols for Mobile Ad Hoc Networks," *IEEE Network*, July, 1999. 48-54.
- [LHCKM00] J. Li, J. Hannotti, D.D. Couto, D. Karger, and R. Morris, "A Scalable Location Service for Geographic Ad-Hoc Routing," *Proceedings of MobiCom 2000*, Boston, August 2000, pp. 120-130.
- [LO98] H. Lorenz and A. Orda, "QoS Routing in Networks with Uncertain Parameters," *IEEE/ACM Transactions on Networking*. Vol. 6, No. 6, August 1998, pp. 768-778.

- [LSHGB00] S.J. Lee, W. Su, J. Hsu, M. Gerla, and R. Bagrodia, "A Performance Comparison Study of Ad Hoc Wireless Multicast Protocols," *Proceedings of IEEE INFOCOM 2000*, Tel Aviv, Israel, Mar.2000, pp. 565-574.
- [Luc00] Lucent Technologies, "WaveLan product specification," <http://www.wavelan.com>, 2000.
- [Max93] N.F. Maxemchuk, "Dispersity Routing in High-speed Networks," *Computer Networks and ISDN system*, Vol. 25, 1993, pp. 645-661.
- [MBJ00] D.A. Maltz, J. Broch, and D.B. Johnson, "Quantitative Lessons from a Full-scale Multi-hop Wireless Ad Hoc Network Testbed," *Proceedings of the IEEE Wireless Communications and Networking Conference*, Chicago, September 2000.
- [MBJJ99] D.A. Maltz, J. Broch, J. Jetcheva, and D.B. Johnson, "The Effects of On-Demand Behaviour in Routing Protocols for Multi-Hop Wireless Ad Hoc Networks," *IEEE Journal on Selected Areas in Communications Special Issue on Mobile and Wireless networks*, August 1999.
- [MG98] A. Muir and J.J. Garcia-Luna-Aceves, "An Efficient Packet-sensing MAC Protocol for Wireless Networks," *ACM Journal on Mobile Networks and Applications*, Vol. 3, No. 2, August 1998, pp. 221-234.
- [MJKLD00] R. Morris, J. Jannotti, F. Kaashoek, J. Li, and D. Decouto, "CarNet: A Scalable Ad Hoc Wireless Network System," *9th ACM SIGOPS European Workshop*, Kolding, Denmark, September 2000.
- [MLTB98] A. McAuley, M.K. Liu, R. Talpade, E. Bommaiah, "AMRoute: Adhoc Multicast Routing Protocol." *Internet Draft*, August 1998.
- [ND99] A. Nasipuri and S. R. Das, "On-Demand Multipath Routing for Mobile Ad Hoc Networks," *Proceedings of the 8th Int. Conf. On Computer Communications and Networks (IC3N)*, Boston, October 1999, pp. 64-70.
- [NI97] J.C. Navas and T. Imielinski, "Geocast- geographic addressing and routing," *Proceedings of MobiCom 1997*, Budapest, Hungary, September 1997, pp. 66-76.
- [NJZ99] K. Nichols, V. Jacobson, and L. Zhang, "A Two-bit Differentiated Services Architecture for the Internet," *IETF RFC2638*, July 1999.
- [PB94] C.E. Perkins and P. Bhagwat, "Highly Dynamic Destination-Sequenced Distance-Vector Routing for Mobile Computers," *Proceedings of ACM SIGCOMM 1994*, London, UK, Aug. 1994, pp. 234-244.
- [PC97] V.D. Park and M.S. Corson, "A Highly Adaptive Distributed Routing Algorithm for Mobile Wireless Networks," *Proceedings of IEEE INFOCOM 1997*, Japan, April 1997, pp. 1405-1413.
- [PCB00] N.B. Priyantha, A. Charkraborty, and H. Balakrishnan, "The Cricket Location-Support System," *Proceedings of MobiCom 2000*, Boston, August 2000, pp. 32-43.



- [PHST00] M.R. Pearlman, Z.J. Hass, P. Sholander, and S.S. Tabrizi, "On the Impact of Alternate Path Routing for Load Balancing in Mobile Ad Hoc Networks," *Proceedings of IEEE/ACM MobiHoc 2000*, Boston, Aug. 2000, pp. 3-10.
- [PR99] C.E. Perkins and E.M. Royer, "Ad-hoc On-Demand Distance Vector Routing," *Proceedings of the 2<sup>nd</sup> IEEE Workshop on Mobile Computing Systems and Applications*, New Orleans, LA, February 1999, pages 90-100.
- [PRD01] C.E. Perkins, E.M. Royer, S.R. Das, and J. Broch, "Ad hoc On-Demand Distance Vector (AODV) Routing," *Internet Draft*, draft-ietf-manet-aodv-08.txt, March 2001.
- [PW93] U.W. Pooch and J.A. Wall, *Discrete Event Simulation, A Practical Approach*, CRC Press, 1993.
- [Rap95] T.S. Rappaport, *Wireless Communication: Principle and Practice*, Prentice Hall, Oct. 1995.
- [RM99] V. Rodoplu and T.H. Meng, "Minimum Energy Mobile Wireless Networks," *IEEE Journal on Selected Areas in Communications*, Vol. 17, No 8, Aug. 1999, pp. 1333-1344.
- [RMM01] E.M. Royer, P.M. Melliar-Smith, and L.E. Moser, "An Analysis of the Optimum Node Density for Ad Hoc Mobile Networks," *Proceedings of ICC 2001*, Helsinki, Finland, June 2001.
- [RP99] E.M. Royer and C.E. Perkins, "Multicast Operation of Ad-hoc On-demand Distance Vector Routing Protocol," *Proceedings of MobiCom 1999*, Seattle, WA, August 1999, pp. 207-218.
- [RST91] T.S. Rappaport, S.Y. Seidel, and K. Takamizawa, "Statistical Channel Impulse Response Models for Factory and Open Plan Building Radio Communication System Design," *IEEE Transactions on Communications*, Vol. COM-39, No. 5, May 1991.
- [RT99] E.M. Royer and C.K. Toh, "A Review of Current Routing Protocols for Ad-Hoc Mobile Wireless Networks," *IEEE Personal Communications Magazine*, April 1999, pp. 46-55.
- [SG99] W. Su and M. Gerla, "IPv6 Flow Handoff In Ad Hoc Wireless Networks Using Mobility Prediction," *Proceedings of IEEE GLOBECOM'99*, Rio de Janeiro, Brazil, Dec. 1999, pp. 271-275.
- [SL02] I. Stojmenovic and X. Lin, "Loop-free hybrid single-path/flooding routing algorithms with guaranteed delivery for wireless networks," *IEEE Transactions on Parallel and Distributed Systems*, to appear.
- [SPG97] S. Shenker, C. Partridge, and R. Guerin, "Specification of Guaranteed Quality of Service," *IETF RFC 2212*, Sept. 1997.
- [SSL97] A.R. Shahani, D.K. Schaeffer, and T.H. Lee, "A 12mW Wide Dynamic Range CMOS Front-end for a Portable GPS Receiver," *Proceedings of IEEE International Solid-state Circuits Conference*, Vol. 40, Feb. 1997, pp. 368-369.

- [ST92] H. Suzuki and F.A. Tobagi, "Fast Bandwidth Reservation Scheme with Multi-link & Multi-path routing in ATM networks," *Proceedings of IEEE INFOCOM 1992*, Florence, Italy, May 1992.
- [SWR98] S. Singh, M. Woo, and C.S. Raghavendra, "Power-Aware Routing in Mobile Ad Hoc Networks," *Proceedings of MobiCom 1998*, Dallas Texas, October 1998, pp. 181-190.
- [TB00] D. Tang and M. Baker, "Analysis of a Local-Area Wireless Network," *Proceedings of MobiCom 2000*, Boston, MA, August 2000, pp. 1-10.
- [Toh96] C.K. Toh, "A Novel Distributed Routing Protocol to Support Ad Hoc Mobile Computing," *Proceedings of 15th IEEE Annual International Conference on Computers and Communications*, Phoenix, 1996, pages 480-486.
- [Toh97] C.K. Toh, "Associativity-Based Routing for Ad-hoc Mobile Networks," *Wireless Personal Communications Journal*, vol 4, no 2, March 1997, pp. 103-139.
- [WG01] MANET Working Group, <http://www.ietf.org/html.charters/manet-charter.html>, 2001
- [WH00] K. Wu and J. Harms, "Location Trace Aided Routing in Mobile Ad Hoc Networks," *Proceedings of the 9th Int. Conf. On Computer Communications and Networks (IC3N)*, Las Vegas, NV, Oct. 2000, pp. 354-359.
- [WH01] K. Wu and J. Harms, "On-demand Multipath Routing for Mobile Ad Hoc Networks," *Proceedings of 4th European Personal Mobile Communication Conference (EPMCC)*, Vienna, Austria, Feb. 2001.
- [WH01-2] K. Wu and J. Harms, "Performance Study of a Multipath Routing Method for Wireless Mobile Ad Hoc Networks," *Proceedings of 9th International Symposium on Modeling, Analysis and Simulation (MAS-COTS '01)*, Cincinnati, Ohio, Aug. 2001, pp. 99-107.
- [WH01-3] K. Wu and J. Harms, "Load-Sensitive Routing for Mobile Ad Hoc Networks," *Proceedings of the 7th Int. Conf. On Computer Communications and Networks (IC3N)*, Phoenix, Arizona, Oct. 2001, pp. 540-546.
- [WH01-4] K. Wu and J. Harm, "QoS Support in Mobile Ad Hoc Networks," *Crossing Boundaries- the GSA Journal of University of Alberta*, Vol. 1, No. 1, Nov. 2001, pp.92-106.
- [WHE01] K. Wu, J. Harms, and E.S. Elmallah, "Profile-Based Protocols in Wireless Mobile Ad Hoc Networks," *Proceedings of IEEE International Conference on Local Computer Networks (LCN 01) and Workshop on Wireless Local Networks (WLN 01)*, Tampa, Florida, Nov. 2001, pp. 568- 575.
- [WH02] K. Wu and J. Harms, "Multipath Routing for Mobile Ad Hoc Networks," *IEEE ComSoc/KICS Journal of Communications and Networks, Special Issue on Innovations in Ad Hoc Mobile Pervasive Networks*, Vol. 4, No. 1, March 2002, pp. 48-58.
- [Wro97] J. Wroclawski, "Specification of the Controlled-Load Network Element Service," *IETF RFC 2211*, Sept. 1997.

- [WTK98] C.W. Wu, Y.C. Tay, and C.K. Toh, "Ad hoc Multicast Routing protocol utilizing Increasing Id-numberS (AMRIS) Functional Specification," *Internet Draft*, Nov. 1998.
- [YK01] S.S. Yau and F. Karim, "Context-Sensitive Object Request Broker for Ubiquitous Computing Environments," *proceedings of the 8th IEEE Workshop on Future Trends of Distributed Computing Systems (FT-DCS 2001)*, Bologna, Italy, Oct. 2001.
- [ZBG98] X. Zeng, R. Bagrodia, and M. Gerla, "GloMoSim: a Library for Parallel Simulation of Large-Scale Wireless Networks," *Proceedings of PADS'98*, Banff, Canada, May 1998.
- [ZS00] H. Zhou and S. Singh, "Content Based Multicast (CBM) in Ad Hoc Networks," *Proceedings of IEEE/ACM MobiHoc 2000*, Boston, MA, August 2000, pp. 51-60.