An Improved Agent-Based Routing Protocol for Mobile Ad-Hoc Networks

by

Wenwei Yue

Submitted in partial fulfillment of the requirements
for the degree of
Master of Computer Science

at

Dalhousie University
Halifax, Nova Scotia
July, 2004

DALHOUSIE UNIVERSITY

To comply with the Canadian Privacy Act the National Library of Canada has requested that the following pages be removed from this copy of the thesis:

Preliminary Pages
- Examiners Signature Page (pii)
- Dalhousie Library Copyright Agreement (piii)

Appendices
- Copyright Releases (if applicable)

Dedicated to my grandmothers

Table of Contents

List of Tables

List of Figures

List of Abbreviations and Symbols

AODV:	Ad-hoc On-Demand Distance Vector
CBR:	Constant Bit Rat
CGSR:	Clusterhead Gateway Switch Routing
CSMA/CA:	Carrier Sense Multiple Access with Collision Avoidance
CTS:	Clear To Send
DARPA:	Defense Advance Research Project Agency
DSDV:	Destination Sequenced Distance Vector
DSR:	Dynamic Source Routing Protocol
GPS:	Global Positioning System
IEEE:	Institute of Electrical and Electronics Engineers
LMAR:	Local Mobile Agent Routing protocol
MAC:	Medium Access Control
MANET:	Mobile Ad-hoc Networks
MAR:	Mobile Agent Routing Protocol
NAM:	Network Animator
NS:	Network Simulator
PDA:	Personal Digital Assistant
RTS:	Request To Send
UDP:	User Datagram Protocol
VINT:	Virtual InterNetwork Testbed
WLAN:	Wireless Local Area Network
WRP:	Wireless Routing Protocol
ZRP:	Zone Routing Protocol

Acknowledgements

I would like to express my sincere thanks to my supervisors, Dr. Ernst W. Grundke and Dr. Nur Zincir-Heywood, for their guidance throughout my thesis study. I would also like to thank Donald Morrison, Nick Pilon and other colleagues in the Ant Project group at the Faculty of Computer Science, and Yan Zhou, for their contributions and advice on the protocol development and simulation experiments.

Special thanks go to my wife, Linhua, and my lovely daughter, FeiFei, for their great support, understanding, and encouragement.

Abstract

A mobile ad-hoc network is an infrastructureless network temporarily formed by wireless mobile nodes. A mobile node can act as a router forwarding data packets between other nodes. The dynamic changes of network topology have been a challenge for routing data packets across the ad-hoc networks.

Inspired by the cooperative effort of ants on finding optimal path to food sources, researchers have used the social ant metaphor to develop mobile agent routing protocols. The agent population is an important parameter in an agent-based protocol. In the Mobile Agent Routing Protocol (MAR), a global registry of mobile agents was used; however, we believe that it is not realistic to apply a global data structure in an ad-hoc network.

In this thesis, we introduce an improved agent-based ad-hoc routing protocol, LMAR, to remove the global registry and use only local information of a node to build the routing table. Extensive experiments were carried out using the simulator NS-2 to compare the network performance of LMAR with other three ad-hoc routing protocols, MAR, AODV and DSDV. The simulation results of the t-Test show that the LMAR has almost the same performance as the MAR.

Chapter 1 Introduction

1.1 Ad-hoc Networks

Mobile Ad-hoc Networks (MANET) consist of many wireless mobile nodes without any existing communication infrastructure or centralized administration node. The transmission range of wireless network interfaces is limited, so a data packet may need to travel multiple hops from its source to the destination. Each mobile node acts not only as a host, but also as a router, forwarding data packets between other nodes in the network. The mobile nodes can build their own ad-hoc network on the fly and dynamically establish routing among them.

Mobile Ad-hoc networks have been the research interests of many institutes and corporations [Perk2001, Toh2002]. Applications of ad-hoc networking can be used in many areas. In military and emergency networks, the use of ad-hoc networks supports robust and survivable communications in a highly dynamic battlefield environment. There are two deployed MANET in the US military [Perk2001]: the U.S. Army's Advanced Warfighting Experiment, which built a tactical internet comprising thousands of vehicular and man-packed radios; the DARPA Global Mobile (GloMo) Information Systems program, which aimed to provide connectivity any time, anywhere in mobile devices. Mobile ad-hoc networking can also be used for extending the coverage of cellular or WLAN networks. MeshNetworks [Mesh2004] has developed commercial mobile ad-hoc networking products and created a city-wide wireless broadband network for police, public works and other municipal agencies in Medford, Oregon. Other

examples include students with laptop computers sharing information during a meeting, and people setting up ubiquitous wireless networking of home electronics and appliances [Perk2001].

1.2 Ad-hoc Network Routing

The nodes in an ad-hoc network are mobile. The network topology may change continuously as the nodes move or adjust their transmission and reception parameters. Mobile ad-hoc networks have several characteristics that make the routing protocols hard to design [Cors1999].

1. Dynamic topologies: The free movement of mobile nodes may change the network topology randomly and rapidly.
2. Limited link bandwidth: The transmission rate is often affected by multiple access, fading, noise, and interference, etc.
3. Energy-constrained operation: Mobile nodes usually depend on batteries or other exhaustible resources.
4. Limited physical security: Wireless signals and lack of central support make ad-hoc networks more vulnerable.

Several routing protocols have been proposed for ad-hoc networks. These routing algorithms can usually be divided into two categories [Roye1999]: proactive routing (or table-driven) and reactive routing (or on-demand). Proactive protocols keep routing tables all the time, which creates extra overhead due to periodical routing updates. Reactive protocols find routes on request, which may increase the latency. Proactive protocols maintain routes to every node in the network, while reactive protocols retain only active routes.

Inspired by the cooperative effort and indirect inter-ant communication of ants in finding optimal path to food sources, researchers have used this social ant metaphor to solve network problems. Early work was done by Appleby and Steward in 1994 to use mobile software agents for control in telecommunication networks [Appl1994]. Mobile agents were first used for adaptive routing in wired network in 1998 by Di Caro and Dorigo [DiCa1997]. Researchers at the MIT Media Lab proposed mobile software agents for ad-hoc routing [Mina1999]. Another intelligent agent routing protocol for mobile ad-hoc networks was introduced in [Zhou2003].

1.3 Objectives and Organization

The Mobile Agent Routing Protocol (MAR) [Zhou2003] uses mobile agents to explore and collect the network routing information. This information is exchanged indirectly at the mobile nodes and used to build routing tables. Because the population of agents may decrease for several reasons in the simulation, MAR defines a global registry system to control the exact number of agents in the network. However, the ad-hoc network is infrastructureless, so we need to give up this idea of central data structure.

The objectives of this thesis are to remove the global registry and design a new ad-hoc routing algorithm by using only local information of nodes to build routing tables. We compare this new Local Mobile Agent Routing protocol, LMAR, with three other ad-hoc routing protocols: MAR, Destination-Sequenced Distance-Vector (DSDV) [Joha1999], and Ad-hoc On-Demand Distance Vector (AODV) [Perk2001]. MAR is an agent-based routing, and AODV is a reactive routing, while DSDV is a proactive routing. We carried

out extensive simulations to study the network performance of these four mobile ad-hoc routing protocols. From the simulation results we can better understand the indirect inter-agent communication of mobile agents, and make further improvement to our agent-based ad-hoc routing protocol.

The rest of this thesis is organized as follows: Chapter 2 describes the existing ad-hoc routing protocols: reactive routing, proactive routing and agent-based approach. Chapter 3 outlines the LMAR algorithm and a proof of its correctness. The simulation model and performance measurements are introduced in Chapter 4, and then Chapter 5 presents our simulation results and analysis. Finally, Chapter 6 shows the conclusion of our project and discusses future works.

Chapter 2 Literature Review

2.1 Introduction

Numerous mobile ad-hoc routing protocols have been proposed to the IETF (Internet Engineering Task Force) MANET working group. These routing protocols are typically divided into two categories [Roye1999]: proactive routing and reactive routing.

Proactive or table-driven protocols require each node to maintain one or more tables containing routing information to every destination node in the ad-hoc network. All nodes periodically refresh or update the routing information in order to maintain an up-to-date routing table. When the network topology changes, the nodes propagate update messages throughout the network. The advantage of the proactive approach is that once a route is formed, the data packets can use that routing information immediately. However, the proactive protocols have a huge routing overhead due to the periodical routing table exchange. Typical proactive protocols include Destination-Sequenced Distance-Vector (DSDV) [Joha1999], Cluster-head Gateway Switch Routing Protocol (CGSR) [Lars1998], and Wireless Routing Protocol (WRP) [Cele2002]. DSDV protocol is studied and simulated in our thesis.

Reactive or source-initiated on-demand routing protocols do not periodically update the routing information, and the routes to the destination are created only when required. Reactive protocols can greatly reduce the routing overhead and save node resources like memory, battery power and CPU time. The disadvantages of the reactive protocols are

that there are initial delays to find necessary routes. Typical reactive protocols include Ad-hoc On-Demand Distance Vector Routing (AODV) [Perk2001], Dynamic Source Routing (DSR) [John2002], and Temporally Ordered Routing Algorithm (TORA) [Broc1998]. The AODV protocol is studied and simulated.

Zone Routing Protocol (ZRP) [Hass2001] combines and uses the advantages of both reactive and proactive routing protocols. It divides the network into several routing zones. A proactive protocol operates inside the routing zone and learns all the possible routes, so routes can be found very quickly within the zone; a reactive is used for finding routes between different routing zones, so the control overhead can be limited. However, the zone diameter is not adjusted by the network itself, but instead it is set by the network administrator or the manufacturer. A change in network topology by node movement can affect several routing zones, and adjust the routing zones.

2.2 Proactive Routing: DSDV

Destination Sequence Distance Vector (DSDV) [Joha1999] is based on the classical Bellman-Ford distance vector routing algorithm [Kuro2002] in which each node maintains a distance table including a column for each neighbour node and a row for each destination node in the network. A source node X's table entry to destination Y through its neighbour Z is the sum of the direct one-hop link between X and Z, plus Z's currently known minimum-cost path from itself (Z) to Y. In DSDV, each node has a routing table containing the next hop information for all reachable destinations. Because DSDV requires each node periodically to broadcast routing updates, it needs some time for the routing protocol to converge for building up a useable routing table. This converge time

will probably be long if the topology of an ad-hoc network changes dynamically. DSDV uses two types of periodic broadcasts to update information: full and incremental dump. The full dump broadcasts all routing information, while the incremental dump only broadcasts the difference since the last dump. Trigged updates are also added to this protocol to take care of the topology changes between broadcasts.

DSDV uses a sequence number to show the freshness of a route. The routes with higher sequence number are preferred. If the routes have the same sequence number, the route with low hop count will be selected. If a node *A* detects that a route to a destination *D* has broken, the node *A* will advertise its route to node *D* with an infinite hop count and an increased sequence number.

2.3 Reactive Routing: AODV

The Ad-hoc On-Demand Distance Vector (AODV) [Perk2001] is a reactive distance vector protocol and one of the most important ad-hoc protocols. AODV can quickly create new routes to the destination node when needed, and does not maintain routes to inactive nodes. AODV uses destination sequence numbers in a way similar to DSDV to avoid loops in the routing path.

When a node requires a route to the destination, it broadcasts a *Route Request (RREQ)* message to all its neighbors which contains latest known sequence number for that destination. The *RREQ* message is flooded until it reaches the destination. Each node receiving the *RREQ* message creates a reverse route to the source. After finding the destination, a *Route Reply* message which includes number of hops and its sequence

number is sent back to the source along the temporary reverse route. Each node receiving the reply message creates a routing table entry to the destination.

AODV discovers nodes in the neighbourhood using local "hello" messages. If a node does not receive "hello" messages from a particular neighbour, the communication link between these nodes may be broken. A link failure message is sent to the affected set of nodes. If an existing routing entry is unused for a specified time period, the invalid entry will be removed from the node, and a route error message will be send to all affected set of nodes. AODV also uses the information from the MAC layer to detect the link failures, which results in quicker failure detection and faster routing response.

2.4 Ant Metaphors

Insects that live in colonies like ants are self-organized and can make a cooperative effort toward food searching or path optimization. When ants are walking from their nest to a food source, they deposit a chemical on the ground, and meanwhile follow the chemical previously left by others. Ants have been shown to be able to find shortest paths using only information to pheromone (chemical) trail deposited by other ants [Dica1997].

Algorithms which were inspired by ants' indirect communication in finding the shortest paths have been successfully applied to control in distributed telecommunication network systems [Appl1994] and adaptive learning of routing tables in wired data communications networks [DiCa1998]. In an ant-based algorithm, a set of artificial ants collectively solve the problem through the cooperative effort of indirect communication of information.

Mobile agents are simple packets carrying data and exploring the network. They leave information data behind on nodes that they have visited, so other agents can use it. This indirect inter-agent communication is a suitable technique for distributed network systems. Some of the benefits of using mobile agents discussed in [Lang1999] are reduction of the control overhead and network latency, dynamic adaptation to the execution environment and robustness and fault-tolerance for distributed systems.

2.5 Mobile Software Agent

Computer networks are becoming more heterogeneous and dynamic: more different devices are connected to the networks and mobile devices may move from place to place. Conventional centralized approaches are inappropriate, unsuitable to the computer networks. The mobile software agents designed at the MIT Media Lab [Kram1999] that cooperate to collect and distribute connectivity information for a dynamic wireless network have important properties:

- Each agent is independent of all others with its code and data.
- An agent can move easily around the network.
- The size of an agent is small in order to reduce the transmission overhead
- Agents should cooperate to complete tasks. They can exchange information indirectly by reading from and writing to a node.
- Agents can use the resources and information of visiting nodes.

Kramer et al. designed a simple wireless network and used a simulator to study the network performance of mobile software agents [Kram1999]. There are about 250 nodes with low-power, short-range radio transceivers which can move around randomly and

slowly through a simulated space. The network topology is quite dynamic as the wireless links are established and broken when the nodes move in and out of range of each other. The routing agent keeps a history list of where it has been. When a node receives a mobile agent, it uses the information in the agent's history to update the routing table with possible best routes. The routing agents communicate with each other indirectly by writing the information to the routing tables of mobile nodes, but do not read information from the mobile nodes.

The simulation at the MIT Media Lab [Kram1999] shows that a certain number of agents can cooperatively build and maintain reasonable routing tables even in constant changing environments. The total number of agents and the history size of agent are two important parameters. The system always works better with more routing agents and longer history size; however, additional agents and increased history size will cost more network bandwidth.

2.6 Mobile Agent Routing

2.6.1 Introduction

The Mobile Agent Routing protocol (MAR) [Zhou2003] is a routing protocol based on the mobile software agent concept from MIT Media Lab. In the MAR protocol, a "thorough stigmergy" method was used to make the cooperation of agents more efficiently: the mobile agents not only write routing information to the nodes they visit, but also read routing table information left by other mobile agents. This scheme allows a

mobile agent to acquire the routing information about the mobile nodes it has not visited yet, so it is important for a new mobile agent to quickly obtain network information.

The work of the MIT Media Lab [Nels1999] showed that the performance of a dynamic network depend on the number of mobile agents in the system. They used 100 long-lived mobile agents with history size 25 in an ad-hoc network of 250 mobile nodes. In a realistic environment, the number of agents may decrease for at least two reasons [Zhou2003]. First, if a node holding a mobile agent moves out of the transmission range of all other nodes for a long time, this agent will become isolated and cannot provide useful routing information. Second, mobile agents could be dropped during transmission from one node to another due to MAC layer collisions or packet errors. The loss of agents will eventually degrade packet routing performance, especially in highly dynamic circumstances.

2.6.2 Global Registry in MAR

MAR introduced a global registry to control the exact number of mobile agents in the system. The global registry records all current mobile agents and its expiry time. A new agent will be generated and registered with its expiry time in the registry until there are enough agents in the network. In an ad-hoc network with N nodes, the following procedure is to create and register M mobile agents in the global registry [Zhou2003].

1. When a simulation starts, mobile nodes are created from node number *0* to node number *N-1*. These nodes then generate mobile agents and register them with expiry time in the registry until there are M agents.

2. When a node receives a mobile agent, it updates the agent's expiry time. Each node checks the registry periodically and eliminates any expired agents.
3. Nodes will make new agents if the number of agents in the global registry falls below *M* due to any reason. A node can generate and register only one mobile agent at any time.

MAR uses "hello" messages to handle link change the same as AODV and DSDV do. Each node in MAR periodically broadcast a "hello" message to its immediate neighbours (one hop away). When a node receives a "hello" message from its neighbour, it adds this node to its neighbour list. If a node does not hear a "hello" message from a neighbour for a time period, it considers that node is unreachable and then removes that node from its neighbour list. All entries in its routing table using that neighbour as the next hop are also deleted. The interval of "hello" message is an important parameter in MAR. In a network with a dynamic topology, too long an interval cannot detect link changes, while too short an interval causes routing overhead and congestion of the network.

2.6.3 MAR Agent Format

MAR agent has a simple structure.

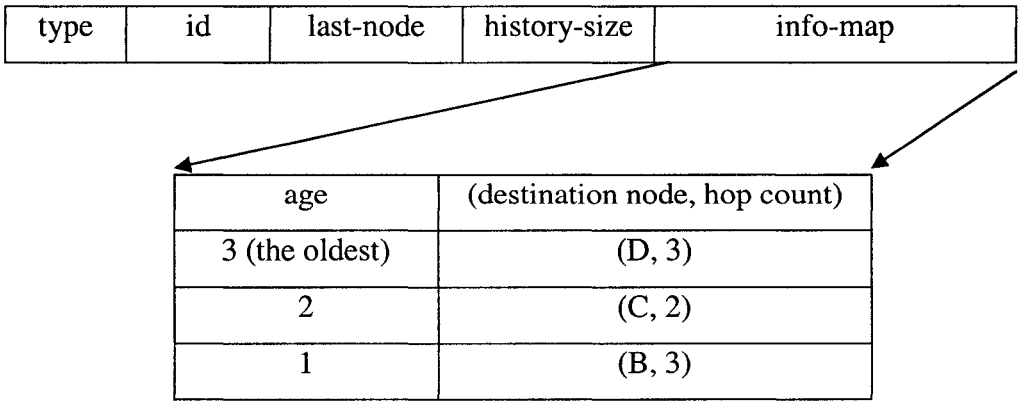

Figure 2.1: MAR agent format [Zhou2003]

In Figure 2.1, the "type" field indicates it is a MAR agent or a "hello" message. The "id" is used to distinguish every agent packet, and the "last-node" records the previous node just visited. The "history-size" gives the size of the "info-map" which stores the routing information collected by the agent. In the "info-map" data structure, the "age" field keeps the sequence of the nodes visited. The larger the sequence value, the older the information is.

Chapter 3 The LMAR Protocol

MAR uses the global registry to control the number of mobile agents in an ad-hoc network. However, as there is no master node in a mobile ad-hoc network, it is not realistic to apply a global data registry except for simulation purposes. The protocol code in a mobile node cannot use global data. A mobile node can only know the local circumstances, such as its neighbours, incoming and outgoing traffic, battery level, moving speed and direction, and position if it is equipped with GPS equipment. By using only local information of node, an ad-hoc routing protocol can become feasible.

Even if a master node were available, a protocol using global information would not scale with network size.

3.1 Algorithm

To avoid the global registry in the MAR routing protocol, a novel scheme is proposed in order to control the average number of mobile agents in wireless ad-hoc networks. The basic concept is that, in an ad-hoc network with N mobile nodes, if a node can create a new mobile agent every C seconds with a fixed lifetime of T seconds, then after a relative long period compared to the agent's life time, the average number of agents (n) is

$$n = NT / C \tag{1}$$

where n = average number of mobile agents

N = number of mobile nodes

T = expiry time of mobile agents

C = time interval to create a new agent

The proof of this formula will be given in the Section 3.2.

The following are our basic steps to control the average number of agents in an ad-hoc network with N nodes:

1. When the NS-2 simulator starts, nodes are initially created from node number *0* to node number *N-1*. Node number *0* first creates a mobile agent and sends it out, and then the node number *1*, and all other nodes in sequence.
2. Each node repeats the generation of an agent at a time interval C. For example, the node number *0* produces its ants at time *0*, C, $2C$ second, and the node number *1* generates ants at its time clock C/N, $C/N+C$, $C/N+2C$ seconds, and so on.
3. All agents have a life-time of T seconds. When a node receives a mobile agent, it will check the agent's life time to decide whether to drop the agent or not.

To implement the LMAR protocol in the NS-2 simulator, we made a little change to MAR format. A new field called "ant-lifetime" was added to create a new LMAR agent format. The "ant-lifetime" is used to store the expiry time of a mobile agent, and is the current time plus agent's expiry time T. The new data format of the mobile agent for LMAR is shown in Figure 3.1.

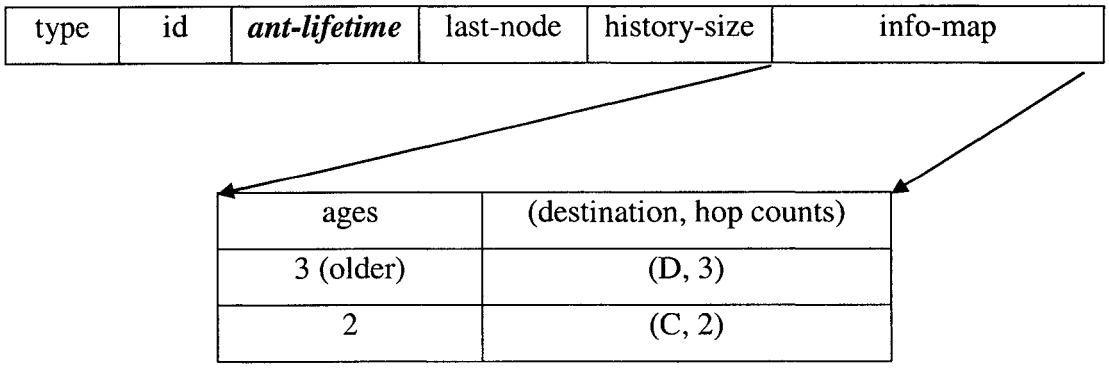

Figure 3.1: LMAR agent data structure

In our program, each node uses its own timer to decide when to create a new agent and when to delete an expired one. In the NS-2 simulator environment, all mobile nodes use the timer of the NS-2 system, so the time clocks at all mobile nodes are synchronized. However, in the real situation, when our agent-based routing protocol is installed in laptop and palmtop computers to form an ad-hoc network, these mobile devices have their own time clocks. These time clocks may not be synchronized which would make the calculation of agent's lifetime incorrect.

One solution to this issue is to set the value of "ant_lifetime" to the expiry time T, and reduce the agent's life-time in each mobile node. When the life-time of an agent in a node becomes zero, the agent is destroyed. The latency of an agent packet in a network usually includes the propagation delay, the transmission delay, the processing delay and the queuing delay. For a wireless transmission range of about 100 meters, the propagation delay at the speed of light ($3*10^8$m/s) is about 0.33 microseconds; for an ant packet of about 100 bytes, the transmission delay in a 2Mbps bandwidth channel is about 400 microseconds. Thus, the latency will mainly consist of processing delay and queuing delay. The processing delay can be measured from the ant arrived time and the time the ant is put into the output queue. When the network is sometime congested, the agent may stay in the queue of a node's MAC layer for a while. If we can add a new time field in the data frame structure of MAC layer to record the out-queue time of the agent, the ant's queuing delay can be calculated by subtracting the agent's enqueuing time from the dequeuing time. When an ant's lift time is less than 0, it is destroyed. From the above

analysis, we can believe that the use of NS-2 timer in our simulation is not a critical issue in the implementation of the LMAR protocol.

3.2 Proof of Equation (1)

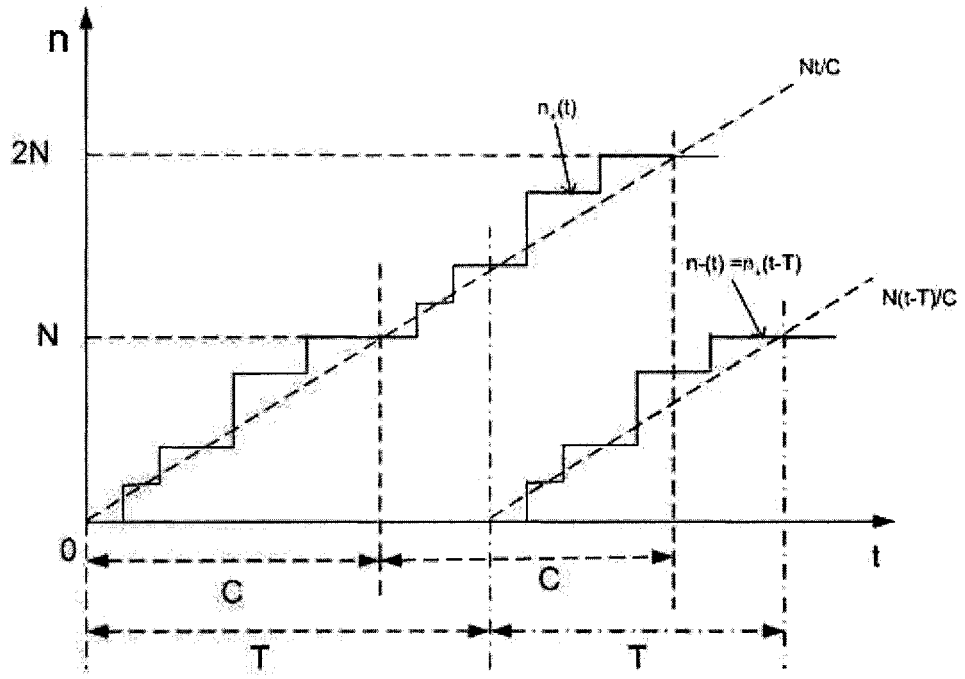

Figure 3.2: The number of agent *n* at time *t*

Suppose that there is an ad-hoc network consisting of N mobile nodes, and each node creates a new ant with life-time of T seconds every C seconds. The Figure 3.2 shows the number of mobile agents in the network at time t. The ant creation function, $n_+(t)$, is the total number of created ants in the network up to time t seconds, and the ant destruction function, $n\_(t)$, is the total number of destroyed ants up to time t seconds. Because we can create a total number of N ants during C seconds, and every node repeats its creation of ants after C seconds interval, we have

$$n_+(t+C) = n_+(t) + N, \tag{2}$$

On the other hand, any agent will be destroyed from the network when its lift age passes T seconds, so the ant destruction function, $n\_(t)$, is the same as the ants creation function, $n_+(t)$, but with a T second delay,

$$n\_(t) = n_+(t-T) .$$

If we consider the ant creation function, $n_+(t)$, consisting of a linear function, Nt/C, and a difference between $n_+(t)$ and NT/C, $g(t)$, we can have

$$n_+(t) = Nt / C + g(t) . \tag{3}$$

Then, at time $t+C$ second, the ant creation function $n_+(t+C)$ becomes

$$n_+(t+C) = N(t+C)/C + g(t + C) = (Nt/C) + N + g(t+C) . \tag{4}$$

From the Equations (2) and (3), we also have

$$n_+(t+C) = n_+(t) + N = Nt/C + g(t) + N . \tag{5}$$

As the Equations (4) and (5) are equal, we can obtain

$$g(t+C) = g(t). \tag{6}$$

The average number of ants in the network after relatively long time is the total number of created ants minus the total number of expired ants during that period. The following is the derivation of equation (1).

$$n_{ave} = 1/C \int_{t'}^{t'+C} [n_+(t) - n_-(t)]dt$$

$$= \frac{1}{C}\int_{t'}^{t'+C} [n_+(t) - n_+(t-T)]dt$$

$$= \frac{1}{C}\int_{t'}^{t'+C} [\frac{Nt}{C} + g(t) - (\frac{N(t-T)}{C} + g(t-T))]dt$$

$$= \frac{1}{C}\int_{t'}^{t'+C} \frac{NT}{C}dt + \frac{1}{C}\int_{t'}^{t'+C} (g(t) - g(t-T))dt$$

$$= \frac{NT}{C} + \frac{1}{C}(\int_{t'}^{t'+C} g(t)dt - \int_{t'}^{t'+C} g(t-T)dt)$$

If we let $r = t - T$ and $r' = t' - T$, so $dt = d(r + T) = dr$, then we can have

$$\int_{t'}^{t'+C} g(t-T)dt = \int_{t'-T}^{t'+C-T} g(r)dr = \int_{r'}^{r'+C} g(r)dr .$$

The average number of agents in the network is:

$$n_{ave} = \frac{NT}{C} + \frac{1}{C}(\int_{t'}^{t'+C} g(t)dt - \int_{t'}^{t'+C} g(t-T)dt)$$

$$= \frac{NT}{C} + \frac{1}{C}(\int_{t'}^{t'+C} g(t)dt - \int_{r'}^{r'+C} g(r)dr)$$

$$= \frac{NT}{C}$$

This result points out that the average population of agents is determined by the agent's expiry time and creation interval, and is proportioned to the total number of nodes. We can use different mobile agent creation functions, such as a fixed creation interval function, or a random creation function, or some mathematical distribution, like Poisson distribution, for the simulation implementations of mobile ad-hoc networks.

Chapter 4 Simulation Model

NS-2 [NS2003] is an object-oriented network simulator for research and education. It provides support for protocol design, network traffic studies, and network protocol comparison in wired and wireless world. We chose the NS-2 network simulator to evaluate the performance of our improved agent-based routing protocol, LMAR, and three other ad-hoc network protocols, MAR, DSDV, AODV.

4. 1 NS-2 Simulator Introduction

NS-2 simulator began as a network simulator for studying the dynamic behaviour of flow and congestion control in packet-switched data networks. Then, it was funded by DARPA (Defense Advance Research Project Agency) through the VINT [VINT2003] research project involving Xerox PARC, UC Berkeley, LBNL (Lawrence Berkeley National Laboratory) and USC/ISI (University of Southern California/Information Science Institute). Because the rapid expansion of the Internet and multimedia applications have made the current protocol design and engineering practices inadequate, these protocols must be changed in order to produce robust and evolvable network technologies in the future.

NS-2 simulation software is free and open source. The architecture of NS-2 provides a framework for composing simulation modules to allow independent development of new protocols and application by researchers. However, since NS-2 is still an on-going project

and there is no user-friendly documentation about its complex system, it is quite difficult to develop and integrate new models and protocols into the NS-2 simulator.

NS-2 is written in the C++ and OTcl languages. The detailed protocol implementation uses C++ to efficiently manipulate bytes, packet headers and algorithms. The OTcl language is used as a command and configuration interface to change the simulation parameters quickly and interactively.

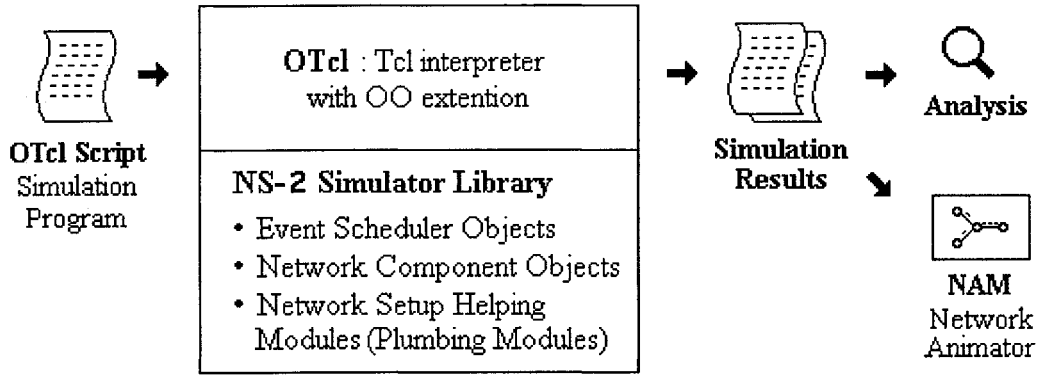

Figure 4.1: Simplified User's View of NS-2 [Chun2003]

The Figure 4.1 [Chun2003] shows that NS-2 is an OTcl script interpreter that has simulation event scheduler objects, network component libraries, and network setup module libraries. To setup and run a simulation, we should write an OTcl script program that initiates an event scheduler, sets up the network topology using the network component objects, and defines all traffic flows in the network. When a simulation is finished, the simulation results are output to trace files that contain detailed simulation data which can be analyzed to obtain the network performance of protocols, or can be demonstrated using a graphical simulation display tool called Network Animator (NAM).

The NS-2 simulator can be built and run on multiple platforms, including FreeBSD, Linux, Sun Solaris, Microsoft Windows, and Apple Mac. It is now used by more than ten thousand users in over one thousand institutes in fifty countries [NS2003]. We built our NS-2 version 2.26 on the locutus.cs.dal.ca and flame.cs.dal.ca Sun servers at the Faculty of Computer Science of Dalhousie University. Locutus is a Sun Enterprise 4500 server for graduate study and research with eight 400MHz Sun Sparc processors and 3GB RAM. Flame is a Sun Fire 4800 server with eight 900MHz Sun UltraSPARC-III processors, 32GB RAM, and a dedicated 100 Mbps full-duplex network connection.

The wireless model in NS-2 allows simulation of pure wireless LANs, multi-hop ad-hoc networks, Mobile IP, satellite networks, directed diffusion networks. Four ad-hoc routing protocols, DSDV, DSR, AODV and TORA, are currently implemented for mobile networks in NS-2.26. There are also other contributed codes that are maintained by users and that have not been incorporated into the NS-2 distributions, which including routing modules like Zone Routing Protocol and Dynamic Link State Routing protocol, and mobility and wireless modules like BlueHoc and Mobile IPv6 in large wide area networks.[Fall2003]

4.2 Mobile Ad-hoc Model in NS-2

Mobilenode is the core of the wireless model in the NS-2 system. A mobilenode [NS2003] can move within a given region, and receive and transmit wireless signals through wireless channels. The Figure 4.2 shows that the network stack for a mobilenode consists of a LL (link layer), an ARP module, an IFQ (interface priority queue), a MAC layer, a

NetIF (network interface), a Radio Propagation Model, and a wireless communication channel.

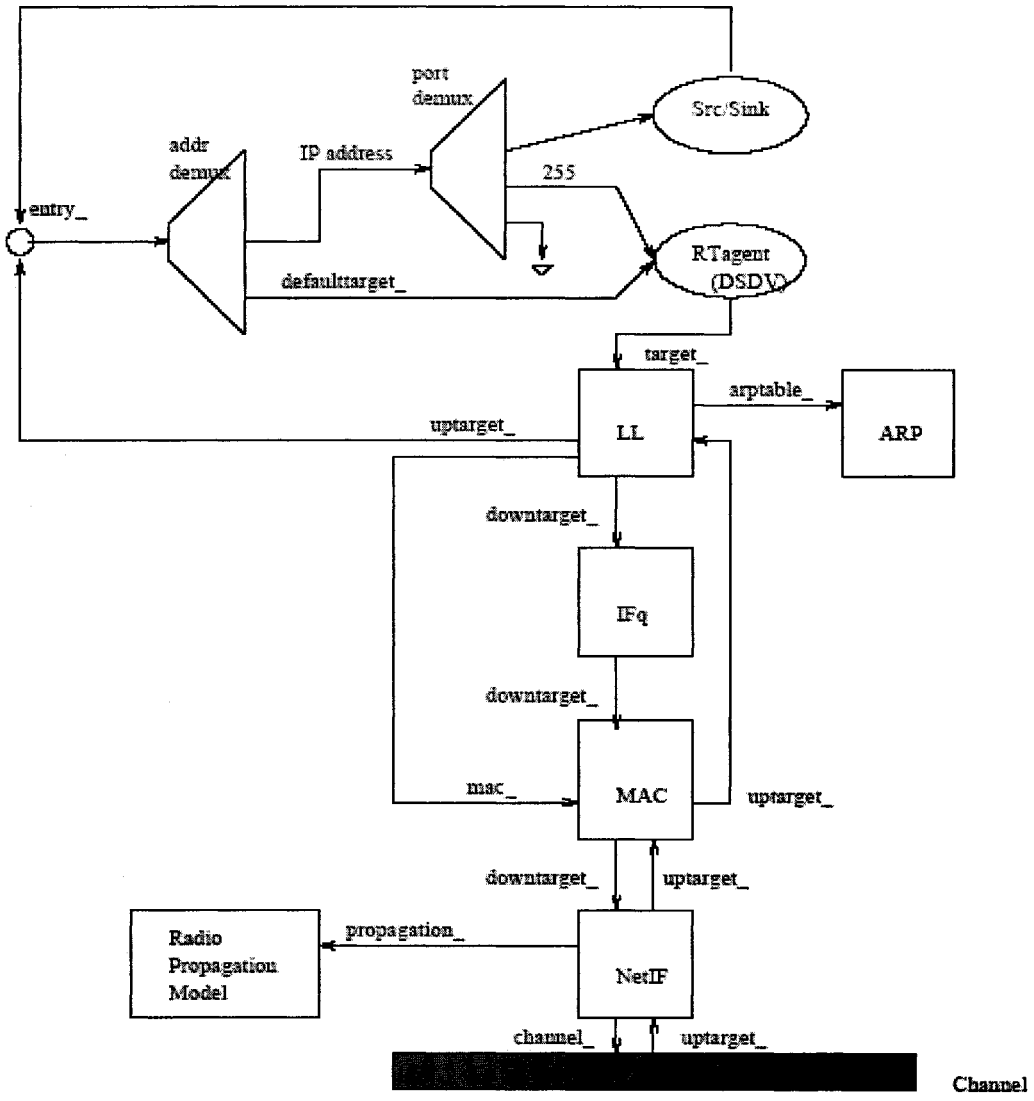

Figure 4.2: Schematic of a mobilenode in NS-2 [Fall2003]

The wireless channel layer is used to transmit data and control packets through air. A mobilenode uses an omni-directional antenna having unity gain, located 1.5 meters above the ground, and operation in the 1-2 GHz band. The radio propagation model uses Friss-

space attenuation ($1/r^2$) at near distances, and an approximation to Two Ray Ground reflection ($1/r^4$) at far distances.

Mobilenode uses the network interface layer to access the wireless shared media channel. For each transmitted packet, the interface records information related to the transmitting interface, like the transmission power and wavelength. The model employs the Lucent WaveLan DSSS (Direct Sequence Spread Spectrum) interface protocol.

The MAC layer in NS-2 uses the IEEE 802.11 Carrier Sense Multi-Access with Collision Avoidance (CSMA/CA) mechanism to handle collision detection, fragmentation and acknowledgement. A source node senses the medium before it sends. If the medium is free longer than a pre-defined time period, the source can send out packet directly; otherwise, it waits for medium to become free. If many nodes want to send after the waiting time, a collision occurs. Therefore, these stations use an exponential backup algorithm to resolve contention. When a node tries to access the medium, it first sends a Request To Send (RTS) message to notify other nodes that they must wait during the next transmission. When the destination node receives the RTS, it sends back a Clear to Send (CTS) message to allow the source node to send a data packet. This CTS message also prohibits data transmission by other nodes to the destination.

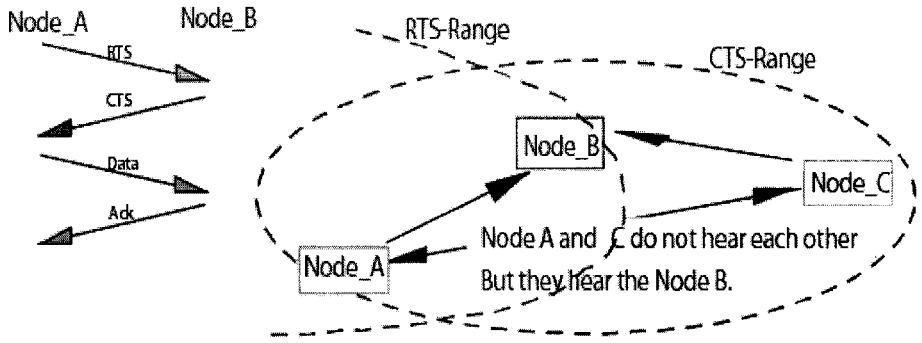

Figure 4.3: IEEE 802.11 four ways handshake [Diep1996]

When a data packet is received, an acknowledgement (ACK) will be send back. This four-way handshake pattern, RTS/CTS/DATA/ACK, solves the hidden station problem in wireless networks [Kuro2002, Stal2000]; see Figure 4.3.

4.3 Adding the new protocol into NS-2

To integrate our agent-based routing protocol into the NS-2 simulator, we have to make change in some of the C++ and OTcl source files, which include adding our new packet type and packet format into the packet header file, inserting the packet type into the C++ source file for tracing the output, and adding a Tcl procedure to create a new LMAR agent in the NS-2 simulator.

4.4 Running the NS-2 Simulation

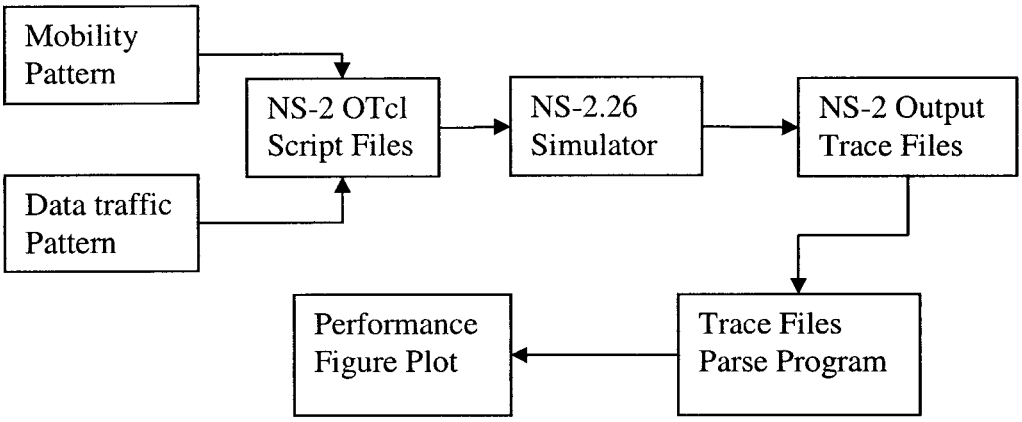

Figure 4.4: Simulation procedure

Figure 4.4 shows our procedure to run NS-2 simulation. We first generate a NS-2 OTcl script file, in which we define mobility pattern, data traffic pattern, network topology, ad-hoc routing protocol, radio propagation mode, and other parameters. Then, after running this script file in the simulator, we have output trace files which record all data packets that are dropped, received and sent by MAC layers or interface queues. These trace files are parsed to extract information needed to measure the protocol performance such as the packet delivery ratio, the average end-to-end delay and the average packet hop counts. A performance figure can be plotted later.

We ran our simulations at six moving speed scenarios. Each scenario was run twenty times with different nodal movement and data traffic pattern. In order to manage these script files and output trace files, three simulation tools from [Morr2003], pattern

coordinator, script coordinator and simulation coordinator, were used to improve our efficiency.

4.5 Performance Measurement

To evaluate the performance of the four ad-hoc routing protocols, we use the following performance measures.

- Packet Delivery Ratio: The ratio of all received packets at their destinations to all transmitted packets from the CBR sources. In the NS-2, all data packets are assigned unique identifiers when they leave the source MAC. We first calculate the packet delivery ratio for each run of twenty simulations at a particular moving speed, and then the average value and standard deviation are computed.
- Average End-to-End Delay (in seconds): This is the average delay time for a data packet travelling from its source to destination. The delay time of all successfully received packets is summed, and then the average delay time and standard deviation is calculated.
- Average packet hop counts: The average number of hop counts of a data packet from its source to the destination.

4.6 Configuration

4.6.1 Node Mobility

We performed all simulations in random scenarios. The node mobility is the maximum nodal velocity. When the simulation starts, fifty nodes are initially randomly placed within a 50m by 50m simulation area. Then each node moves to a random destination

with a randomly chosen speed which is uniformly distributed between 0 and the maximum moving speed. After arriving at that point, the mobile node will stay for a pause time and then move to the next random destination. The maximum node velocity in our six simulation scenarios is: 0.01m/s, 0.05m/s, 0.1m/s, 1.0m/s, 2.0m/s, 3.0m/s, respectively. At the slow velocity, (0.01m/s, 0.05m/s, 0.1m/s), we simulate people walking carrying handheld devices; at the high velocity situations, (1.0m/s, 2.0m/s, 3.0m/s), we simulate people jogging, or riding bicycle.

During our simulation, we found that the network performance varied a lot at the same maximum velocity, so we ran our simulations twenty times for each ad-hoc routing protocol at the same maximum moving speed to reduce the influence of one particular run. For any one of those twenty simulations, same node movement pattern and data traffic pattern are used for all ad-hoc routing protocols.

4.6.2 Radio Range

The mobile node network interface model in NS-2 approximates the Lucent WaveLAN DSSS radio interface [Fall2003]. We configured the radio transmission range to 10 meters. Longer transmission range covers more nodes and reduces packet hops, but more wireless signals transmitted within the same area can produce more collision, thus causing more delay and reducing the network throughput. With smaller transmission ranges, nodes may have connectivity problems and more partitioned networks may be formed; however, smaller transmission range allows more nodes to transmit at the same time with less interference.

4.6.3 Data Traffic

We used constant bit rate (CBR) over UDP as our traffic source. There are 15 different traffic flows with randomly chosen source and destination nodes. The start and stop time for each traffic flow are different and chosen randomly. The data packet size is 512 bytes, and its sending interval is 0.2 seconds using a wireless channel of 2Mbps. Simulation parameters are listed in Table 4.1.

Table 4.1: Simulation parameters for random scenarios

Parameter	Value
Number of nodes	50
Environment size	50m x 50m
Simulation time	250 seconds
Transmission range	10m
Bandwidth	2Mbps
Pause time	2 seconds
Traffic type	CBR over UDP
Packet interval	0.2 seconds
Packet size	512 bytes
Number of traffic flows	15
MAC	IEEE 802.11
Maximum moving speed (m/s)	0.01, 0.1, 0.5, 1.0, 2.0, 3.0
Hello messages interval (second)	MAR:1s; LMAR:1s; AODV:1s
Route advertisement time (DSDV)	1 second

Chapter 5 Simulation Results and Performance Analysis

Four ad-hoc routing protocols, LMAR, MAR, DSDV and AODV, were chosen in our simulations. The results of these simulations are illustrated and the network performances of routing protocols are compared.

5.1 Packet Delivery Ratio

The packet delivery ratio is the fraction of packets that successfully arrive at their destination. Figure 5.1 illustrates the simulation results of the average packet delivery ratio with standard deviation bar at six node velocities for four ad-hoc routing protocols.

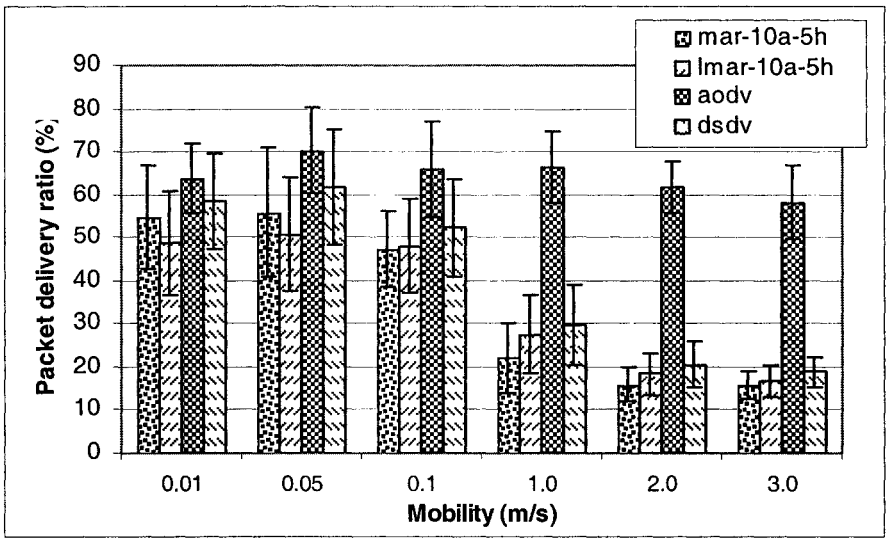

Figure 5.1: Packet delivery ratio with standard deviation bar

Figure 5.1 shows that all routing protocols have some packets loss at any mobility which can be caused by packet collisions, invalid routes, or packet drops. The average packet delivery ratio drops with the increase of maximum moving speed of mobile nodes;

however, AODV just drops from 64 percent to 58 percent, while DSDV, MAR and LMAR plunge from about 55 percent to 20 percent.

In our simulation, AODV has a high packet delivery ratio even at high node velocity. The reason is that AODV not only uses periodic "hello" messages to detect immediate neighbours, but also uses a MAC layer to detect link status. Larsson and Hedman in [Lars1998] indicated that AODV with only "hello" message dropped a very large portion of the data packets when node speed increased. The interval between the "hello" messages and the number of allowed "hello" message losses are important to detect link broken. If the "hello" message interval is shorter, the link breakages are detected earlier, but it would increase the control overhead in the networks. In NS-2, AODV uses both "hello" message and MAC layer support, so it has higher received packets ratio.

The DSDV, MAR and LMAR routing protocols have almost the same curve of packet delivery ratio. They drop almost 80 percent of data packets at the fastest speed, 3m/s. These three protocols all use only periodic broadcasts of "hello" messages to detect the wireless connection between nodes. When the moving speed increases, many packets will be dropped if the update of routing table cannot catch up with the change of network topology. In the DSDV protocol, the route advertisement time was set to one second which is same as the "hello" message interval in MAR and LMAR. The DSDV protocol has a higher packet delivery ratio than MAR and LMAR at all six moving speeds. One possible reason is that DSDV is based on the classical Bellman-Ford routing algorithm [Kuro2002] which has the advantage of finding the shortest route path.

Figure 5.1 also shows that the average packet delivery ratio of LMAR is about five percent lower than that of MAR at moving speed of 0.01m/s and 0.5m/s, almost the same value as that of MAR at 0.1m/s, and about four percent higher than that of MAR at moving speed of 1.0m/s. But, because the standard deviation of packet delivery ratio for MAR and LMAR are bigger than their difference in Figure 5.1, we consider they have similar performance of packet delivery ratio. For the same reason, we think that LMAR, DSDV and MAR protocols have similar packet delivery ratio at moving speed 0.1m/s and above.

5.2 End-to-End Delay

The end-to-end delay measures the time a data packet takes to move from its source node to its destination. Latency in a network usually includes four parts: the propagation delay, transmission delay, processing delay and queuing delay. In our simulation environment, since the wireless transmission range is 10m and wireless bandwidth is 2 Mbps, the propagation delay is about 0.033 microseconds, and transmission delay for agent packets, control messages and UDP data packets (512 bytes) are about 2 milliseconds at the most. They are smaller than the processing and queuing delays. When nodes move faster, the route discovery time could increase because of the more frequently invalid route paths. For the queuing delay, a routing failure or congestion will both force the packet to stay in the queue until it is timed out or transmitted.

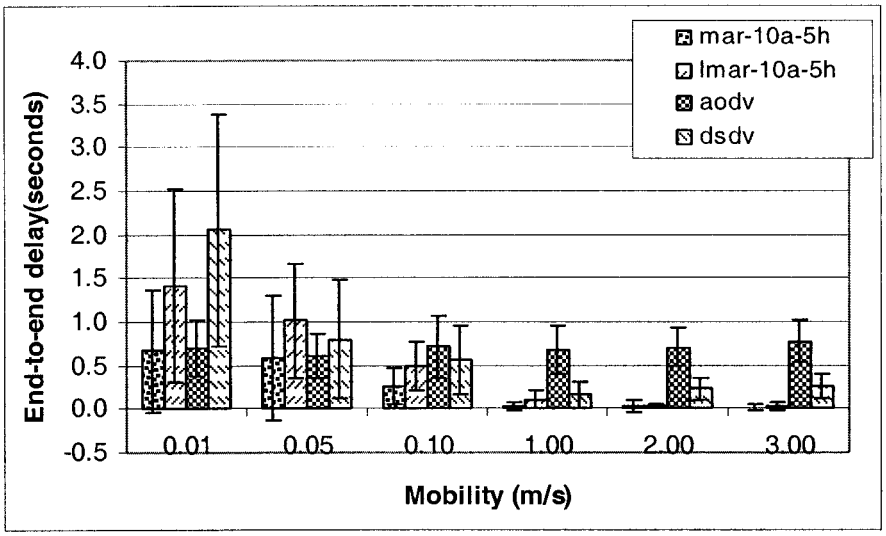

Figure 5.2: End-to-end delay with standard deviation bar

Figure 5.2 shows line graphs of the average end-to-end delays plotted against the maximum nodal velocity. The delay time in DSDV, MAR and LMAR decreases sharply when the mobile nodes move faster, while AODV has a relatively flat delay curve.

DSDV has the highest average delay time of 2.0 seconds at the lowest speed of 0.01 m/s, but then its delay time significantly reduces to below 0.25 seconds at high speed. MAR and LMAR exhibit the similar performance to DSDV. At the lowest moving speed of 0.01 m/s, LMAR has the second longest delay time of about 1.5 seconds, and MAR has the lowest delay of 0.67 seconds. These three protocols all use pre-built routing tables to find the next hop for packets, so packets will be dropped if nodes do not have valid route path. The data packets travelling through more hops will have a higher chance to be dropped than the packets with few hops because of invalid paths or packets timeout in ad-hoc networks. We can see from Figure 5.1 that DSDV, MAR, and LMAR protocols drop about 80 percent of packets in high speed scenarios, so only the packets with a short delay and few hops will tend to reach their destination successfully. As a result, the

average end-to-end decreases with the increase of nodal velocity. The DSDV protocol periodically dump incremental or complete routing table into the network, so the control packet overhead in DSDV is higher than other protocols, resulting in higher collision rate and more end-to-end delay

AODV has the lowest end-to-end delay time at 0.01m/s moving speed. Because AODV is a reactive routing protocol which only searches route to the new destination when it is needed, the control message overhead in the network is quite low. Therefore, data packets can often reach their destinations with little collision and low delay. We also see that AODV has the longest delay at the high speeds (1.0 m/s, 2.0m/s, 3.0m/s). Because AODV delivers many more packets than other three protocols in Figure 4.1, we expect higher end-to-end delay.

5.3 Packet Hop Count

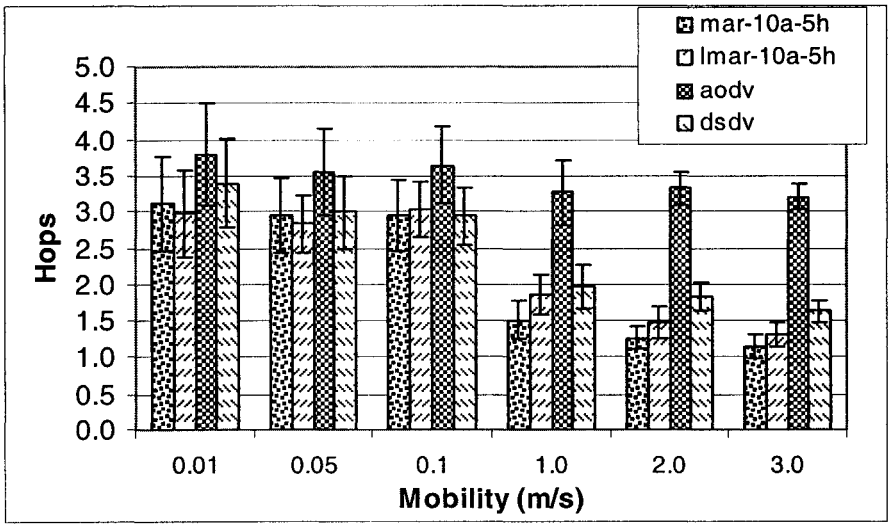

Figure 5.3: Hop count vs. mobility with standard deviation bar

Figure 5.3 shows the average packet hop counts at six moving speeds for four ad-hoc routing protocols with standard deviation bar. The figure shows that the average hop count drops when the maximum moving speed increases from 0.01m/s to 3.0m/s. DSDV, MAR and LMAR protocols drop more quickly than AODV.

AODV has the highest average packet hop counts at all six moving speed, which means that data packets routed by AODV can reach nodes farther away. Even at 3.0 m/s maximum moving speed, AODV has an average 3.3 hop counts, which is a very small drop from its peak value of 3.7 hop counts at 0.01 m/s moving speed. This indicates that the topology changes in our simulation have little influences on the average hop counts of AODV.

From Figure 5.3, we can see that LMAR, MAR and DSDV have very similar performance of hop counts at six moving speeds. They have average 3.0 hop counts at low nodal velocity, 1.5 hop counts at high nodal velocity. When nodes in the ad-hoc network move at high speed, for example 3.0m/s, the wireless links between nodes changes rapidly, so the packets sent to far destination will typically result in invalid path as the lack of routing update. Only packets to nearby nodes are received.

We can see that these three performance measures, packet delivery ratio, end-to-end delay, and packet hop counts, are not completely independent. For example, a lower packet delivery ratio means that the end-to-end delay is evaluated with fewer data packets, and packets with longer path lengths are probably dropped.

5.4 LMAR and MAR: Agent Number and History Size

LMAR and MAR are two agent-based ad-hoc routing protocols using agents to explore the network and find the routing path for data packets. The MIT paper [Nels1999] indicated that more agents are helpful for routing discovery. In this section, we test the influence of the number of agents on the network performance for LMAR and MAR protocols.

In our additional NS-2 simulations, we changed the number of agents from 10 to 50 and the history size from 5 to 7 for both LMAR and MAR protocols. We labelled our protocols with a suffix: "-10a-5h" means 10 agents and history size 5, while "-50a-7h" means 50 agents and history size 7. The additional simulation results are plot together with the prior results in Figure 5.4, Figure 5.5, and Figure 5.6. The network performance does not have any significant improvement with the increase of agents and history size; for the end-to-end delay, the results are worse when the nodal velocity is below 1.0m/s.

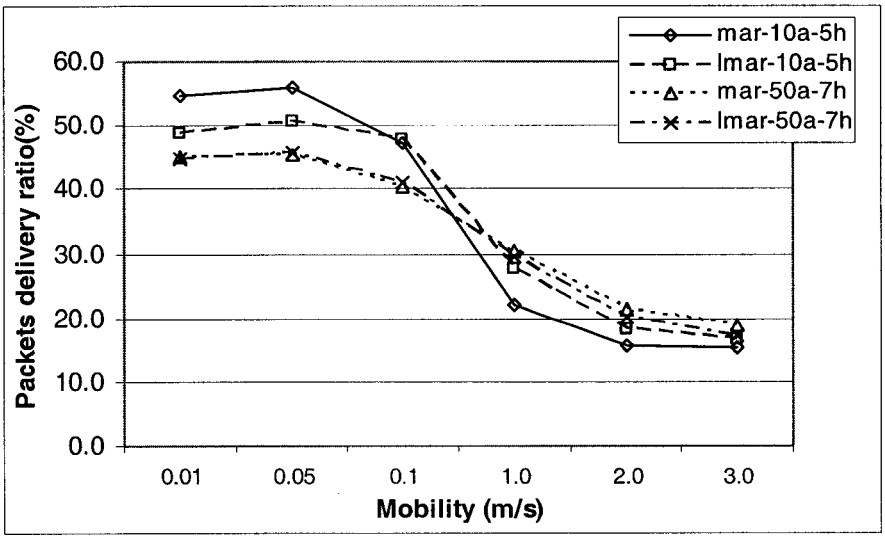

Figure 5.4: Packet delivery ratio vs. mobility

Figure 5.4 shows the average packet delivery ratio at six mobilities for MAR and LMAR. The MAR-10a-5h and LMAR-10a-5h have better packet delivery ratio than MAR-50a-7h and LMAR-50a-7h when nodes move at speed of 0.1m/s and below; if the moving speed increases to 1.0m/s and higher, MAR-50a-7h and LMAR-50a-7h have better delivery ratio. A possible reason is that 10 agents can handle the routing tasks without any problem at the slow speed scenarios; more agents in the network will create more congestion. When nodes move faster in the network, the links between nodes built or broken quickly and many data packets will be dropped because of invalid routes. More agents and longer history size can deliver extra information around the network, more data packets will arrive at their destinations.

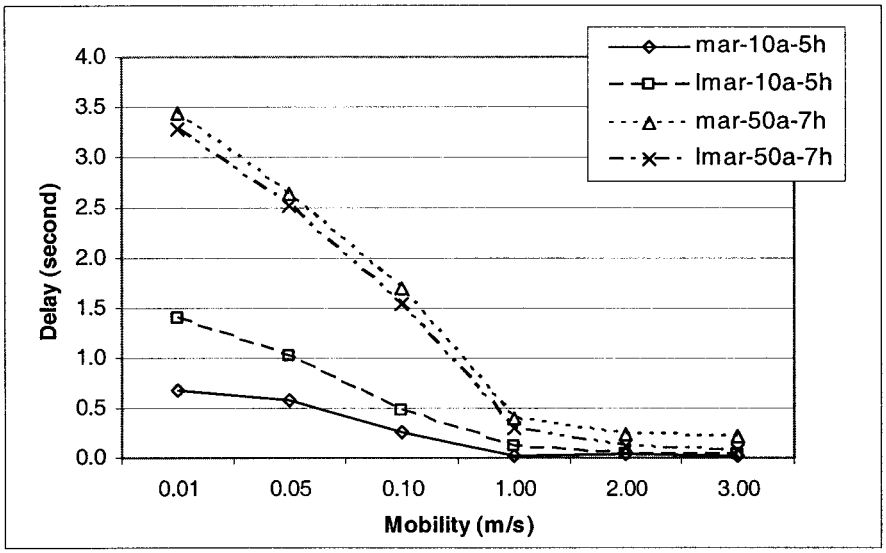

Figure 5.5: End-to-end delay vs. mobility

Figures 5.5 is the average end-to-end delay at different nodal velocities for LMAR and MAR with various agent numbers and history sizes. We can see that the LMAR-10a-5h and MAR-10a-5h always have lower end-to-end delay than LMAR-50a-7h and MAR-50a-7h. At the 0.01m/s moving speed situation, they have a very big difference. The

delay time of MAR-10a-5h is only about one fifth of that of MAR-50a-7h, and LMAR-10a-5h is about one fourth of LMAR-50a-7h. In the high data traffic, more mobile agents moving around cause more congestion which forces the data packets to stay in the interface queues for a longer time, thus increasing the average end-to-end delay time. In the high mobility situation, as the data packets can usually be transmitted for only 2 hops in our simulations, the difference of delay time between 10 agents/5 history size and 50 agents/7 history size is not substantial.

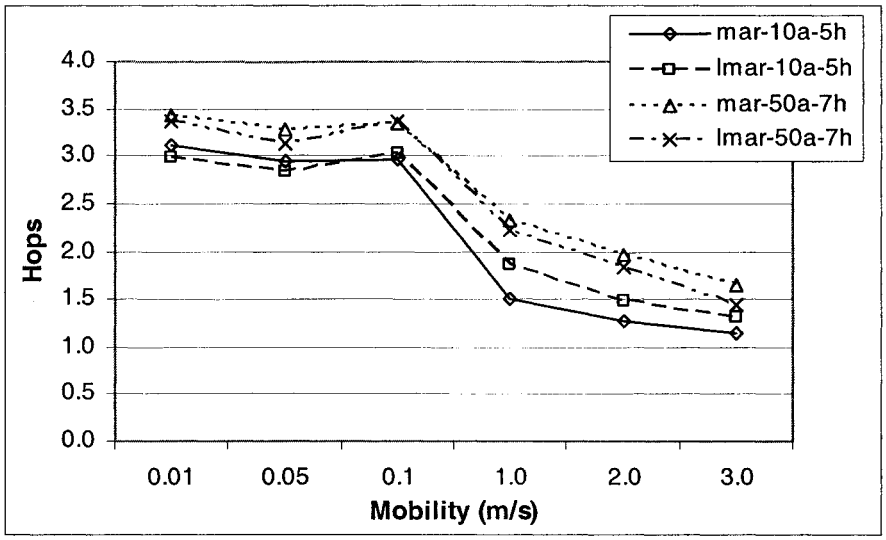

Figure 5.6: Hop count vs. mobility

Figure 5.6 is the average packets hop counts at six nodal velocities with different settings of MAR and LMAR protocols. We can see that MAR-10a-5h and LMAR-10a-5h have smaller hop counts than MAR-50a-7h and LMAR-50a-7h at all six moving speed scenarios. LMAR and MAR acquire their routing information through indirect communication between agents. When mobile nodes receive the information gathered by the agents, the algorithm they use to find the shortest path is not good as the Bellman-Ford routing algorithm [Kuro2002], so nodes may not discover the shortest path for their data packets to the destination with too many routing information. In our simulation

results, the MAR and LMAR protocols with 10 mobile agents deliver more data packets with lower end-to-end delay and shorter hop counts than those with 50 agents.

5.5 End-to-End Delay Distribution

In Figure 5.2 (Section 5.2), the standard deviation of delay time is quite long compared to the average delay time, which means that the delay time varied over a wide range. To understand the delay, the distribution of delay time is measured with range from 0 second to 20 seconds. Any data packets with delay times between 0 and 1 second will be considered as having 1 second delay, between 1 and 2 seconds as 2 seconds, and so on. Any data packet with more than a twenty second delay is counted as twenty seconds.

Figure 5.7 and 5.8 shows the delay distribution for the 0.01 m/s moving speed at 0 to 20 seconds range and 5 to 20 seconds range, respectively. Figure 5.7 illustrates that most packets in these four routing protocols are transmitted within 1 second, hundreds within 2 to 20 seconds, and few over 20 seconds. In Figure 5.8, we can see that in each delay range, DSDV has more data packets than the other three protocols which is probably caused by the control overhead of DSDV. We can also see the similar pictures in the Figure 5.9 and 5.10, which is for the 0.1m/s nodal velocity.

When the nodes move at 1.0m/s and faster, the average end-to-end delay of the AODV protocol is higher than other three routing protocols in Figure 5.2. We can also see from the Figure 5.11, 5.12, 5.13 and 5.14 that AODV has many more data packets with delay time of 5 seconds and above than DSDV, MAR and LMAR. AODV can deliver more

data packets in the high moving speed scenarios. Because some of them travel through more hops, we expect more delay.

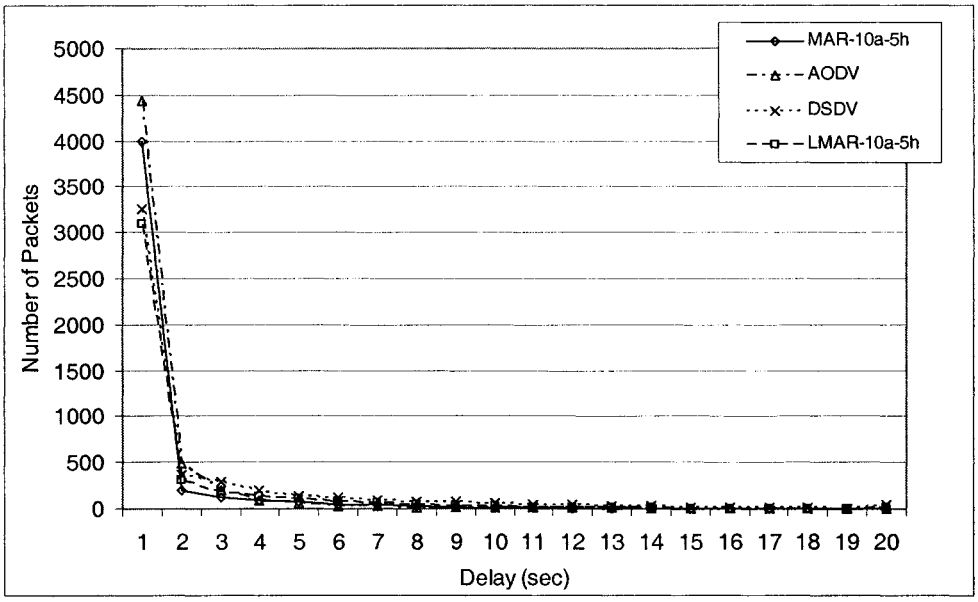

Figure 5.7: Delay distribution (0 – 20s) at 0.01m/s

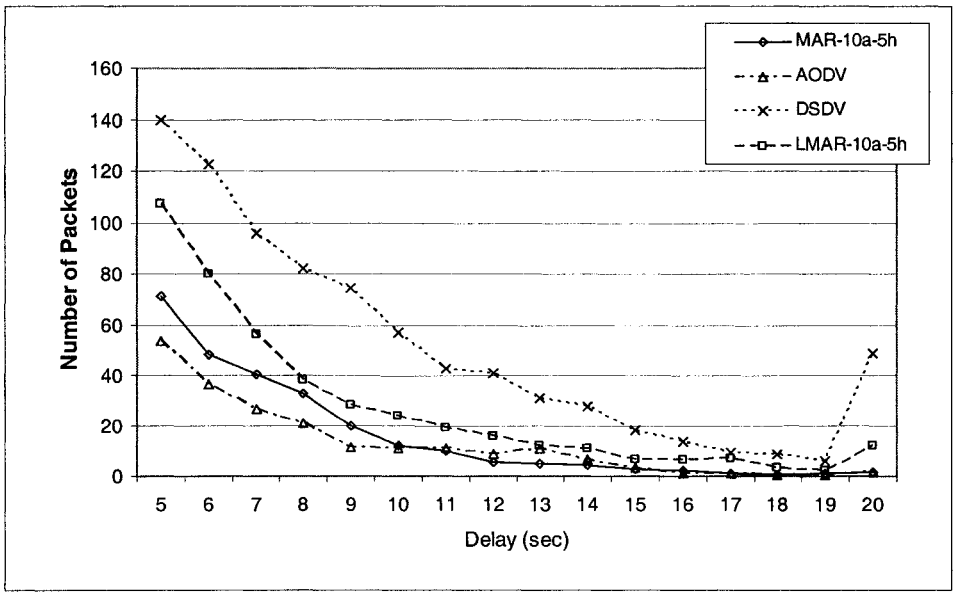

Figure 5.8: Delay distribution (5 - 20s) at 0.01m/s

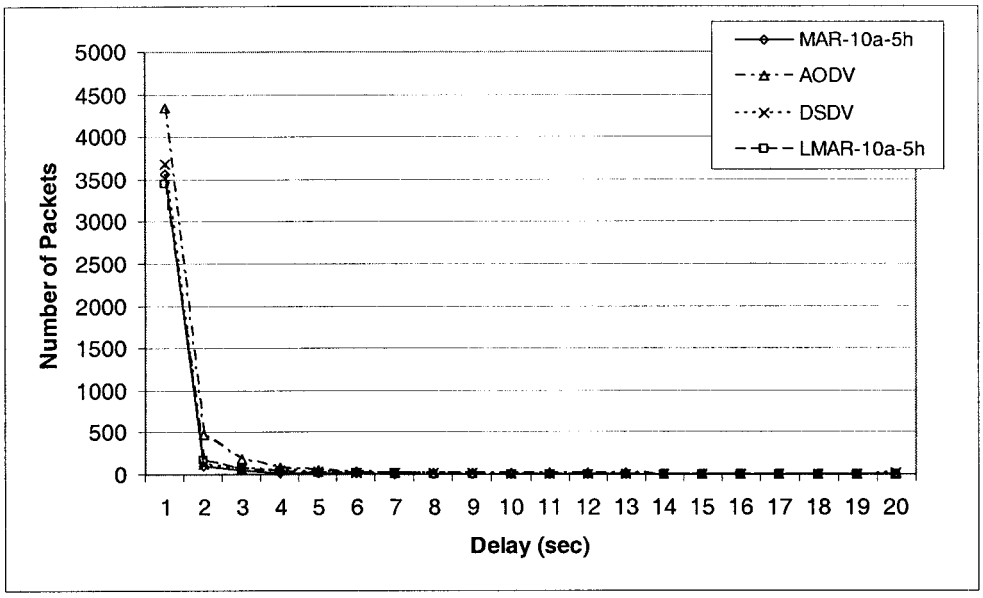

Figure 5.9 : Delay distribution (0 - 20s) at 0.1m/s

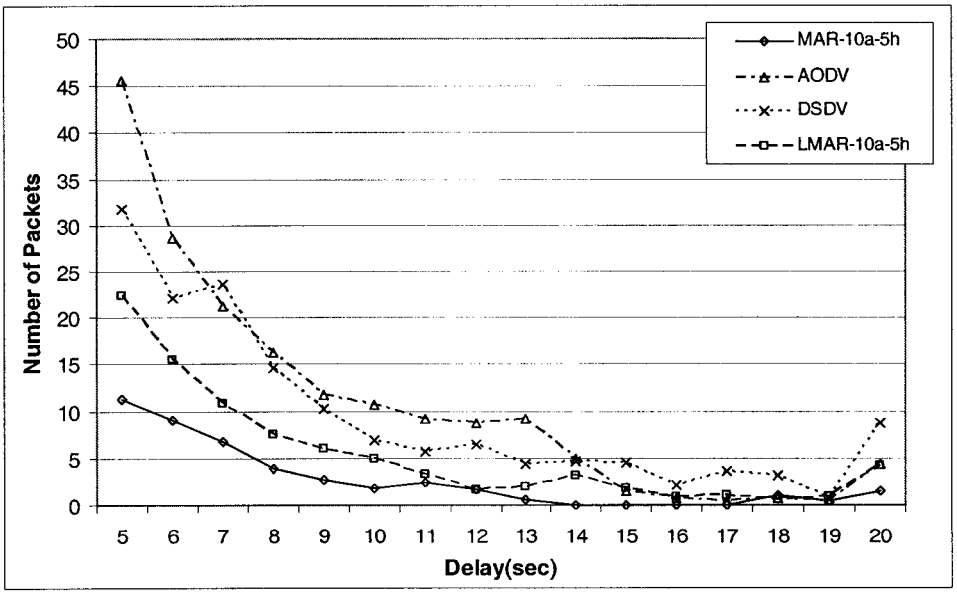

Figure 5.10: Delay distribution (5 - 20s) at 0.1m/s

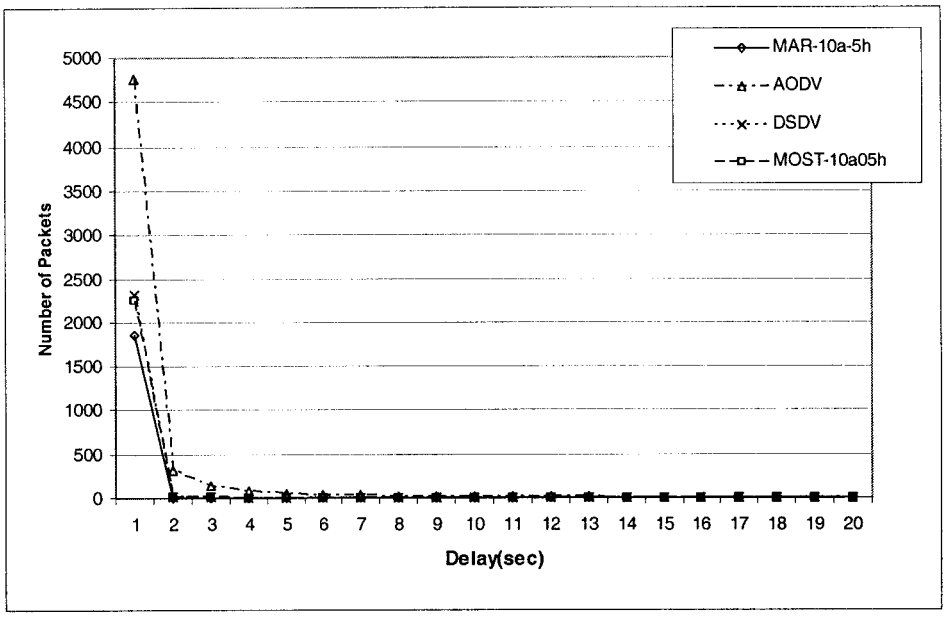

Figure 5.11: Delay distribution (0 - 20s) at 1.0m/s

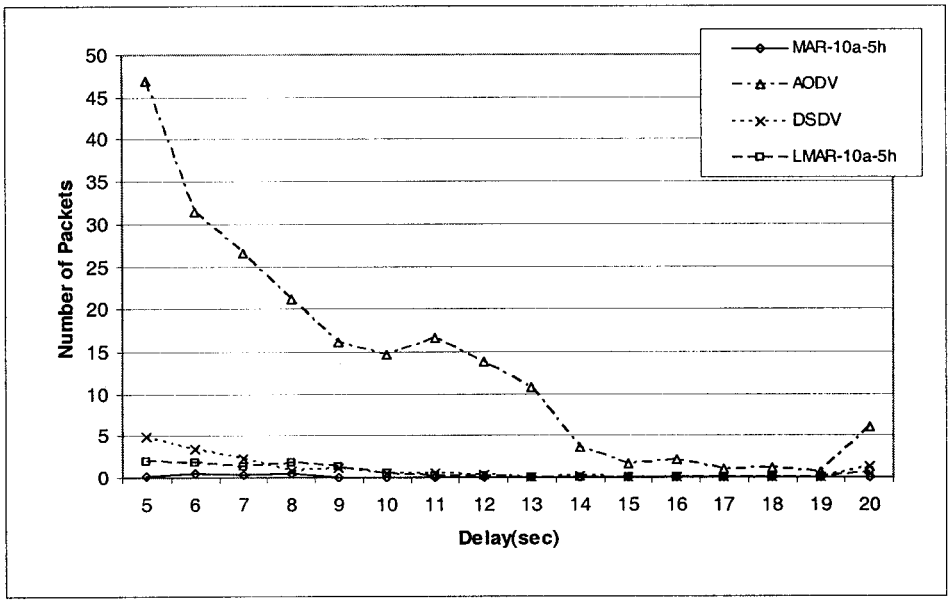

Figure 5.12: Delay distribution (5 - 20s) at 1.0m/s

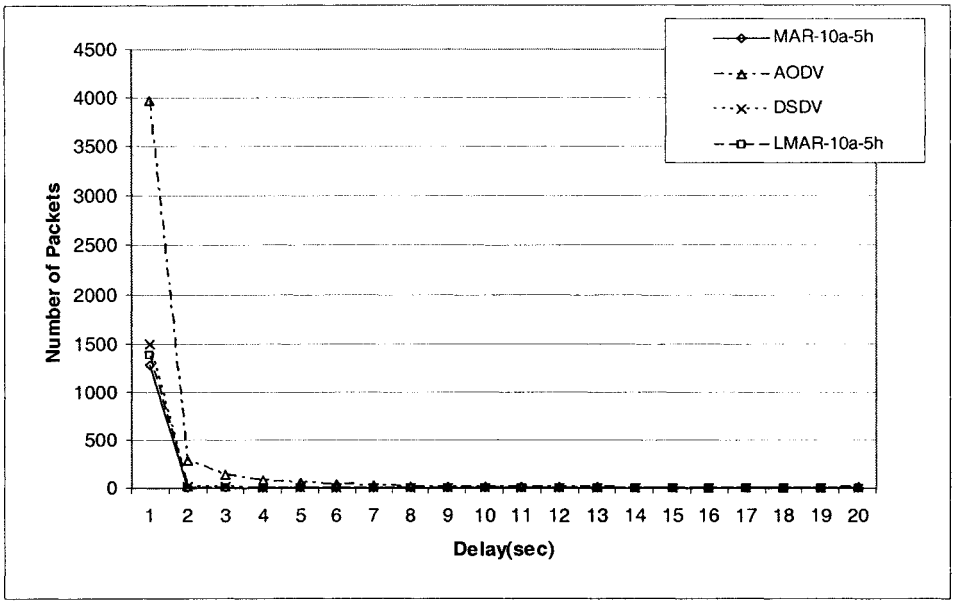

Figure 5.13: Delay distribution (0 – 20s) at 3.0m/s

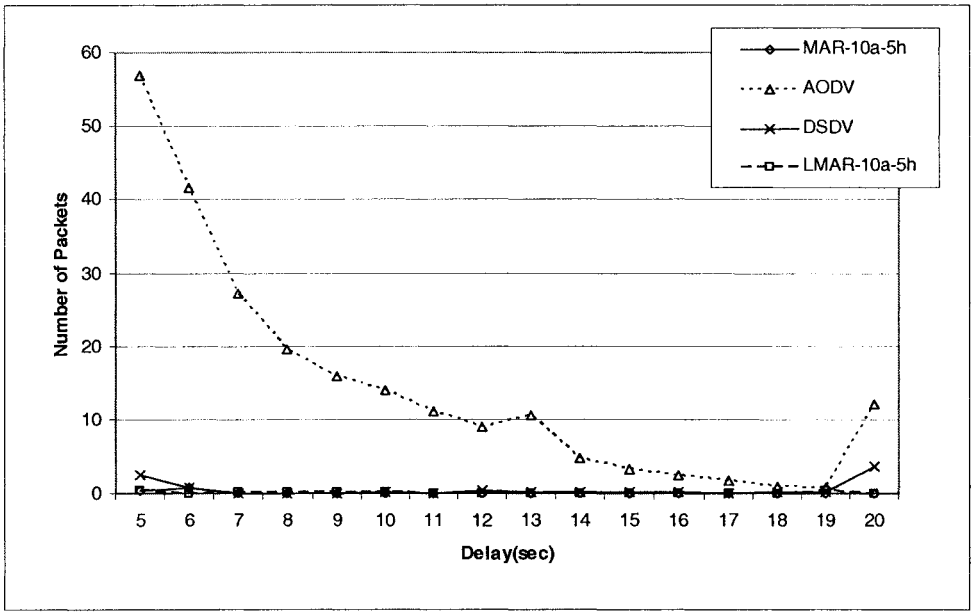

Figure 5.14: Delay distribution (5 – 20s) at 3.0m/s

5.6 Packets Hop Count and End-to-End Delay

To further understand the performance of the ad-hoc networks in our simulation, we investigated the correlation between end-to-end delay and hop counts. In our total 20 runs at 0.01 m/s for the four ad-hoc routing protocols, we found that the 13th run had many packets with a very long delay, so we use the result of that run for our analysis. For all received data packets, we recorded their delay time and hop counts, then we plotted them as a scatter-gram on the following figures.

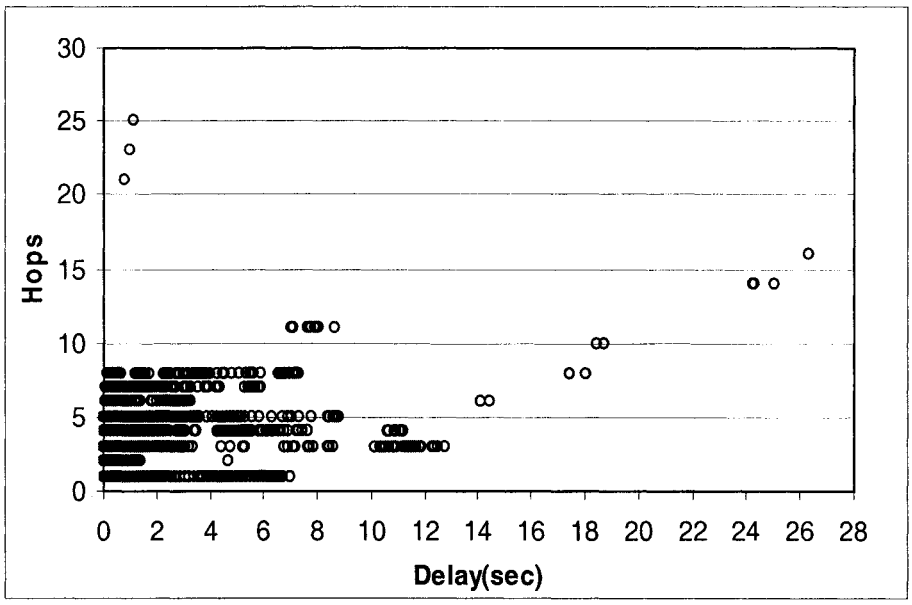

Figure 5.15: A scatter chart of hop count vs. delay for MAR at 0.01m/s

Figure 5.15 shows the packets hop counts and end-to-end delay for MAR at the 0.01m/s moving speed in the 13th run. We can see that there are three data packets that have less than 1 second delay, but travel more than 20 hops. Because the simulation field is 50m x 50m, and the wireless transmission range is 10m, 10 hops are normally enough for data packets travel through from the source nodes to the destination nodes. We looked at a particular data packet with 25 hops in the NS-2 output trace file, and found out that there

was a loop in the packet's path. We also observe that some data packets travel about 10 hops, but with about 20 seconds delay. As the processing delay and propagation delay are within 1 second as we calculated before, these long delays should be queuing delays which are caused by network congestion.

In the simulation trace file, we picked up a data packet with delay time 19.67 seconds. The packet's id number was 4292, and it was sent from source node 19 to the destination node 21. When the packet was on its way to the final end, it stayed in the node 33 for about 12 seconds. Further study indicated that the packet 4292 was received at 151.10 seconds, but only sent out from the node's MAC layer at 163.03 seconds. The queue in the node's MAC layer is first-in-first-out (FIFO), and it worked normally. As mobile nodes in ad-hoc networks act as routers forwarding packets between other nodes, in some circumstances, many data packets will pass through one particular mobile node, which will cause the network congestion and long delay of data packets.

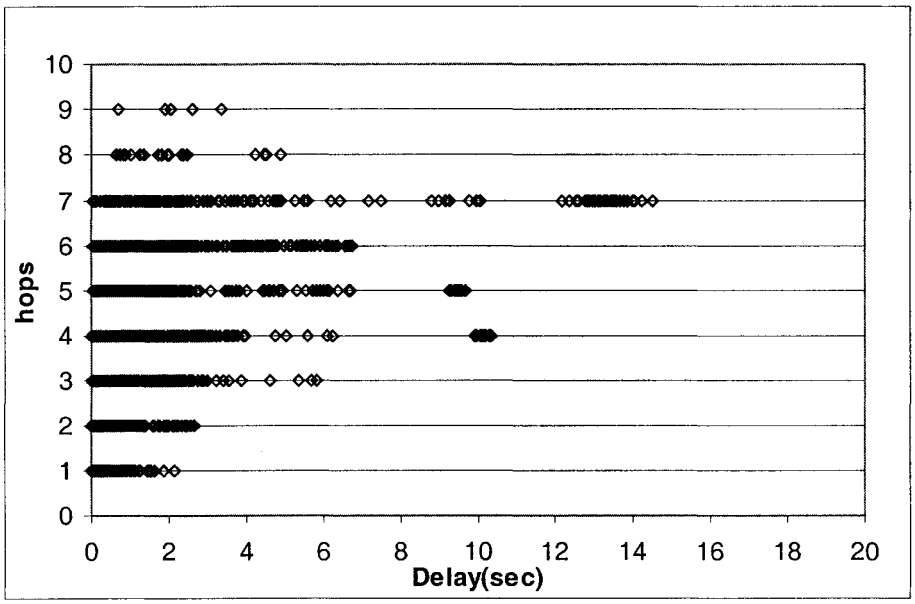

Figure 5.16: A scatter chart of hop count vs. delay for AODV at 0.01m/s

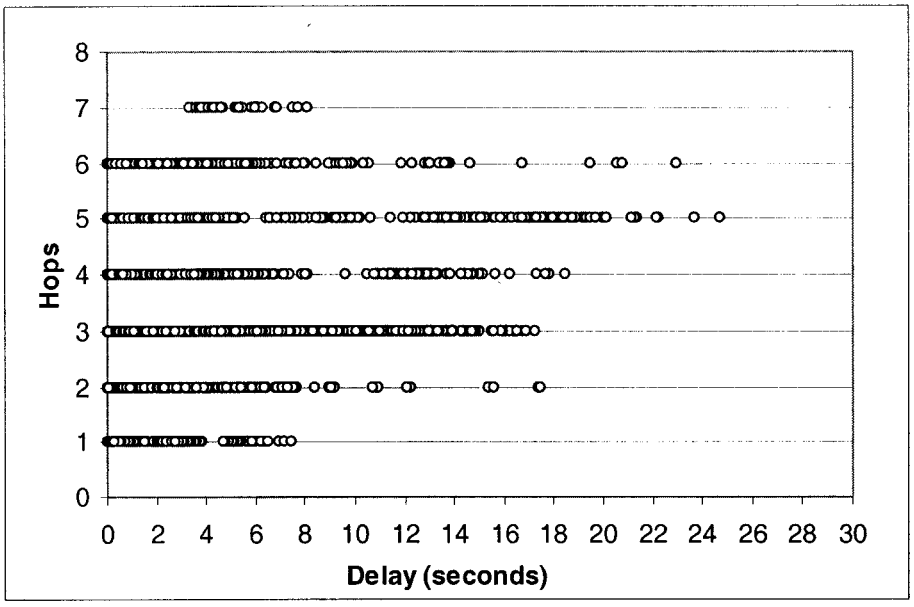

Figure 5.17: A scatter chart of hop count vs. delay for DSDV at 0.01m/s

Figure 5.16 and 5.17 shows the data packets hop counts with delay time at 0.01 m/s for AODV and DSDV routing protocols, respectively. We see that the delay time increases with the adding of hop counts in general. When the packets travel more hops, they will take more time. In Figure 5.16, we can see that all packets in AODV have hop counts less than 8, and the Figure 5.7 in Section 5.5 indicates that 90 percent packets in AODV have delay time of less than 2 seconds; on the other hand, many packets in DSDV take more than 10 seconds to their destinations in Figure 5.17. The Figure 5.8 in Section 5.5 also shows that there are about 50 data packets in DSDV experience delay of more than 20 seconds. As DSDV is a proactive routing protocol, the periodic update of whole network routing information will generate network congestion, and cause the longer delay.

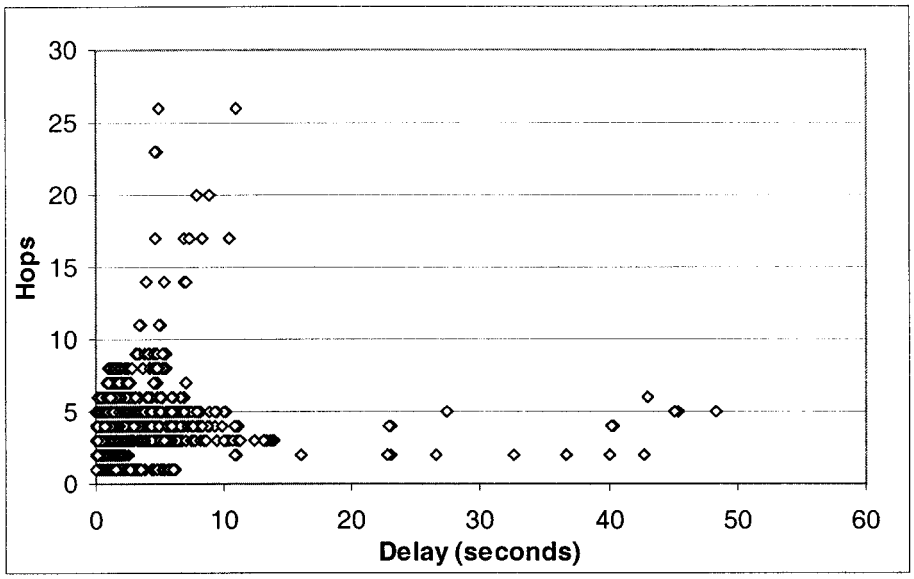

Figure 5.18: A scatter chart of hop count vs. delay for LMAR at 0.01m/s

Figure 5.18 shows that most data packets in the LMAR protocol take less than 8 seconds and 8 hops to their destinations. Few data packets have big hop counts which are caused by loops in their route paths. Other packets take more than 20 seconds to reach their destination. If we looked at the delay time along their route, we find out that sometimes a data packet stay on the interface queue of MAC layer of one node for a long time. Further investigation showed that at that time the data traffic around that node is very busy, so the data packet waits in the queue until it has chance to be sent out.

5.7 LMAR and MAR

In the LMAR protocol, we remove the global registry and control the average number of mobile agents in the ad-hoc network through the agent's lifetime and creation interval. By using the local information of node, our routing protocol can become feasible. The simulation results in the Figure 5.1, Figure 5.3 and Figure 5.5 show that our changes do not affect the network performance. The t-Test results within 95% confidence interval for

the different scenarios showed on Table 5.1 – 5.3 indicate that the LMAR and MAR routing protocols do not have significant different between their packet delivery ratio at all six simulation scenarios, end-to-end delay at high nodal velocities, and hop counts at low nodal velocities.

Table 5.1: t-Test results for packets delivery ratio

Nodal velocity	Mean for LMAR	Mean for MAR	t-Test results (Significant different)
0.01 m/s	48.75	54.86	No
0.05 m/s	50.75	55.90	No
0.1 m/s	47.93	47.17	No
1.0 m/s	27.60	22.11	No
2.0 m/s	18.39	15.97	No
3.0 m/s	16.86	15.66	No

Table 5.2: t-Test results for end-to-end delay

Nodal velocity	Mean for LMAR	Mean for MAR	t-Test results (Significant different)
0.01 m/s	1.41	0.68	Yes
0.05 m/s	1.02	0.59	No
0.1 m/s	0.49	0.27	Yes
1.0 m/s	0.11	0.03	Yes
2.0 m/s	0.03	0.03	No
3.0 m/s	0.03	0.02	No

Table 5.3: t-Test results for packet hop count

Nodal Velocity	Mean for LMAR	Mean for MAR	t-Test results (Significant different)
0.01 m/s	3.11	2.98	No
0.05 m/s	2.94	2.84	No
0.1 m/s	2.95	3.03	No
1.0 m/s	1.51	1.86	Yes
2.0 m/s	1.27	1.48	Yes
3.0 m/s	1.14	1.32	Yes

5.8 The LMAR Agent Population

In the Chapter 3, The LMAR Protocol, we prove that the average number of agents (n) is

$$n = NT / C,$$

where N is the number of mobile nodes, T the expiry time of agents, and C the time interval to created a new agent . In the experimental situation, the number of agents in the network will change frequently as the agent packets could be dropped early during their transmission. We counted the agent population during the whole simulation time in one run of the NS-2 simulations. The agent number is increased by 1 when a node generates a new agent; on the other hand, the agent number will be reduced by 1 if an agent is expired or dropped.

The Figure 5.19 shows the number of agents in one simulation with 50 mobiles nodes moving at 0.01m/s random speed in the area. The agent's creation interval is 10 seconds and the expiry time is 2 seconds. We ran this simulation for 100 seconds which is much longer than the agent's creation interval time.

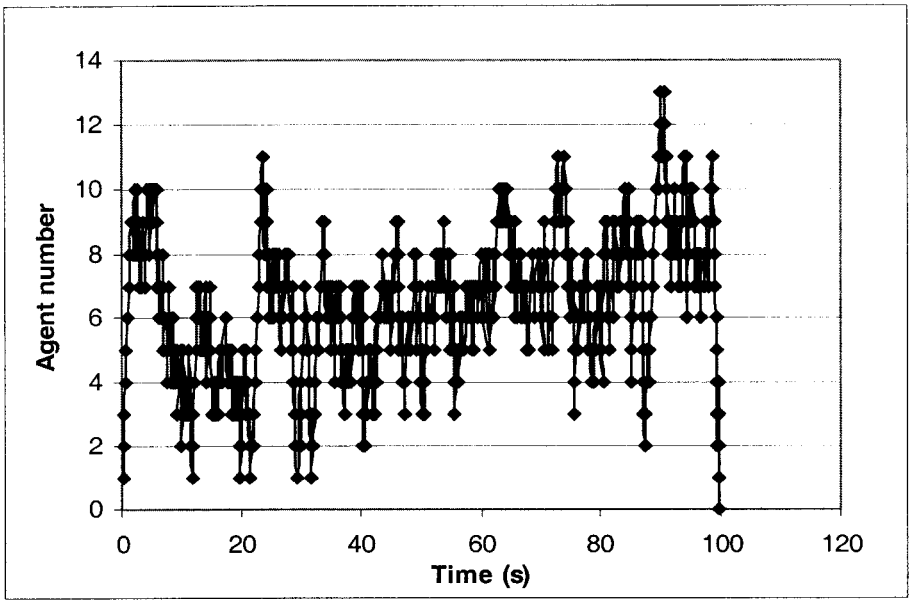

Figure 5.19: A scatter chart of LMAR agent population at 0.01m/s

We can see from the Figure 5.19 that the population of LMAR mobile agents grows up quickly to 10 at the first 2 seconds, and then fluctuates around 7 at the remainder of the simulation time. Finally, the number of agents decreases to 0 at the end. The average number of agents, except the first 2 seconds and the last one second, is 6.47, and the standard deviation of the agent population is 2.13. The experimental result of the average agent population is smaller than our theoretical prediction. We believe the reason is that mobile agent packets could be dropped earlier than their expiration time due to the transmission errors, packet collisions, or packet drops of busy network traffic.

There is a difference between the experimental result and the theoretical calculation; however, we can change the parameters, such as the agent's creation interval or expiry time, to create enough mobile agents. As a result, we can control the average number of mobile agents in ad-hoc networks using our LMAR algorithm.

Chapter 6 Conclusion and Future Work

6.1 Conclusion

In the thesis research, we designed and implemented an improved agent-based routing protocol for mobile ad-hoc networks. This protocol removes the global registry in the MAR routing protocol and uses the local information of a node to build the routing table. We conducted extensive network simulations with 50 mobile nodes moving about and communicate with each other within a 50m x 50m space in the NS-2 simulator. The network performance of the new routing protocol, LMAR, was compared with other three routing protocols, MAR, DSDV, and AODV. The LMAR and MAR are proactive agent-based routing, and DSDV uses the proactive table-driven routing algorithm while AODV uses the reactive on-demand routing strategy.

The simulation results show that the new routing protocol LMAR perform as well as the MAR in terms of the total number of received data packets and end-to-end delay in all simulation situations. In quite dynamic scenarios, LMAR has little higher received packet ratio and smaller end-to-end delay time than MAR. The data packets in LMAR and MAR sometimes experience looping as there is no prevention mechanism such as the destination sequence number adapted in DSDV and AODV.

DSDV updates the routing information by dumping the routing table of nodes throughout the network. When nodes mobility increases, the links between mobile nodes may be built and broken more frequently, which results in more overhead in updating all these

new routing information. When the data traffic is busy, the data packet in DSDV usually has long delay.

AODV uses reactive algorithm to create routes for data packets as and when required, so the overhead in AODV is the smallest in all four routing protocols. AODV also uses the feedback from the IEEE 802.11 MAC layer to indicate a failure of communication link, which results in a high packet delivery ratio even in the high node mobility situation.

6.2 Future Work

Mobile ad-hoc networks are an active research topic, and will become more active in the future. Agent-based routing is one kind of approach to the routing problem. There are still many questions to be investigated and future work to be done.

1. The detection of link failure by MAC layer. The use of MAC layer detection improves the response time of routing protocol for topology change, especially in the high nodal velocity. The routing protocols only depend on "hello" messages drop large amount of packets in high moving speed scenarios. This function should be added to the agent-based routing protocol.
2. The automatic setting of parameters. The agent's expiry time T, creation interval C and history size are now configured manually. A new protocol should be self-tuning to adapt to changes in topology and data traffic.
3. The analysis of bandwidth of mobile agents. The paper [Grun2003] investigated the user data traffic and routing traffic in a simple uniform continuum model for

ad-hoc networks. Simulations should be done to study the control overhead of mobile agents.

4. The prevention of looping in routes. AODV and DSDV routing algorithms use destination sequence number to prevent the route-looping. A similar mechanism or new algorithm should be designed to solve the problem in the agent-based routing protocol

Reference:

[Appl1994] Applyby, S. & Steward, S. 1994. Mobile Software Agents for Control in Telecommunication Networks. *BT Technology Journal,* vol.12, no.2, pp.104-113.

[Broc1998] Broch, J., Maltz, D.A., Johnson, D.B., Hu, Y-C. & Jetcheva, J. 1998. Multi-Hop Wireless Ad-hoc Network Routing Protocols, ACM Mobicom98, Dallas, Texas, USA

[Cele2002] Celebi, E. 2002. Performance Evaluation of Wireless Multi-Hop Ad-hoc Network Routing Protocols, Master Thesis, Bogazici University, Istanbul, Turkey.

[Chun2003] Chung, J., Claypool, M. 2003. NS by Example, http://nile.wpi.edu/ns/

[Cors1999] Corson, S. & Macker, J. 1999. Mobile Ad-hoc Networking (MANET): Routing Protocol Performance Issues and Evaluation Considerations, IETF Request for Comments: 2501, http://www.ietf.org/rfc/rfc2501.txt

[DiCa1997] Di Caro, G. & Dorigo, M. 1997. AntNet: A Mobile Agents Approach to Adaptive Routing, Technical Report 97-12, IRIDA, Universite Libre de Bruxelles

[DiCa1998] Di Caro, G. & Dorigo, M. 1998. AntNet: Distributed Stigmergetic Control for Communication Networks, *Journal of Artificial Intelligence Research,* 9, 317-365.

[Diep996] Diepstraten, W. & Belanger,P. 1996. IEEE 802.11 Tutorial – 802.11 MAC Entity, MAC Basic Access Mechanism, Privary and Access Control, IEEE P802.11-96/49C

[Fall2003] Fall, K. & Varadhan, K. 2003. The ns Manual. http://www.isi.edu/nsnam/ns/ns-documentation.html

[Grun2003] Grundke, E.W. & Zincir-Heywood, N. 2003. A Uniform Continuum Model for Scaling of Ad-Hoc Networks, Adhoc-Now'03, Montreal, Canada

[Hass2001] Hass, Z.J. & Pearlman, M.R. 1998. ZRP: A Hybrid Framework for Routing in Ad Hoc Networks. IN *Ad Hoc Networking*, C. Perkins, Ed. Addison-Wesley, Boston, USA

[Joha1999] Johansson, P., Larsson, T., Hedman, N., Mielczarek, B., Degermark, M. 1999. Scenario-based Performance Analysis of Routing Protocols for Mobile Ad-hoc Networks. ACM Mobicom'99, Seattle, Washington, USA

[John2002] Johnson, D.B. and Maltz, D.A. 2002. The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks (DSR). Internet Draft, IETF MANET Working Group, draft-ietf-manet-dsr-07.txt.

[Kram1999] Kramer, K.H., Minar, N. & Maes, P. 1999. Tutorial: Mobile Software Agents for Dynamic Routing, ACM SIGMOBILE Mobile Computing and Communications Review, vol.3, no.2.

[Kuro2002] Kurose, J.F. & Ross, K.W. 2002. Computer Networking: A Top-Down Approach Featuring the Internet (Second Edition). Addison Wesley, USA

[Lang1999] Lange, D.B. & Oshima, M. 1999. Seven Good Reasons for Mobile Agents. *Communications of the ACM*, vol.42, issue 3, pp.88-89

[Lars1998] Larsson, T. & Hedman, N. 1998. Routing Protocols in Wireless Ad-hoc Networks-A Simulation Study. Master Thesis, Lulea University of Technology, Lulea, Sweden.

[Mesh2004] MeshNetworks, Inc. 2004. Medford Mobilizes Wireless Workforce. http://www.meshnetworks.com/pdf/cs_medford_or.pdf

[Mina1999] Minar, N., Kramer, K.H. & Maes, P. 1999. Cooperative Mobile Agents for Dynamic Network Routing. IN *Software Agents for Future Communications Systems*, Springer-Verlag Telos

[Morr2003] Morrison, D. 2003. Agent-Based Ad-hoc Networking Routing: The Approach to Equilibrium. Bachelor Thesis, Dalhousie University, Halifax, Canada

[NS2003] The Network Simulator – ns-2. 2003. http://www.isi.edu/nsnam/ns/

[Perk2001] Perkins, C. 2001. *Ad Hoc Networking*. Addison-Wesley, Boston, USA

[Roye1999] Royer, E.M. & Toh, C.-K. 1999. A Review of Current Routing Protocols for Ad-Hoc Mobiles Wireless Networks. IEEE Personal Communications

[Stal2000] Stallings, W. 2000. *Data and Computer Communications (Sixth Edition).* Prentice Hall, Upper Saddle River, New Jersey, USA

[Toh2002] Toh, C.-K. 2002. *Ad hoc Mobile Wireless Networks: Protocols and Systems.* Prentice Hall PTR, Upper Saddle River, NJ, USA

[VINT2003] Virtual InterNetwork Testbed. 2003. http://www.isi.edu/nsnam/vint

[Zhou2003] Zhou, Y. 2003. Intelligent Agent Routing for Mobile Ad-hoc Networks. Master Thesis, Dalhousie University, Halifax, Canada