An improved MultiAnts-Aodv routing protocol for ad hoc wireless networks

Jun Wang
University of Nevada, Las Vegas

Follow this and additional works at: https://digitalscholarship.unlv.edu/rtds

Repository Citation
https://digitalscholarship.unlv.edu/rtds/1569

This Thesis is brought to you for free and open access by Digital Scholarship@UNLV. It has been accepted for inclusion in UNLV Retrospective Theses & Dissertations by an authorized administrator of Digital Scholarship@UNLV. For more information, please contact digitalscholarship@unlv.edu.
AN IMPROVED MULTIANTS-AODV ROUTING PROTOCOL FOR AD HOC WIRELESS NETWORKS

by

Jun Wang

Bachelor of Science
Beijing Polytechnic University, Beijing
1997

A thesis submitted in partial fulfillment of the requirements for the

Master of Science Degree in Computer Science
School of Computer Science
Howard R. Hughes College of Engineering

Graduate College
University of Nevada, Las Vegas
December 2003

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
The Dissertation prepared by

Ling Wang

Entitled

Scheduling and Partitioning VLSI Circuit Operation at Multiple Supply Voltages

is approved in partial fulfillment of the requirements for the degree of

Ph.D. in Electrical Engineering

Examination Committee Chair

Dean of the Graduate College

Examination Committee Member

Examination Committee Member

Graduate College Faculty Representative

PR/1017-52/1-00
ABSTRACT

An improved MultiAnts-AODV routing protocol for Ad Hoc Wireless Networks

by

Jun Wang

Dr. Ajoy K. Datta, Examination Committee Chair
Professor of Computer Science
University of Nevada, Las Vegas

Compared to the conventional table-driven and on-demand routing protocols, a hybrid routing protocol [71], which uses mobile agents and reactive route discovery, introduced a more realistic solution to this problem. However, the mobile agents were not fully exploited in this protocol. In this thesis research, we will propose an improved MultiAnts-AODV routing protocol based on ant-AODV. The goal of our design is to reduce the end-to-end delay and route discovery latency. To achieve a better performance, the communication scheme among the agents is strengthened. We also present an improved navigation algorithm for mobile agents to update the routing tables more efficiently. We extend the routing table to reduce the latency of routing discovery in case of link failures. The simulation based comparisons among several navigation algorithms are also presented.
TABLE OF CONTENTS

ABSTRACT.............................................................................................................................................. iii
LIST OF FIGURES...................................................................................................................................... vi
LIST OF TABLES....................................................................................................................................... vii
ACKNOWLEDGMENTS.......................................................................................................................... viii

CHAPTER 1 INTRODUCTION ................................................................................................................... 1
  1.1 The introduction of MANETs ................................................................................................. 1
  1.2 The Overview of routing problems in the MANETs ........................................................... 10
  1.3 Contributions of this thesis ................................................................................................... 14
  1.4 The organization of this thesis .............................................................................................. 15

CHAPTER 2 ROUTING USING MOBILE AGENTS................................................................................. 16
  2.1 The concept of mobile agent and its advantages ................................................................. 16
  2.2 Mobile Agents Applications ............................................................................................... 20
  2.3 The overview of the classic routing algorithm ................................................................. 21
  2.4 Ants based routing algorithms ............................................................................................ 24
  2.5 The benefits from multiple agents' collaborations ............................................................ 28

CHAPTER 3 CONVENTIONAL ROUTING ALGORITHM FOR MANETs................................................... 31
  3.1 Table driven algorithms .................................................................................................... 31
  3.2 On-demand algorithms ..................................................................................................... 36
  3.3 AODV algorithm .............................................................................................................. 41

CHAPTER 4 AN IMPROVED MULTIANTS-AODV ROUTING ALGORITHM.............................................. 50
  4.1 Ant-based routing algorithm for MANETs ........................................................................ 50
  4.2 An ANT-AODV Hybrid routing protocol for MANETs.................................................... 53
  4.3 An improved ANT-AODV Hybrid routing protocol for MANETs ................................. 55

CHAPTER 5 IMPLEMENTATION ISSUES AND SIMULATION RESULTS............................................... 71
  5.1 Introduction of the simulator ............................................................................................. 71
  5.2 Comparisons among 3 navigation algorithms ................................................................. 73

CHAPTER 6 CONCLUSION.................................................................................................................... 80
  6.1 The Thesis summary ......................................................................................................... 80
  6.2 The Future works ............................................................................................................. 81
LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Figure 1.1.1</td>
<td>Handoff between Access Points</td>
<td>2</td>
</tr>
<tr>
<td>Figure 1.1.1</td>
<td>An example of cellular Model</td>
<td>3</td>
</tr>
<tr>
<td>Figure 1.1.2</td>
<td>An example of an ad hoc network</td>
<td>4</td>
</tr>
<tr>
<td>Figure 1.2.1</td>
<td>Classification of Mobile Ad hoc Network Routing Protocols</td>
<td>11</td>
</tr>
<tr>
<td>Figure 2.1</td>
<td>Mobile Agents System</td>
<td>18</td>
</tr>
<tr>
<td>Figure 2.2</td>
<td>The Bellman-Ford algorithm updates the distance vectors</td>
<td>23</td>
</tr>
<tr>
<td>Figure 2.3</td>
<td>A two-bridge experiment demonstration</td>
<td>29</td>
</tr>
<tr>
<td>Figure 3.1</td>
<td>AODV Route Discovery Process</td>
<td>45</td>
</tr>
<tr>
<td>Figure 3.2</td>
<td>AODV Reverse Path Formation</td>
<td>45</td>
</tr>
<tr>
<td>Figure 3.3</td>
<td>AODV Forward Path Formation</td>
<td>47</td>
</tr>
<tr>
<td>Figure 3.4</td>
<td>AODV Route Maintenance</td>
<td>48</td>
</tr>
<tr>
<td>Figure 4.1</td>
<td>A demonstration of ants based routing</td>
<td>52</td>
</tr>
<tr>
<td>Figure 4.2</td>
<td>A demonstration of Ant-AODV routing protocol</td>
<td>55</td>
</tr>
<tr>
<td>Figure 4.3</td>
<td>No return rule</td>
<td>56</td>
</tr>
<tr>
<td>Figure 4.4</td>
<td>The migration strategy of &quot;conscientious agent&quot;</td>
<td>58</td>
</tr>
<tr>
<td>Figure 4.5</td>
<td>An example of &quot;Notification of Link Failures&quot; propagation</td>
<td>68</td>
</tr>
<tr>
<td>Figure 5.1</td>
<td>The snapshot of the simulation</td>
<td>73</td>
</tr>
<tr>
<td>Figure 5.2</td>
<td>The comparison of the convergence speed</td>
<td>74</td>
</tr>
<tr>
<td>Figure 5.3</td>
<td>The comparison of the total steps taken to deliver the data packet</td>
<td>75</td>
</tr>
<tr>
<td>Figure 5.4</td>
<td>The comparison of the actual hops taken to deliver the data packet</td>
<td>76</td>
</tr>
</tbody>
</table>
LIST OF TABLES

Table 3.1. Comparisons of On-Demand versus Table-driven Routing Protocols......41
Table 4.1. An example of AMIT table stored on nodes............................................65
Table 4.2. An example of routing table with two entries for each destination........67
Table 5.1. Sample data for the comparisons (10-30) mobile agents.........................77
Table 5.2. Sample data for the comparisons (40-60) mobile agents..........................78
ACKNOWLEDGMENTS

I would like to thank my thesis advisors Dr. Ajoy K. Datta and Dr. Maria Gradinariu for their guidance, support and encouragement throughout this thesis research. It has been a very rewarding experience to work with them.

I also would like to express my deepest gratitude to my wife and my parents.
CHAPTER I

INTRODUCTION

1.1 The introduction of MANETs

There has been tremendous interest in the past few years in mobile wireless networks, due to the progresses and advances in new technologies, although the concept of mobile wireless networks has existed for several decades. More and more small, inexpensive and powerful mobile computing devices are emerging, ranging from Personal Digital Assistants (PDAs), mobile phones, handholds, wearable computers to laptops. These mobile computing devices have the information processing and accessing capabilities with mobility. Subsequently, the demand of computing-on-the-move causes mobile wireless networking to gain popularity.

With the popularity of the mobile computing devices, “computing anywhere and anytime” is becoming a motto for the computer users naturally. Hence, the networking in the connections among various mobile computing devices becomes more critical than every before. With the development in wireless communication technologies in decades, wireless networking enables wireless mobile units to communicate with each other in diverse methods, but generally there are two distinct types of mobile wireless networks, infrastructured wireless networks and infrastructureless wireless networks.

1) Infrastructured: Wireless mobile networks are based on the traditional cellular networks,
2) where all nodes are one-hop away from a base station. Mobile nodes directly communicate with access points or base stations and generally do not establish point-to-point connections with other mobile nodes. Access points are usually connected to the rest of the network or the Internet. Typical examples of this kind of wireless networks are GSM, UMTS, WLL, WLAN, etc. As shown in Fig. 1.1.1, each access point has a coverage area, in which it is able to send signals to, and receive signals from other mobile nodes. Nodes within the area of an access point are able to communicate directly with that access point. But as the mobile node moves from the coverage area of one access point into that of another, a handoff occurs, where the node ceases communication with the old access point and begins communicating with the new access point. The handoff should be completely seamless so that the user is not aware that a transition in coverage areas has occurred.

![Handoff Area](image)

**Figure 1.1.1. Handoff between Access Points**

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
Figure 1.1.2. An example of cellular Model

Infrastructured wireless networks are commonly used in office buildings and college campuses or locations where the access points can be easily installed and connected to an existing network. WAP products are a typical example of the commercial application for this type of wireless networks.

3) Infrastructureless: In contrast to infrastructured networks, infrastructureless networks, also called mobile ad hoc networks (MANET) have no fixed backbone. All nodes are capable of movement. Communication in ad hoc networks is peer-to-peer as the mobile nodes communicate directly with one another. Since the transmission range of the nodes are also limited, for the communication of a given source and destination, multiple hops may be needed. Hence, the nodes must serve as routers for other nodes in the network so that data packets can be forwarded to their destinations. As shown in Fig. 1.1.3, an example of an ad hoc network. Because there is no fixed backbone along which routing could occur, an ad hoc network needs a routing protocol that can establish and maintain routes, even in networks with dynamic topologies.
1.1.1 The definition of the MANET and its applications

In short, a MANET is a collection of wireless nodes that can dynamically form a network to exchange information without any aid from the pre-existing fixed network infrastructure. This is a very important part of communication technology that supports truly pervasive computing, because in many contexts information exchange between mobile units cannot rely on any fixed network infrastructure, but on the rapid configuration of wireless connections on the fly [1].

Ad hoc networking can be applied anywhere where there is little or no communication infrastructure or the existing infrastructure is expensive or inconvenient to use. Ad hoc networking allows the devices to maintain connections to the network as well as easily adding and removing devices to and from the network. The set of applications for MANETs is diverse, ranging from large-scale, mobile, highly dynamic networks, to small, static networks that are
constrained by power sources. Besides the legacy applications that move from traditional infrastructured environment into the ad hoc context, a great deal of new services can and will be generated for the new environment. Typical applications include [1]:

1) Commercial sector. Ad hoc can be used in emergency/rescue operations for disaster relief efforts, e.g. in fire, flood, or earthquake. Emergency rescue operations must take place where non-existing or damaged communications infrastructure and rapid deployment of a communication network is needed. Other commercial scenarios include e.g. ship-to-ship ad hoc mobile communication, law enforcement, etc.

2) Military battlefield. Military equipment now routinely contains some sort of computer equipment. Ad hoc networking would allow the military to take advantage of commonplace network technology to maintain an information network between the soldiers, vehicles, and military information head quarters. The basic techniques of ad hoc network came from this field.

4) Local level. Ad hoc networks can autonomously link an instant and temporary multimedia network using notebook computers or palmtop computers to spread and share information among participants at a e.g. conference or classroom. Another appropriate local level application might be in home networks where devices can communicate directly to exchange information. Similarly in other civilian environments like taxicab, sports stadium, boat and small aircraft, mobile ad hoc communications will have many applications.

5) Personal Area Network (PAN). Short-range MANET can simplify the intercommunication between various mobile devices (such as a PDA, a laptop, and a cellular phone). Tedious wired cables are replaced with wireless connections. Such an ad hoc network can also
extend the access to the Internet or other networks by mechanisms e.g. Wireless LAN (WLAN), GPRS, and UMTS. The PAN is potentially a promising application field of MANET in the future pervasive computing context.

1.1.2 The History Of The MANETs

Although the mobile ad hoc network has been the focus of many recent research and development efforts, it is not a new concept. The roots of ad hoc networking can be traced back as far as 1968, when work on the ALOHA network was initiated (the objective of this network was to connect educational facilities in Hawaii). Although fixed stations were employed, the ALOHA protocol lent itself to distributed channel access management and hence provided a basis for the subsequent development of distributed channel-access schemes that were suitable for ad hoc networking. The ALOHA protocol itself was a single-hop protocol, that is, it did not inherently support routing. Instead every node had to be within reach of all other participating nodes. Inspired by the ALOHA network and the early development of fixed network packet switching, DARPA began work, in 1973, on the PRnet (packet radio network), a multihop network. In this context, multihopping means that nodes cooperated to relay traffic on behalf of one another to reach distant stations that would otherwise have been out of range. PRnet provided mechanisms for managing operation centrally as well as on a distributed basis. As an additional benefit, it was realized that multihopping techniques increased network capacity, since the spatial domain could be reused for concurrent but physically separate multihop sessions. Although many experimental packet radio networks were later developed, these wireless systems did not ever really take off in the consumer segment. When developing IEEE 802.11 [3], a standard for wireless local area networks (WLAN), the Institute of Electrical and
Electronic Engineering (IEEE) replaced the term packet-radio network with ad hoc network.

A new working group for MANET has been formed within the Internet Engineering Task Force (IETF) [2], aiming to investigate and develop candidate standard Internet routing support for mobile, wireless IP autonomous segments and develop a framework for running IP based protocols in ad hoc networks. The recent IEEE standard 802.11 [3] has increased the research interest in the field. Many international conferences and workshops have been held by e.g. IEEE and ACM. For instance, MobiHoc (The ACM Symposium on Mobile Ad Hoc Networking & Computing) has been one of the most important conferences of ACM SIGMOBILE (Special Interest Group on Mobility of Systems, Users, Data and Computing). Research in the area of ad hoc networking is receiving more attention from academia, industry, and government. Since these networks pose many complex issues, there are many open problems for research and significant contributions.

1.1.3 The Issues And Characteristics Of An Ad Hoc Network

MANETs allow devices to establish communication, anytime and anywhere without the aid of a central infrastructure. The share the following characteristics:

1) Autonomous terminal. In MANET, each mobile terminal is an autonomous node, which may function as both a host and a router.

2) Distributed operation. Since there is no background network for the central control of the network operations, the control and management of the network is distributed among the terminals.

3) Multihop routing. Basic types of ad hoc routing algorithms can be single-hop and multihop, based on different link layer attributes and routing protocols. Single-hop
MANET is simpler than multihop in terms of structure and implementation, with the cost of lesser functionality and applicability. When delivering data packets from a source to its destination out of the direct wireless transmission range, the packets should be forwarded via one or more intermediate nodes.

4) Dynamic network topology. Since the nodes are mobile, the network topology may change rapidly and unpredictably and the connectivity among the terminals may vary with time. MANET should adapt to the traffic and propagation conditions as well as the mobility patterns of the mobile network nodes. The mobile nodes in the network dynamically establish routing among themselves as they move about, forming their own network on the fly. Moreover, a user in the MANET may not only operate within the ad hoc network, but may require access to a public fixed network (e.g. Internet).

5) Fluctuating link capacity. The nature of high bit-error rates of wireless connection might be more profound in a MANET. One end-to-end path can be shared by several sessions. The channel over which the terminals communicate is subject to noise, fading, and interference, and has less bandwidth than a wired network. In some scenarios, the path between any pair of users can traverse multiple wireless links and the link themselves can be heterogeneous.

6) Light-weight terminals. In most cases, the MANET nodes are mobile devices with less CPU processing capability, small memory size, and low power storage. Such devices need optimized algorithms and mechanisms that implement the computing and communicating functions.

Regardless of the attractive applications, the features of MANET introduce several
challenges that must be studied carefully before a wide commercial deployment can be expected. These include:

1) Routing. Since the topology of the network is constantly changing, the issue of routing packets between any pair of nodes becomes a challenging task. Most protocols should be based on reactive routing instead of proactive. Multicast routing is another challenge because the multicast tree is no longer static due to the random movement of nodes within the network. Routes between nodes may potentially contain multiple hops, which is more complex than the single hop communication.

2) Security and Reliability. In addition to the common vulnerabilities of wireless connection, an ad hoc network has its particular security problems due to e.g. nasty neighbor relaying packets.

3) Quality of Service (QoS). Providing different quality of service levels in a constantly changing environment will be a challenge. The inherent stochastic feature of communications quality in a MANET makes it difficult to offer fixed guarantees on the services offered to a device.

4) Internetworking. In addition to the communication within an ad hoc network, internetworking between MANET and fixed networks (mainly IP based) is often expected in many cases.

5) Power Consumption. For most of the light-weight mobile terminals, the communication-related functions should be optimized for lean power consumption. Conservation of power and power-aware routing must be taken into consideration.

It has been widely recognized that routing strategy is the most important research problem
1.2 The Overview of Routing Problems In The MANETs

For mobile ad hoc networks, the issue of routing packets between any pair of nodes becomes a challenging task because the nodes can move randomly within the network. A path that was considered optimal at a given point in time might not work at all a few moments later. Ideally then, a good routing protocol is one that can deal with the typical limitations of mobile ad hoc wireless networks, which include high power consumption, low bandwidth, high error rates and highly dynamic topologies. According to [4] a well-designed routing algorithm for ad hoc wireless network must have the following qualities:

♦ Executes distributedly
♦ Provides loop-free routes
♦ Provides multiple routes to reduce congestion
♦ Establishes routes quickly
♦ Minimizes communication overhead to conserve bandwidth and power

Traditional routing protocols in packet switched networks are based on link-state or distance-vector routing algorithms. Both algorithms allow a host to find the next hop neighbor to reach the destination through minimum number of hops, that is, the shortest path. The message complexity of protocols based on these algorithms is high and hence they need to be modified to make it suitable for the limited bandwidth of wireless links. Also, the rapid changes in topologies make it important to find routes quickly, even if the route may not be the optimum. Ad hoc routing protocols are based on this philosophy. Since the 70’s researches on
the DARPA PRNET project[5], numerous protocols have been developed that take into account the characteristics of wireless networks. Routing for wireless ad-hoc networks can generally be categorized as table-driven (proactive) routing, on-demand (reactive) routing and some other miscellaneous routing protocols. As shown in Fig. 1.2.1.

![Classification of Mobile Ad hoc Network Routing Protocols](image)

Table driven routing protocols attempt to maintain consistent, up-to-date routing information from each node to every other node in the network. Each node needs to maintain one or more tables to store routing information, and they react to any change in the topology by propagating updates throughout the network in order to maintain a consistent topology view, even if no traffic is affected by the change. These protocols require periodic control messages to maintain routes to every node in the network. The rate at which these control messages are
sent must reflect the dynamics of the network in order to maintain valid routes. Thus, scarce resources such as power and link bandwidth will be used more frequently for control traffic as node mobility increases. Destination Sequenced Distance Vector Protocol (DSDV) [6] and Wireless Routing Protocol (WRP) [7] are two widely known table driven routing protocols for wireless networks. DSDV is based on the classic Bellman-Ford routing algorithm [8], in addition to which it uses sequence numbers to distinguish stale routes from new routes, thus avoiding the formation of loops. WRP, on the other hand avoids the "counting-to-infinity" problem by maintaining second-to-last hop information about every destination node. However, as shown in [9], table driven routing protocols that aim to maintain consistent information at all times usually suffer from high control overhead.

The IETF working group for Mobile ad hoc networking (MANET) has provided impetus for routing protocols in ad-hoc networks. Most of the routing protocols proposed in the MANET working group are on-demand routing protocols. On-demand routing protocols are the ones that attempt to create a route only when needed. They are designed with the aim of reducing control overhead, thus increasing bandwidth and conserving power at the mobile hosts. High routing overhead usually has a significant performance impact in low bandwidth networks, and can cause network congestion that in turn leads to data packet delay and eventually data loss. As more data packet are dropped due to network congestion the throughput of the network will drop down. In on demand protocols only when a host has data packet to send to a destination it will initiate a route discovery process within the network. Once a route has been established, it is maintained by a route maintenance procedure until the destination becomes inaccessible along every possible path from the source. Therefore, the
routing is source-initiated as opposed to table-driven routing protocols that are destination initiated.

There are several recent examples of the on-demand routing approach. These examples include Ad-Hoc On-Demand Distance Vector (AODV) [10], Associativity Based Routing (ABR) [11], Dynamic Source Routing (DSR) [12], Temporally Ordered Routing Algorithm (TORA) [13] and Zone Routing Protocol (ZRP) [14]. The routing protocols differ on the specific mechanisms used to disseminate flood-search packets and their responses, cache the information heard from other nodes' searches, determine the cost of a link, and determine the existence of neighbor. However, all on-demand routing protocols use flood search messages that either: (1) give sources the entire paths to destinations, which are then used in source-routed data packets (e.g., DSR); or (2) provide only the distances and next hops to destinations, validating them with sequence numbers (e.g., AODV) or time stamps (e.g., TORA).

The advantage of on-demand protocols is that they require less maintenance overhead, but the route acquisition time is very high and the global search procedure requires quite a bit of control overhead. Thus, these may not be ideal for real-time communication. Fortunately, a novel solution based on Ant-like mobile agents was proposed lately [15]. This family of protocols is inspired by the ant or swarm intelligence, which appears in social insect species. Mobile agents (MAs) are autonomous, intelligent programs that move through the network, performing actions on their creator's behalf [16]. Deploying mobile agents for network routing tasks has several advantages over the traditional methods, as presented in [17]. The usage of mobile agents in a wireless ad-hoc environment is even more appealing [15], because they
easily fit into the distributed nature of MANETs, providing adaptivity, flexibility, robustness and even efficiency, which are prime requisites in such environment. In this thesis, the suitability of mobile agents’ application in wireless mobile ad hoc networks’ routing problem will be argued.

1.3 Contributions Of This Thesis

As the introduction above, the dynamics of wireless ad hoc networks as a consequence of mobility and disconnection of mobile hosts pose a number of problems in designing proper routing schemes for effective communication between any source and destination [18]. Since the mobile agents paradigm was brought in this field, the nature advantages of the mobile agents ease the development of the routing protocols. This thesis is based on a hybrid routing protocol [19], which uses mobile agents and reactive route discovery, introduced a more realistic solution to this problem. However, the mobile agents were not fully exploited in that protocol. In this thesis research, we will propose an improved MultiAnts-AODV routing protocol based on ant-AODV. The goal of our design is to reduce the end-to-end delay and route discovery latency. To achieve a better performance, the communication scheme among the agents is strengthened. We also present an improved navigation algorithm for mobile agents to update the routing tables more efficiently. We extend the routing table to reduce the latency of routing discovery in case of link failures, and to increase the “goodput” by the introduction of the “notification of link failures” into the ants based routing protocols.
1.4 The Organization Of This Thesis

The first chapter has given a brief introduction to mobile ad hoc networks, and the related routing protocols for MANETs. The objectives and the organization of this thesis are also described in this chapter.

In the second chapter, the topics related with the mobile agents are presented. They include the overview of mobile agents: its definition, advantages, and applications. Next, the routing algorithms using mobile agents are introduced. Finally, the benefits from multiple-agents' collaborations are discussed.

The third chapter will talk about the conventional routing protocols for MANETs. Following two classes of the protocols, several major typical examples in each class are presented. At last in this chapter, AODV is emphasized in details. Their goodness and shortcomings are also stated briefly.

In the fourth chapter, an important and promising Ant-AODV routing protocol is argued fully, and to overcome its drawbacks, an improved navigation algorithm for the ants-based routing algorithm composed of the hybrid protocol is proposed and analyzed comprehensively. Besides, to strengthen the advantage of the hybrid protocol, some improving methods are applied to the algorithm and data structures.

The introduction of the simulator and the results of the simulation based comparisons are presented to demonstrate that the proposed navigation algorithm dose improve the performance of the protocol. Finally, the last chapter will come up with the conclusion of this thesis and the future works.
CHAPTER 2

ROUTING USING MOBILE AGENTS

2.1 The Concept of Mobile Agents and Its Advantages

The key features of mobile agents that distinguish them from traditional distributed programming are: mobility; network awareness; communication; intelligence; reactivity; autonomous; goal-oriented; temporally continuous; learning; flexible; and character [20,21]. Mobile agent technology has been proposed for a number of applications such as Internet-wide collaborative systems [22], network management [23], monitoring systems [24], information retrieval [25], intrusion detection systems [26], and e-commerce [27]. Mobile agents are ideal for such environments because of their ability to support asynchronous communication and flexible query processing. This is because user tasks can be delegated to mobile agents, when a mobile client is disconnected. Also, in certain cases, mobile agents can reduce network traffic compared to the traditional client-server approaches and maintain load balancing, thus increase performance of network nodes especially in wireless ad-hoc networks.

2.1.1 What Is A Mobile Agent?

The mobile agent concept is not new; and has been proposed to overcome certain limitations of traditionally designed distributed systems, especially client/server systems, and provide better flexibility by adding mobility of code, artificial intelligence, and improve data
and network management possibilities.

The extensive researches on the social insects show that the social insects have the following features: (1) they can respond to internal perturbations and external challenges; (2) failure of one or several individuals usually does not jeopardize a colony's functioning; (3) they have the ability to solve problems, sometimes difficult problems, in a distributed way, without any central control, on the basis on local information. Given the exhibited adaptability, flexibility and robustness properties and the impressive ecological success of social insects [28], it does not seem unreasonable to try to transfer current knowledge about how insect societies function into the context of engineering and distributed artificial intelligence.

Inherited with the nature advantages of social insects, in the past few years, mobile agents have proven to be an attractive tool in the design of various types of distributed services. A mobile agent is a program that can migrate from one node to another, perform various types of operations at these nodes, and can take autonomous routing decisions. In contrast with messages that are passive, an agent is an active entity that can be compared with a messenger. Thus, a mobile agent is an autonomous entity that has the ability to communicate with other agents and host systems. A mobile agent consists of its code and state, which carries with it during the self-initiated migration.

A typical agent model consists of the following six components [29]:

1. The identifier id, usually the same as the initiator's id. The id is unnecessary, if there is a single agent, but is essential to distinguish between multiple agents in the same system.

2. The agent program A
3. The briefcase B containing a set of variables

4. The previous process PRE visited by the agent

5. The next process to visit NEXT that is computed after every hop

6. A supervisory program S for bookkeeping purposes.

However, in reality a mobile agent system should provide an environment in which mobile agents can exist. This environment is called agent server, which hides the vendor specific aspects of its host platform and offers standardized services to an agent that is docking on to such a server. Services include access to local resources and applications, communication with other agents via message passing, migration, basic security services, creation and termination of agents. The infrastructure is set agent servers that run on top of platforms (nodes) within a possibly heterogeneous network (Figure 2.1). The platform that an agent originates is called home platform and is assumed as a trusted environment for that agent.

![Figure 2.1. Mobile Agents System](Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.)
Some well-known mobile agent systems are: MOLE [30], Telescript [31], Aglets Workbench [32], ffMAIN [33], and D'Agents [34]. Despite the fact that these systems were built to serve the same purpose, they have many differences in terms of terminology, concepts, and architecture. Some of these systems were developed in academic environments and others were developed by the industry. Nevertheless, some of these systems have already disappeared (such as Telescript, Kafta, or Odyssey), others will disappear, while others should emerge in a near future [35].

2.1.2 The Mobile Agents' Advantages

Compared with the traditional distributed systems which are built upon the stationary program and pass data back and forth across a network, mobile agents are programs that they themselves move from node to node: the computation moves, not just the attendant data. So it has the following three important properties:

1. Agents encapsulate a thread of execution along with a bundle of code and data. Each agent runs independently of all others, is self-contained from a programmatic perspective, and preserves all of its state when it moves from one network node to another.

2. An agent is able to cooperate with other agents in order to perform complex or dynamic tasks. Agents may read from and write to a shared block of memory on each node, and can use this facility both to coordinate with other agents executing on that node and to leave information behind for subsequent visitors.

3. Mobile agents can identify and use limited resources prepared for them where they lend now. After mobile agents left, all resources occupied by them are completely released. Such resources like the neighbor information of the node where the mobile agent is staying.
and it can be used by the agent.

4. The mobile agents can be implemented in a very small size, typically about several ten kilobytes or even smaller. This can be taken to make good use the bandwidth resource.

Using small, self-directed, mobile agents as building blocks allows us to design a network architecture that is flexible in several ways. First, because of the fundamentally distributed nature of collections of agents, our architecture can scale upwards in size quite gracefully. Second, because agent populations can change over time, new usage contexts and models can be accommodated. Finally, because all system interaction is mediated by agents, multiple network management strategies can coexist and co-evolve.

Therefore, mobile agents have the ability to support asynchronous communication and flexible query processing. The mobile user can assign a task to a mobile agent and when the agent feels that there is communication availability it will roam the network and fulfill the task delegated by its user. In this way, a mobile node requires less communication connectivity than it would need following traditional client/server approaches. Another equally important reason for mobile agents in wireless networks is that they can reduce network traffic.

2.2 Mobile Agents Applications

So far mobile agents are best viewed as a general tool for realizing arbitrary distributed applications. This view is reflected in the range of applications in which mobile agents are used. Perhaps the most common examples of mobile code are Java applets. Java applets are interactive applications that can be dynamically pulled across the network with a Java-enabled WWW browser. However, Java applets are not true mobile agents since they migrate only once.
True mobile agent systems include Telescript [36,37], Tacoma[38], Mobile service
Agents(MSA) [39], and Agent Tcl [40,41]. Telescript agents are currently used for network
management, active e-mail, electronic commerce, and business process management. In
network management, a Telescript agent might carry a software upgrade onto a machine along
with the code to perform the installation. The most visible use of Tacoma is StormCast, a
system for distributed weather simulation in which the volumes of data are so immense as to
make data movement impractical. Agent Tcl has been used primarily in information-retrieval
applications and is also being used in workflow applications, in which an agent carries a
multi-step task description from one site to another, interacting with the user at each site in
order to carry out that user’s part of the task [42].

Besides these applications, MA based algorithms were also used to solve classical routing
problems such as: Traveling Salesman Problem, Vehicle Routing Problem, Quadratic
Assignment Problem, connection-oriented/connectionless routing, sequential ordering, graph
coloring and shortest common super sequence. Furthermore, MA based routing protocols for
fixed, wired networks also presented the out performance over the traditional methods.

2.3 The Overview Of The Classic Routing Algorithm

Historically, the routing algorithms used in communication networks have evolved from
static routing in which “good routes” are computed off-line to more dynamic routing in which
the routes are computed online to take the node congestion level into account. Classical routing
protocols were successively based on Static Routing, Adaptive Distance Vector Routing in
which the routing tables are regularly updated and Adaptive Link State Routing which also
maintains a map of network topology and load pattern on each of its nodes. However, these routing algorithms react rather slowly to changes in the network load or topology and they are prone to oscillations.

2.3.1 Static Routing

Basic static routing in a communication network is equivalent to finding the shortest paths between the nodes of an associated graph. The metric used here can be the number of hops between two nodes, the physical distance, the transmission delay, etc. The classical Dijkstra algorithm [43] solves the shortest path problem in polynomial time. It can be used to build the routing tables required by the router to transmit entering packets toward their destination. The routing tables are built from the so called distance vector, which assigns the optimal distances to each destination for every outgoing entry.

The method only takes the topology into account but the network loads also need to be considered when the lines only have a limited transmission capacity or bandwidth. And while finding the shortest path can be solved in polynomial time, flow optimization, i.e. maximizing the number of packets transiting in the network per second (throughput), when lines have such transmission limitations is known to be a NP-complete problem [44] for which existing heuristics are still quite complex.
2.3.2 Adaptive Distance Vector Routing:

Dynamic distance vector routing periodically updates the distance vectors by exchanging information between neighbors. The Bellman-Ford distance vector routing algorithm [45] was the original ARPANET routing algorithm and was also used in Internet under the name RIP. It is based on the principle of dynamic programming [46]: an optimal path is made of sub-optimal paths. Each node $i$ periodically updates its distance vector from the distance vector regularly sent by its neighbors as follows:

$$\begin{align*}
D_{i,n}^i(t) &= d_{i,n}, \\
D_{i,d}^i(t) &= d_{i,n} + \min_{j \in N(n)} \{ D_{j,d}^j(t-1) \},
\end{align*}$$

(1)

Where $D_{i,d}^i(t)$ is the cost estimated by $i$ for delivering a packet from $i$ to $d$ by the way of the neighbor $n$ at time $t$ as shown on Figure 1, and where $d_{i,n}$ is the known distance between $I$ and its neighbor $n$.

This procedure may converge to the correct answer slowly, and particular this procedure is known to react promptly to good news (e.g. a new transmission lines) but slowly to bad news (e.g. a link failure), and it is also prone to oscillations [47]. For these reasons, it is nowadays often replaced by link state routing.
2.3.3 Adaptive Link State Routing

The link state algorithm essentially maintains a dynamic map of the complete network. This dynamic map is replicated on each routed and is used to estimate the optimal distances between nodes (usually with Dijkstra's algorithm). Each node periodically broadcasts its routing information to all destinations with a distributed flooding mechanism [47] trying to minimize the number of re-transmissions. The metric usually estimates the delays between a node and its neighbors based on the queue lengths on transmitting and receiving nodes. The OSPF protocol, which is increasingly being used in the Internet, uses such a link state algorithm.

2.4 Ants Based Routing Algorithm

Once a packet is required to be sent to a destination (point-to-point) or to multiple destinations (multicast), the router should recommend a good path (or even the shortest path) for sending this packet over the network. As searching for the optimal path in a stationary network is already a difficult problem [48], the searching for the optimal path in a faulty network or mobile network will be much more difficult. The ant routing algorithm [54], [49], [50], [51], [52], [53] is a recently proposed routing algorithm for use in these environments. The basic idea is similar to the path searching process of an ant. Once a request for sending a message is received from a server, the server will generate a number of mobile agents like ants. Those agents will then move out from the server to search for the corresponding destination host. Once an agent has reached the destination, it traverses all the way back to the source host, the server, following the path searched and leaves marks (just like the pheromone) on the hosts.
along the path. When all agents have been back, the source host will evaluate the costs of those paths collected and pick up the best path. If a connection is required, the server will send out an allocator agent to reserve resources from the hosts along the best path.

The philosophy of ants based routing algorithm is that since the goal of every routing algorithm is to direct traffic from sources to destinations maximizing network performance while minimizing costs, the general problem of determining an optimal routing algorithm can be stated as a multi-objective optimization problem in a non-stationary stochastic environment. Furthermore, information propagation delays and the difficulty to completely characterize the network dynamics under arbitrary traffic patterns, make the general routing problem intrinsically distributed. Routing decisions can only be made on the basis of local and approximate information about the current and the future network states. These features make the problem well adapted to be solved following the rationale of the ants' intelligence. The intelligence gives rise to complex and often intelligent behavior through complex interaction of thousands of autonomous ant members. Interaction is based on primitive instincts with no supervision. The end result is accomplishment of very complex forms of social behavior and fulfillment of a number of optimization and other tasks.

2.4.1 Two Types Of Ants Based Routing Algorithms

The ants based routing algorithms can be classified two types depending on when the distance vector is updated. This update can either be performed by routing agents going from the source to the destination---forward routing, or by agents retracing their way back to their source---round trip routing. Round trip routing is based on two sets of homogeneous mobile agents, respectively called the forward agents and the backpropagating agents. The forward
agents share the same queues as data packets and use the same routing tables. They keep track of their journey and the associated costs between hops in an internal stack. Forward agents also implement a mechanism to avoid loops in their routes. The backpropagating agents retrace their way back to the source and update the distance vector accordingly. These backtracking agents however have a higher priority over data for a faster propagation of the accumulated information. The typical algorithm in this class is the AntNet system proposed by Di Caro and Dorigo [53], which has been shown to outperform many aspects of the OSPF and Bellman-Ford routing algorithms.

However, backward routing is intrinsically slow since it requires the agent to reach its destination before any update to begin. This slow round trip reaction to changes in the network might induce oscillations. Forward routing offers an alternative by removing the need of round trips. It was first introduced by Schoonderwoerd et al. [55] in the case of virtual circuit based symmetric networks. It uses a population of simple mobile agents with behaviors modeled on the trail laying abilities of ants. The ants move across the network between randomly chosen pairs of nodes; as they move they deposit simulated pheromones as a function of their distance from their source node, and the congestion encountered on their journey. They select their path at each intermediate node according the distribution of simulated pheromones at each node. The MAs based routing algorithms for MANETs which will be talked about in chapter four is very similar as this one. This protocol showed very promising results and turned out to be highly adaptive in dynamic network environments.

2.4.2 The Overview Of The AntNet Algorithms

Suppose a datagram network, with N nodes, being s a generic source code that generates an
agent (or ant) toward a destination d. A routing table is organized as in vector-distance algorithms, but in the table, a probability value \( P_m \) which express the goodness of choosing \( n \) as next node when the destination node is i. Each entry is stored for each pair \( (i, n) \) with the constraint:

\[
\sum_{n \in N_i} P_n = 1, \ i \in [1, N], N_i = \{\text{neighbors}(k)\};
\]

Two types of ants are defined:

a) Forward Ant, or \( F_{s \rightarrow d} \), which will travel from a source s to a destination d. The identifier of every visited node k and the time elapsed since its launching time to arrive at the k-th node, are pushed into a memory stack \( S_{s \rightarrow d}(k) \) and inserted in a dictionary structure \( D_{s \rightarrow d} \), carried by the agent.

b) Backward Ant, or \( B_{s \rightarrow d} \), that will be generated by a forward ant \( F_{s \rightarrow d} \) in the destination d. It will return to s through the path used by \( F_{s \rightarrow d} \). In its way to s, \( B_{s \rightarrow d} \) updates routing tables of the visited nodes using the information already collected by \( F_{s \rightarrow d} \).

Each traveling agent selects the next hop node using the information stored in the routing table. If the node chosen (proportionally to the goodness of each neighbor node) was already visited, a uniformly random selection among the neighbors is applied.

If a cycle is detected, that is, if an ant is forced to return in an already visited node, the cycle’s nodes are popped from the ant’s stack and all the memory about them destroyed. \( B_{s \rightarrow d} \) updates the k routing table and list of trips, for the entries regarding to nodes k’ between k and d inclusive, according to the data carried in \( S_{s \rightarrow d}(k') \), increasing probabilities associated to path used and decreases other paths probabilities, by mean of a criteria explained in [56].
2.5 The Benefits From Multiple Agents' Collaborations

Another very useful and important feature of multiple agents module is the collaborations among the MAs. This approach is inspired by the work of biologists studying social insects, who have uncovered the mechanisms controlling the foraging behaviors of ants. The most important method is the laying and sensing of trails of pheromones—specialized chemical substances which are laid in amounts determined by local circumstances, and which by their local concentration subsequently directly influence an ant's choice of route. The principle behind these interactions is called stigmergy, or communication through the environment [56].

An example [57] is pheromone laying on trails followed by ants. Pheromone is a potent form of hormone that can be sensed by ants as they travel along trails. It attracts ants and therefore ants tend to follow trails that have high pheromone concentrations. This causes an autocatalytic reaction, i.e., one that is accelerated by itself. Ants attracted by the pheromone will lay more of the same on the same trail, causing even more ants to be attracted. The two-bridge experiment demonstrates the function of the stigmergy. When food source is separated from an ant nest by two bridges R1 and R2, where R1 being longer than R2, then the shorter bridge R2 is selected by the colony (if R1 is sufficiently longer than R2). This is attributed to the trail-laying and trail following characteristic of ants in which ants lay pheromone as described above. This attracts the other ants. The ants returning first to the nest would have laid the pheromone twice on the shorter path hence influencing outgoing ants to take the shorter route instead of the longer one. Even if the long bridge is presented first, the shorter one will still be selected subsequently as the pheromone trace on the longer branch would evaporate and it would be difficult to maintain a stable pheromone trail on a longer path.
than on a shorter one. Therefore, as a result of this "auto-catalytic" effect, the shortest path will emerge rapidly. It is presented in fig.2.3 in below.

![Diagram](image-url)

**Figure 2.3. A Two-Bridge Experiment Demonstration**

Another form of stigmergy alters the environment in such a manner as to promote further similar action by the agents. This process is dubbed task-related stigmergy. An example is sand grain laying by termites when constructing nests [58]. In the initial stages of construction, termites lay sand grains at random locations. This stimulates further laying by other members of the swarm, until a single heap of sand grains randomly reaches a critical mass that is larger than its neighboring heaps. At that point, most termites are attracted to that specific heap, thereby selecting that specific site for construction of their nest.

As the description above, the ant intelligence boasts a number of advantages due to the use of mobile agents and stigmergy. These are:
1. Scalability: Population of the agents can be adapted according to the network size. Scalability is also promoted by local and distributed agent interactions.

2. Fault tolerance: Swarm intelligent processes do not rely on a centralized control mechanism. Therefore the loss of a few nodes or links does not result in catastrophic failure, but rather leads to graceful, scalable degradation.

3. Adaptation: Agents can change, die or reproduce, according to network changes.

4. Speed: Changes in the network can be propagated very fast, in contrast with the Bellman-Ford algorithm [45].


6. Autonomy: Little or no human supervision is required.

7. Parallelism: Agent's operations are inherently parallel.

These properties make ant intelligence very attractive for ad-hoc wireless networks. They also render ant intelligence suitable for a variety of other applications, apart from routing, including robotics [59] and optimization [60].
CHAPTER 3

CONVENTIONAL ROUTING ALGORITHM FOR MANETS

As presented in chapter 1, the routing protocols for ad hoc networks can be categorized in two ways: (1) table-driven and on-demand (source initiated) based on the way routers obtain information and (2) link state and distance vector based on the type of information that they use. Routers running a distance-vector protocol use distance or path information to destinations, whereas routers running a link-state use topology information to make routing decisions. In this thesis, we will review the protocols in the first way of the categorization.

Table-driven protocols are traditionally proactive in nature, which find routes between all source-destination pairs regardless of the use or need for such routes. On-demand protocols are reactive in nature, in the sense that they initiate routing activities only in the presence of data packets in need of a route. Also there are protocols, which are hybrid between proactive and reactive protocols. They combine the advantages of both.

3.1 Table Driven Algorithms

Characteristics of table-driven protocols are that they attempt to maintain consistent, up-to-date routing information from each node to every other node in the network. These protocols require each node to maintain one or more tables to store routing information. They
respond to changes in network topology by propagating updates throughout the network in order to maintain a consistent network view. The aspects in which each protocol differs are in the number of necessary routing-related tables and the methods by which changes in the network structure are broadcast. Protocols that keep track of routes for all destinations in the ad hoc network have the advantage that communications with arbitrary destinations experience minimal initial delay from the point of view of application. When the application starts, a route can be immediately selected from the route table. They are called proactive because they store route information even before it is needed. The disadvantage with table-driven protocols is that they create additional control traffic that is needed to continually update stale route entries. An increased control overhead gives rise to bandwidth resources wastage, congestion in the network and possible loss of data packets, retransmission of data packets and further delays.

Here are some popular table-driven protocols below:

3.1.1 DSDV

DSDV [61] is based on the classical distributed Bellman-Ford routing algorithm. DSDV is a hop-by-hop distance vector routing protocol requiring each node to periodically broadcast routing updates. The key advantage of DSDV over traditional distance vector protocols is that it guarantees loop-freedom. The routing table maintained in each DSDV node lists the next hop for each reachable destination. DSDV tags each route with a sequence number and considers a route with a greater sequence number more favorable than others with a lower number. Or in the case that all routes have equal sequence numbers, the one with a lower metrics will be chosen. Each node in the network advertises a monotonically increasing even sequence number for itself. When a node S decides that its route to a destination D has broken, it advertises the
route for D with an infinite metric and a sequence number one greater than its sequence number for the route that has broken (making an odd sequence number). This causes any node A routing packets through S to incorporate the infinite-metric route into its routing table until node A hears route to D with a higher sequence number.

To reduce the potentially large volume of network traffic produced by routing updates, DSDV uses two different types of update. Full updates are broadcast periodically and include every entry in the routing table. Incremental updates include only those routing entries that have changed since the last full update. Incremental updates are triggered when significant changes are made to the routing table. For instance, a route invalidation is considered sufficiently important to trigger an update. Still, DSDV has been shown to have very high routing overhead compared to on-demand routing protocols. While the number of routing packets transmitted per second will be smaller for DSDV, the large number of routing entries in each update packet accounts for the higher overhead.

3.1.2 Cluster-head Gateway Switch Routing (CGSR)

CGSR [62] uses DSDV as the underlying routing scheme but differs from it in the type of addressing and network organization scheme. It modifies DSDV by using a cluster-head-to-gateway routing approach to route traffic from source to destination. The motivation behind the CGSR protocol is that clustered based control structures promote more efficient use of resources in controlling large dynamic networks. To achieve this, the physical network is transformed into a virtual network of interconnected node clusters. Each cluster has one or more leaders acting its behalf to make control decisions for cluster members. CGSR is a clustered multi-hop mobile wireless network with several heuristic routing schemes. Gateway
nodes are those nodes that are within communication range of two or more cluster-heads. A packet sent by a node is first routed to its cluster-head, and then the packet is routed from the cluster-head to another cluster-head through a gateway, and so on until the cluster-head of the destination node is reached. The packet is then transmitted to the destination. Each node keeps a cluster member table, where it stores the destination cluster-head for each mobile node in the network. Each node periodically broadcast these cluster member tables using the DSDV algorithm. In addition to the cluster member table, each node maintains a routing table to determine the next hop in order to reach the destination. On receiving a packet, a node will consult its cluster member table and routing table to determine the nearest cluster-head along the route to the destination. Next the node will check its routing table to determine the next hop to reach the selected cluster head. It then transmits the packet to this node. Updates are needed for both routing and cluster member tables in CGSR.

However, the frequent cluster head changes can adversely affect the performance of the routing protocol as the nodes become busy in cluster-head selection rather than relaying the packets. To avoid invoking cluster-head reselection every time the cluster membership changes, a least cluster change (LCC) algorithm is introduced. Using the LCC algorithm, cluster heads only change when two cluster-heads come into contact or when a node moves out of all other cluster heads.

3.1.3 Wireless Routing Protocol (WRP)

In WRP [63], each node in the network is responsible for maintaining four tables, which are Distance table, Routing table, Link-cost table, and Message Retransmission List (MRL) table. Nodes learn of the existence of their neighbors from the receipt of acknowledgements.
and other messages. Flow of 'hello' messages within specified time intervals are used to ensure connectivity among the nodes. Mobiles inform each other of link changes through the use of update messages. An update message is sent only between neighboring nodes and contains a list of updates, which include the destination, the distance to the destination and the predecessor of the destination, and as well as a list of responses indicating which mobiles should acknowledge the update. A retransmission of an update message is based on the Message-retransmission-List table records. Each entry in this table contains the sequence number of the update message, a retransmission counter, and an acknowledgement required flag vector with one entry per neighbor and a list of updates sent in the update message.

In its distance table, a node A keeps track of the distances to every destination node via the neighboring node, N, the downstream neighbor of node N. The routing table of A contains the distance to each destination node from node A, the predecessor and the successor of node A on this path, and a tag to identify if the entry is a simple path, a loop, or invalid. The upstream and downstream nodes are kept to check the link consistency and loop-freed property of the routes. The link-cost table is used to keep the costs of links to the neighboring nodes with the number of time-outs since the last communication with the nodes. If a node dose not have any change in its links’ states, it broadcasts a hello message after a time-out period to ensure connectivity. MRL is used to keep track of the acknowledgements for the update messages received from the neighboring nodes. Each entry in the MRL has a sequence number of the update message, a retransmission counter, and an ack-required flag for each of the neighbors of the node.
3.2 On-Demand Algorithms

On-demand protocols are proposed to overcome the disadvantages of table-driven protocols. With these protocols routers maintain path information for only those destinations that they need to contact as source or as a relay of information. On-demand routing has two major components: route discovery and route maintenance. When a node requires a route to a destination, it initiates a route discovery process. The route discovery function requires a source to use some form of flooding. Upon receiving a query, the transit nodes learn the path to the source and enter the route in their forwarding tables. The destination node responds using the path traversed by the query. Each discovered route is stored in the route cache for a short lifetime. Route maintenance is responsible for reacting to topological changes in the network, and its implementation differs from one algorithm to another. Some popular on-demand protocols are outlined below.

3.2.1 Dynamic Source Routing (DSR)

The DSR [64] protocol uses source routing to deliver data packets. Routes are stored in a route cache, and each cache entry contains the entire path to be traversed to the destination. When a data packet is originated, the source places the entire path in the packet header. The intermediate nodes along this path simply forward the packet to the next hop specified in the header. Avoiding routing loops is clearly trivial with the use of source routing.

If a source dose not have a route to the destination in its cache, it begins a route discovery process by broadcasting a route request (RREQ) packet. Each node receiving the RREQ searches its own route cache for a route to the requested destination. If no route is found, it adds its own address to the hop sequence contained in the RREQ header and broadcasts the
RREQ again. A RREQ is tagged with an identification number that each node records so that it will not broadcast the request more than once.

The RREQ propagates through the network until it reaches either the destination or an intermediate node which has a route to the destination in its route cache. The RREQ header contains a record of the hops taken from the source, so this route can be reversed and used to unicast a route reply (RREP) packet back to the source. In the case that bi-directional links cannot be assumed, the RREP is piggybacked on a new request for a route to the source. If an intermediate node is unable to forward a data packet to the next hop in its source route, it unicasts a route error (RERR) packet back to the source informing it of the broken link. The source removes the broken link from its route cache and all routes containing this hop are truncated at the point of the broken link. Any transit node that forwards the RERR will learn of the broken link and remove it from its route cache as well. The source can then attempt to use another route to the destination if one exists in the route cache, or it can initiate a new route discovery.

There is no mechanism in DSR by which a stale route can be expired, nor is DSR able to choose the freshest route when multiple choices are available in the cache. If stale routes are used, they may cause other caches to become polluted. The use of promiscuous listening coupled with node mobility can result in stale routes polluting caches faster than they can be deleted by route error packets. A detailed discussion of DSR's stale route problem is presented in [65].

3.2.2 Temporally Ordered Routing Algorithm (TORA)

TORA [66] is a distributed routing protocol based on a “link reversal” algorithm that finds
and maintains routes via local relaxation of link direction. It was designed to discover routes
on demand, provide multiple routes to a destination, establish routes quickly, and minimize
communication. Longer routes are considered more important than route optimality to avoid
the overhead of discovering newer routes.

When a node needs a route to a particular destination, it broadcasts a QUERY packet
containing the address of the destination for which it requires a route. This packet propagates
through the network until it reaches either the destination, or an intermediate node having a
route to the destination. The recipient of the QUERY then broadcasts an UPDATE packet
listing its height with respect to the destination. As this packet propagates the neighbor from
which the UPDATE was received. This has the effect of creating a series of directed links from
the original sender of the QUERY to the node that initially generated the UPDATE.

When a node discovers that a route to a destination is no longer valid, it adjusts its height
so that it is local maximum with respect to its neighbors and transmits an UPDATE packet. If
the node has no neighbors of finite height with respect to this destination, then the node instead
attempts to discover a new route as described above. When a node detects a network partition,
it generates a CLEAR packet that resets routing state and removes invalid routes from the
network. TORA is layered on top of IMEP, the Internet MANET Encapsulation Protocol,
which is required to provide reliable, in-order delivery of all routing control messages from a
node to each of its neighbors, plus notification to the routing protocol whenever a link to one
of its neighbors is created or broken. The key design concept of TORA is the localization of
control messages to a very small set of hosts near the occurrence of a topological change. To
achieve this, hosts need to maintain routing information about adjacent hosts.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
TORA's metric comprises five elements: (1) logical time of link failure (2) the unique id of the node that defined the new reference level (3) a reflection indicator bit (4) a propagation ordering parameter and (5) the unique id of the node. The first three elements collectively present the reference level. A new reference level is defined each time a node loses its last downstream link due to a like failure. Timing is an important factor for TORA because the height metric is dependent on the logical time of a link failure, and TORA assumes all links have synchronized clocks, accomplished via an external source such as the Global Positioning System. For link status sensing and maintaining a list of a node’s neighbors, each IMEP node periodically transmits a BEACON packet, which is answered by each node hearing it with a HELLO packet.

3.2.3 Associativity-Based Routing (ABR)

The Associativity-Based Routing (ABR) [67] protocol introduced a new metric known as degree of association stability. In ABR, a route is selected based on the degree of association stability of mobile hosts. Each node periodically sends a beacon to signify its existence. When received by neighboring nodes these beacons cause their associativity tables to be updated. For each beacon received, the associativity tick of the current node with respect to the beaconing node is incremented. Association stability is defined by connection stability of one node with respect to another node over time and space. A high degree of association stability may indicate a low state of node mobility, while a low degree may indicate a high state of node mobility. A fundamental objective of ABR is to derive longer-lived routes for ad hoc mobile networks.

ABR includes three phases (1) route discovery (2) route reconstruction (RRC), and (3)
route deletion. A node accomplishes a route discovery by a broadcast query (BQ message) and
delay-reply cycle (BQ-REPLY). All nodes receiving the query append their addresses and their
associativity ticks with their neighbors. A successor node erases its upstream node neighbors
associativity ticks entries and retains only the entry concerned with itself and its upstream node.
In this way, each resultant packet arriving at the destination will contain the associativity ticks
of the nodes along the route to the destination. The destination then selects the best route based
on the associativity ticks along each of the paths. When multiple paths have the same overall
degree of association stability, the route with the minimum number of hops is selected. The
destination then sends a REPLY packet back to the source along this path. Nodes propagating
the REPLY mark their routes as valid. All other routes remain inactive, and possibility of
duplicate packets arriving at the destination is avoided.

The route reconstruction phase may consist of partial route discovery, invalid route erasure,
valid route updates, and new route discovery, depending on which node(s) along the route
move. Movement by the source results in a new BQ-REPLY process. The route notification
message is used to erase the route entries associated with the downstream nodes. When the
destination moves, the immediate upstream node erases its route and determines if the node is
still reachable by localized query process. If the destination receives the localized query packet,
it REPLYs with the best partial route; otherwise the initiating node times out and the process
backtracks to the next upstream node. A route notification message is sent to the next upstream
node to erase the invalid route and inform this node that it should invoke the localized query
process.

When a discovered route is no longer desired, the source node initiates a route delete (RD)
broadcast so that all nodes along the route update their routing tables. The RD message is propagated by a full broadcast as opposed to a direct broadcast, because the source node may not be aware of any route node changes that occurred during RRCs.

The following table shows a contrast between On-demand source initiated protocol and a Table-driven protocol. It is clear from the function of Table driven protocol that it is a pro-active protocol, where as on-demand source initiated protocol is basically a reactive protocol.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>On-demand Routing Protocol</th>
<th>Table-driven Routing Protocol</th>
</tr>
</thead>
<tbody>
<tr>
<td>Routing Information</td>
<td>Available when needed</td>
<td>Always available regardless of need</td>
</tr>
<tr>
<td>Periodic update of tables</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Mobility</td>
<td>Uses localized route discovery</td>
<td>Always informs other nodes to maintain consistent routing table</td>
</tr>
<tr>
<td>Signaling traffic generated</td>
<td>Grows with increasing mobility</td>
<td>Greater than on-demand routing</td>
</tr>
<tr>
<td>QoS (Quality of Service)</td>
<td>Few can support QoS</td>
<td>Shortest Path as a major QoS metric</td>
</tr>
</tbody>
</table>

3.3 Ad-hoc On-Demand Distance Vector Routing (AODV)

The Ad-hoc On-Demand Distance Vector Routing protocol (AODV) [68] is widely known as a pure on-demand route acquisition system. AODV is essentially a combination of both DSR and DSDV. It borrows the basic on-demand mechanism of route discovery and route maintenance from DSR, plus the use of hop-by-hop routing, sequence numbers, and periodic beacons from DSDV. Nodes that do not lie on active paths neither maintain any routing
information nor participate in any periodic routing table exchanges. Moreover, a node does not have to discover and maintain a route to another node until the two need to communicate, unless the former node is offering its services as an intermediate forwarding station to maintain connectivity between two other nodes.

Route discovery in AODV follows a route request/route reply query cycle. When a node needs a route to a destination, it broadcasts a “Route Request” (RREQ). Any node with a current route to that destination (particularly the destination itself) can send a “Route Reply” (RREP) back to the source code. Route information is maintained by each node in its route table. Information gathered through RREQ and RREP messages is kept with other routing information in the route table. AODV uses sequence numbers to eliminate stale routes and keep loop-free. Routes with old sequence numbers are aged out of the system.

3.3.1 AODV Routing Tables

Each mobile node maintains a route table entry for each destination of interest. Each entry contains the following fields:

- Destination node address
- Next hop address
- Number of Hops to destination
- Sequence number for the destination
- Active neighbors for this route
- Expiration time for the route table entry
- Last Hop Count

A neighbor is considered active, if it originates or relays at least one packet for that
destination within the most recent active timeout period. A route entry is active, if it is in use by any active neighbors. The path from a source to a destination, which is followed by packets along active route entries, is called an active path. The active timeout period is reset each time a route entry is used to transmit data. The route entry updates are based on higher destination sequence numbers or less number of hops to the destination. New entries are placed in the route table upon the reception of RREQs and RREPs. When a node receives RREQs or RREPs and it dose not already have a route entry for the source of the message, it places an entry in the table listing the indicated information.

Associated with each destination is a sequence number. If a node receives routing information for a destination, and it already has a route table entry for that destination, it updates the entry only if the destination sequence number associated with the new information is greater than that already contained in the route table entry. The use of sequence number guarantees that routing loops cannot from, even under extreme conditions of out-of-order packet delivery and high node mobility. Additionally, each entry has a corresponding lifetime that indicates the length of time the route entry will expire. Beyond this period, the routes are invalidated if they are not updated or used. When a route entry expires, the Hop Count value is copied into the “Last Hop count” field, and the Hop Count is set to \( \infty \). The Destination Sequence Number field is also incremented.

3.3.2 Route Discovery

This thesis will be focus on the unicast communication in the protocol, although AODV provides unicast, multicast and broadcast communication. Route discovery with AODV is on-demand and occurs when a node requires a route to a destination for which it dose not have
a recorded route. The node initiates route discovery by broadcasting a RREQ packet. The message format of the RREQ is as follows:

<Flags, Hop_Count, Broadcast_ID, Source_Addr, Source_Seq#, Dest_addr, Dest_Seq#>

The only flag used for unicast is the “Gratuitous RREP” flag. The source of the RREQ sets this flag when a node generating a RREP for this RREQ should send a gratuitous RREP to the destination node. The “Hop_Count” field is initialized to zero and incremented each time the RREQ is forwarded. Each node in the network is responsible for maintaining two separate counters: a sequence number and a broadcast ID. The sequence number ensures the freshness of routes to the node. The broadcast ID, together with the source node’s IP address, uniquely identifies each RREQ. The sequence number is incremented when the node acquired new neighbor information, and the broadcast ID is incremented for each RREQ the node initiates.

A node receiving a RREQ first updates its route table to record the sequence number and next hop information for the source node. This reverse route entry may later be used to relay a RREP back to the source. The node then checks this table to see whether it has a route to the requested destination. In order to respond to a RREQ, a node must either be the destination itself, or must have an unexpired route to the destination with a sequence number at least as great as that indicated in the “Dest_Seq#” field of the RREQ. A node having such a route generates a RREP. Otherwise, it rebroadcasts the packet to its neighbors. Fig.3.1 illustrates the propagation of the RREQ throughout the network. A node may receive the same RREQ multiple times. When a node receives a RREQ, it records the source IP address and broadcast ID of the packet. If it later receives a RREQ with the same information, it does not process the packet but instead discards it.
3.3.3 Reverse Path Setup

As stated above, when the RREQ travels from a source to various destinations, it automatically sets up the reverse path from all nodes back to the source node S as illustrated in the Fig. 3.2 [68]. To set up a reverse path a node records the address of the neighbor from which it received the first copy of the RREQ.

The freshness information about the reverse route is indicated by the source sequence
number, and the destination sequence number specifies how fresh a route to the destination must be before it can be accepted by the source. The reverse path route entries are maintained for at least enough time for the RREQ to traverse the network and produce a reply back to the sender.

3.3.4 Forward Path Setup

When a node satisfies the requirements that it is the destination of the RREQ or it has a current route to the destination, the node sends a RREP back to the source node. The RREP message format is as follows:

\[<\text{Flags, Hop\_Count, Source\_Addr, Dest\_Addr, Dest\_Seq\#, Lifetime}>\]

The Acknowledgment flag is set in cases where the link over which the RREP message is sent may be unreliable. It indicated that a RREP-ACK message should be transmitted by the receiving node to acknowledge receipt of the RREP message. The Dest\_Addr field is set to the destination address specified in the RREQ, and “Dest\_Seq\#” is set to the responding node’s record of the destination’s sequence number. The “Hop\_Count” field is set to the distance of the responding node from the destination or zero if the destination itself sends the RREP. The responding node sends the RREP to the next hop towards the source node. The node receiving the RREP increments the “Hop\_Count” field by one and then creates or updates its entry for the destination node in its route table, thereby establishing the forward path to the destination. It then sends the RREP to its recorded next hop for the source node. This continues until the RREP reaches the source node. Fig. shows the forward path formation. Nodes that are not along the path determined by the RREP will timeout after a set time and will delete the reverse pointers.
A node receiving the RREP propagates the first RREP to the source node. If it receives further RREPs, it updates its routing information and propagates it only if the RREP contains higher destination sequence number. The source node can begin data transmission as soon as the first RREP is arrived.

3.3.5 Route Maintenance

If the source node moves during an active session, then a new route discovery process is initiated to find the new route to the destination. If the link breakage occurs due to the movement of the destination or an intermediate node, then a route error (RERR) message conveys the same to the affected source nodes. Upon receiving notification of a broken link, source nodes can restart a route discovery process if they still require a route to the destination. This procedure is illustrated in Fig 3.4. As in figure, the original path from the source to the destination is through nodes 1, 2 and 3. Suppose node 3 then moves away, causing a link
failure. Node 2 notices this break in connectivity and sends RERR message to node 1, node 1 marks this route as invalid and then forwards the RERR to the source. If the source node still needs a route to the destination, it may find a new route through node 4 as shown in figure (b).

Figure 3.4. AODV Route Maintenance

3.3.6 Local Connectivity Management

In AODV, the local connectivity management is accomplished mainly through the regular hello messages between the neighboring nodes. To utilize Hello messages, a node broadcasts to its neighbors a Hello message if it has not transmitted anything within the last “hello_interval” msec. This informs its neighbors that the node is still within their transmission range. A Hello message is a special unsolicited RREP which contains a node's IP address and current sequence number. The Hello message is prevented from being rebroadcast outside the neighborhood of the node because it contains a TTL value of 1. Neighbors that receive this packet update their local connectivity information to include the node. The failure to receive any transmissions from a neighbor in the time defined by the periodic transmission of
"allowed_hello_loss" Hello message is an indication that the local connectivity has changed, and the route information for this neighbor should be updated.
CHAPTER 4

AN IMPROVED MULTIANTS-AODV ROUTING ALGORITHM

As stated in chapter 1, the NANET has a nature of a dynamic topology, which makes the conventional algorithms discussed in chapter 3 not applicable to be realized. The conventional proactive routing protocols that require knowing the topology of the entire network is not suitable in such a highly dynamic environment, since the topology update information needs to be propagated frequently throughout the network. Hence, the pure proactive schemes are likewise not appropriate for the MANET environment, as they continuously use a large portion of the network capacity to keep the routing information current. On the other hand, a demand-based, reactive route discovery procedure generates large volume of control traffic and the actual data transmission is delayed until the route is determined. Because of this long delay and excessive control traffic, pure reactive routing protocols may not be applicable to real-time communication.

4.1 Ant-based Routing Algorithm For MANETs

The mobile agents based paradigm provides a novel solution for such difficulties. The agents hop from node to node, collect information from these nodes, interact with other agents
directly or indirectly, and gift these collected data sets to newly visited nodes. They are particularly attractive in dynamic network environment involving partially connected computing elements. They can help to achieve two general goals: reduction of network traffic and asynchronous interaction, because the number of messages in MAs based model is bounded by the number of constituent agents in the network.

The idea of using mobile agents for routing purposes in dynamic ad-hoc networks has been explored in MIT Media Lab [69] a few years earlier. Chpudhury et al. [70] followed the same direction by proposing a distributed mechanism for topology discovery in ad-hoc wireless networks using mobile agents. In their work they try to overcome certain limitations observed in MIT's research. Unlike the NetAnt algorithm which is appropriate for the fixed network, the forward routing mechanism is adopted in both systems, instead of the round trip routing, due to the intrinsically slow nature in the latter one. Ants in network routing applications are simple agents embodying intelligence and moving around in the network from one node to another, updating the routing table on the nodes with the knowledge which they have learnt in their traversal so far, as shown in Fig. 4.1.
Routing ants keep a history of the nodes previously visited by them. Upon arriving at a node, the ant uses the information stored in the history memory to update the routing table at that node with the best routes which it has for the other nodes in the network. The size of the history window is an important parameter: the longer the history, the higher the overhead of moving the agent, and also small size will give rise to the problem of the efficiency on the employ of the agents’ traversal. Hence the history size of the ants needs to be carefully decided, and it was fully investigated in [69]. All the nodes in the network rely on the ants for providing them the routing information, as they themselves do not run any program for finding the routes, and they simply host agents and provide a place to store a database of routing information. Hence in this approach, the route discovery is manifested in the movement of agents carrying routing information from one node to another rather than the propagation of individual update messages. As discussed in chapter 2, an agent can be formally described as:

\[ A(i, N_x, N_y, R_x, \mu) \]
Where $A$ is an agent with ID $i$ migrating from node to $N_x$, node $N_y$, carrying the routing information $R_x$ of $N_x$ and using the navigation strategy $\mu$ to move among adjacent nodes. Considering the route discovery is totally rely on the agent migration in this approach, the navigation strategy adopted by the MAs based protocols substantially determine the efficiency of the algorithm. Hence it is very important that the agents navigate intelligently, otherwise an imprudent strategy can severely affect the performance of the algorithm. The significance of the navigation strategy will be demonstrated based on the comparisons in section 4.3.

The population size is also an important parameter: the more routing agents, the higher the overhead. However, with more a larger population, there are more agents to look for routes, and the gain from adding more agents can decline the history size. Moreover, more agents also narrows the spread between maximum and minimum connectivity, and having some often-redundant agents can help lend stability to the network [69].

4.2 An ANT-AODV Hybrid Routing Protocol For MANETs

Although deploying mobile agents has several advantages in the ad-hoc environment due to their flexible, robust and autonomous nature, and they can be used for efficient routing in a network and discover the topology to provide high connectivity at the nodes. Nevertheless, the ant-based algorithms in wireless ad hoc networks have certain drawbacks. In that the nodes depend solely on the ant agents to provide them the routes to various destinations in the network. This may not perform well when the network topology is very dynamic and the route lifetime is short. In pure ants based routing, the mobile nodes may have to wait to start a communication till the ants provide them with the routes. In some situations it may also happen that the nodes carrying the agents suddenly get disconnected with the rest of the
network. This may be due to their movement away from all other nodes in the network or they might go into sleep mode or simply turned off. In such situations, the amount of ants left for routing are reduced in the network which could lead to ineffective routing.

To overcome these inherent drawbacks of ant routing, another very encouraging research by Marwaha et al. [71] proposes a combination of Ad-Hoc On-Demand Distance Vector (AODV) with the distributed topology discovery mechanism using ant-like mobile agents. The results of the research show that their scheme achieves reduced end-to-end delay compared to conventional ant-based and AODV routing protocols. The hybrid technique enhances the node connectivity and decreases the end-to-end delay and route discovery latency [71].

In conventional ant-based routing techniques, route establishment depends on the ants visiting the node and providing it with routes. If a node wishes to send data packets to a destination for which it does no have a fresh enough route, it will have to keep the data packets in its send buffer until an ant arrives and provides it with a route to that destination. Ant-AODV utilizes ants working independently and providing routes to the nodes as shown in Fig.4.2. The nodes also have capability of launching on-demand route discovery to find routes to destinations for which they do not have a fresh enough route entry. The use of ants with AODV increases the node connectivity, which in turn reduces the amount of route discoveries. Even if a node launches a RREQ for a destination it does not have a fresh enough route, the probability of its receiving replies quickly (as compared to AODV) from nearby nodes is high, in that the increased connectivity of all the nodes resulting in reduced route discovery latency. Especially, as ant agents update the routes continuously, a source node can switch from a longer and stale route to a newer and shorter route provided by the ants. This leads to a
considerable decrease in the average end-to-end delay as compared both AODV and ant-based routing [71].

![Diagram of Ant-AODV routing protocol]

Figure 4.2. A demonstration of Ant-AODV routing protocol

4.3 An Improved ANT-AODV Hybrid Routing Protocol For MANETs

But unfortunately, the ant-based routing algorithm proposed by Marwaha did not exert the advantage of the multiple agents. They ignored the forte of the inter-agents communication. The simulation results also present this shortcoming in the protocol. Although the MA’s navigation strategy plays a key role on the efficiency, the proposed protocol just simply adopted the conventional random scheme in ants-based routing. Considering two separate protocols running simultaneously, the entries in the routing table become critical resources in the situation of updating the routing table. To prevent the conflicts, an altered structure of routing table is used in this thesis. In next two subsections, an improved Ant-AODV routing protocol will be explained in detail.
4.3.1 Navigation Algorithms

As stated above, in the ant agents based routing protocol, to establish routes for every pair of nodes in the network entirely relies on the agents’ migrations among the nodes. Hence the efficiency of the routing protocol, in terms of the route discovery, is characterized by the navigation strategy of the agents.

So far, there are two navigation strategies mainly used by agents based routing protocols for MANETs. The first one is based on random selection, in which the next node is selected randomly. This strategy is widely used in the conventional approaches due to its simplicity, and was explored in MIT Media Lab earlier [69]. In [72], Matsuo proposed a little revised version based on the random scheme. An idea of “no return rule” was adopted to speed the convergence, since if the next hop selected is the same as the previous node from which the agent leaves to the current node, this route would not be optimal. As shown in Fig.4.3, this technique is used to eliminate the detours in the dynamic network.

![Figure 4.3. No return rule](image-url)
The second approach was also proposed by Minar in [69], in which a type of “oldest-node” agent adopts a strategy that it preferentially visits the adjacent node it last visited longest ago (or never visited, or doesn't remember visiting). The backtracking also can be avoided by consulting the history information carried by agents. At the same time, “oldest-node” agents try to jump onto the nodes they have never been on first. This gives rise to an inclination that all agents attempt to diffuse as widely as possible, as soon as the migration process starts. As a result of this tendency, the agents can collect the topology information and spread the information to all mobile nodes widely and quickly, other than blocking in some region and missing the chances to advertise the recent topology information to others timely. The greatest advantage of this strategy is to shorten the period of convergence on the process of route discovery. Meanwhile, it also accomplishes the function of “no return rule”. Minar also called the agent using “oldest-node” strategy as “conscientious agent” in [73]. The experimental result on the comparison between these two types of agents is the same as the intuition. The “conscientious agent” is much more efficient than the “random agent”. The idea of the second strategy can be shown in Fig. 4.4 followed.
Even though the "conscientious agent" navigation strategy improves the performance of the route discovery a lot over the random approach, it still can not achieve a good performance in some scenarios, such as in the case that all the neighbors nodes, from which the agent will select as the next hop, are already stored in the agent's history memory. Following the strategy of "conscientious agent", it has to reinitiate another random selection on the nodes pool. Even considering the "no return rule" applied here, the randomly selected node may not be the good one at that moment. The direction to improve the navigation algorithm is derived from the experimental results in [73].

In [73], Minar also did a comparison between the "conscientious agent" and "superconscientious agent". The type of "superconscientious agent" adopts the same navigation strategy as the "conscientious agent", but it has the capability to communicate with other agents to exchange the topology knowledge, and it uses its own first-hand knowledge and
learned data from its peers in deciding which nodes to move to. In a common sense, the second method is better than the previous one, since it uses the potential of inter-agent communication and after all, the more information that is factored into a decision the better that decision should be. But the result is surprising. It turns out that in small populations “superconscientious agents” do perform the best, but only just barely, and with the populations increase, the “superconscientious agents” perform worse than the “conscientious agent”. This is because “superconscientious agents” tend to duplicate one another’s exploratory efforts and tend to cluster together, while “conscientious agents” remain evenly dispersed. As a result, the whole system based on “superconscientious agents” is less efficient. This result demonstrates that in the agents based system, the whole performance is dependent on the behavior of the all of the agents. Hence a good navigation algorithm should make the agents spread in the network evenly, and keeps the varieties of the multiple agents. Based on this idea, and inspired by the potency of “stigmergy” (a method of indirect inter-agent communication), an improved navigation strategy is proposed in below.

An agents migration information table (AMIT) stored on each node is used to record the history of the agents’ traversal. When an agent decides which neighbor is the next hop to move to, it writes the ID of the neighbor node into the table before it will jump to. As soon as the agent leaves, a timer needs to work for that record, to indicate how old that entry is right now. This freshness information is used by the later landing agents to select the next hop. All agents will choose the neighbor node with the highest value on the time field (the oldest entry). This can guarantee that multiple agents from the same mobile node do not visit the same neighbor nodes consecutively. In the situation of topology changes, if a mobile node detects the new
emerging neighbors, it initiates associated records for those nodes and sets the time value \( \infty \), and if it detects some neighbors are disconnected, it removes the records associated with those nodes in the AMIT. When an agent finds several \( \infty \) in the time field, it randomly selects one as the next hop.

The advantage of introducing such a table on nodes is that with the indication of the migration freshness on the selected neighbor nodes, the agents which are choosing the next destination can get the information of other agents' choices, and disperse themselves to other places to avoid clustering together, and this can improve the whole system's performance. Another very important benefit is that this algorithm makes the agents land on the nodes with topology changes in a high priority. As discussed in previous chapters, the most challenging characteristic of MANETs to design an efficient routing protocol is their highly topology changes at all time. More prompt response on the topology changes, more efficient a routing protocol will be. All proactive protocols attempt to react all changes timely, even before the real data's transmission, but the overhead brought by this idea is so huge that they may not be appropriate for such an environment. In contrast, the reactive protocols may take a long delay to establish the connection. Nevertheless, the adoption of freshness' indication makes the agents react the topology changes as soon as possible. Since when a new neighbor node B emerges to a node A, it will appear on the AMIT with the largest value \( \infty \) on A, the agents landing on node A will choose the recent emerging neighbor B first, after it migrate to B from A, all the routing information in node A's region will be brought to new connected B's region. The same as in the opposite direction, the agent from B arriving on node A can introduce the current routing information about B's region to A. The prompt exchange of routing information
could be very essential for a data packet being delivered to some node in A’s region through node A. Without the quick response like this, the opportunity for the packet to be delivered to A may be missed due to the delay. This advantage can be observed by the simulation’s result which will be discussed in next chapter.

The new proposed navigation algorithm can be described in pseudo code as follows:

Variables carried by a routing agent $j$ are:

1. $NHT_j$—Navigation History Table stored in a structure of stack like

\[
\{ \text{NodeID 0, NodeID 1, ...} \}, \quad \text{initialization } NHT_j := \{ \text{StartNodeID} \};
\]

2. $NEXT_j$—the node ID of next hop which will be selected and visited

Variables stored in a mobile node $i$:

1. $AMIT_i$—Agents Migration Information Table in a structure of vector, each element in the vector is like

\[
\{ \text{NodeID, LivedTime} \}, \quad \text{initialization } AMIT_i := \emptyset
\]

2. $NT_i$—Neighbors Table in a structure of vector like

\[
\{ \text{NodeID 0, NodeID 1, ...} \}, \quad \text{values are returned from the low level protocol}
\]

Program for the agent $j$ while visiting node $i$:

agent variables $NHT_j, NEXT_j$, node variables $AMIT_i, V_i, \text{oldest, next}_j$

Upon arriving at a node $i$:

- Update the routing table stored on the node;

/*Select the next hop to move to*/
Initialization

\[
V_i := \emptyset;
\]

/*the First part*/
For each neighbor $n \in \text{AMIT}_i$ do
  If ($n \notin \text{NHT}_j$) then
    $V_i := V_i \cup n$;
  Endif
Done
If $V_i \neq \emptyset$ then
  NEXT$_j := \text{a random element in } V_i$;
Else /*the Second part*/
  For each neighbor $n \in \text{AMIT}_i$ do
    oldest := 0;
    If $\text{AMIT}_i(n).\text{LivedTime} > \text{oldest}$ then
      oldest := $\text{AMIT}_i(n).\text{LivedTime}$;
      NEXT$_j := n$;
      $V_i := \emptyset$;
    Else if $\text{AMIT}_i(n).\text{LivedTime} = \text{oldest}$ then
      $V_i := V_i \cup n$;
    Endif
  Endif
  Done
Endif
If $V_i \neq \emptyset$ then
  NEXT$_j := \text{a random element in } V_i$;
Endif
Push NEXT$_j$ into NHT$_j$

next$_j := $NEXT$_j$

Once an agent lands on a node, it updates the routing table stored on the node first. This procedure is not addressed here, and we'll concentrate on the navigation algorithm. After the agent finishes updating the routing table, it needs to calculate the next hop. First, it checks the AMIT on the current node to see if there exist some nodes it hasn't visited yet (those nodes' IDs are not stored in its NHT table) or it forgets (the nodes' IDs are thrown away due to the size of the history memory). If multiple nodes are found, only one node is selected randomly from them. The unvisited node chosen if it exists is the next hop, and the agent records the selection in the node's variable "next$_j"", which will be fetched by the node soon. To put the unvisited nodes on the first position to concern can guarantee all agents tend to traverse all
nodes in the network. The first part in this algorithm is the same as the "conscientious agent" [69] or "oldest-node" algorithm proposed by Minar [73].

The second part is the difference from Minar's "conscientious agent" migration algorithm, in which the agents just randomly select a node from the all-visited neighbors. In this algorithm, the second level selection continues. Considering after the protocol runs for some time, most agents already visit all nodes, the "conscientious agent" migration algorithm is not good any more to keep the performance of the routing algorithm. A revised algorithm needs to be applied here. The second part above in the algorithm can satisfy this requirement. It proceeds to the second level, in which the agent retrieves the node with the oldest time field or an emerging node with a time value of "\(\infty\)". The benefits brought by this part have been stated above. One point needs to be addressed here is that the AMIT table is playing a role as the pheromone, by which an agent can inform other agents its behavior at this moment. But in contrast to the conventional usage of the pheromone, in which it is used to make more agents attracted to follow the agent laying the pheromone, the AMIT table is used to disperse the following agents evenly in the network (different intentions but the same method—indirect inter-agent communication).

Besides the agents' program, some important jobs need to be done by each node too:

**Program for the mobile node** \(i\):

Node variables \(\text{AMIT}_i, V_i, \text{oldest}, \text{next}_i\)

Upon detecting a new neighbor \(n \in \text{NT}_i \rightarrow\)

\[\text{AMIT}_i := \text{AMIT}_i \cup n;\]

\[\text{AMIT}_i(n).\text{LivedTime} := \infty;\]
Upon losing a neighbor \( n (n \notin NT_i) \rightarrow \)
\[
AMIT_i := AMIT_i - n;
\]

Upon an agent's leaving \( \rightarrow \)
\[
AMIT_i := AMIT_i \cup next_i;
\]

Set a timer on \( AMIT_i(next_j) \). \( LivedTime \)

Because each mobile node can read the current neighbors from its Neighbor Table (NT), which is maintained by a protocol in a lower level (discussed later), it is able to track the local topology changes real-timely. If these local information can be promptly advertised by agents and used to update the routing tables, the aim to make the routing tables reflect the current topology can be achieved. The AMIT table is also used to inform the agents the changes happening in the local network. As soon as the mobile node learns the emerging neighbors, it adds an entry in the AMIT and assigns a value of \( "\infty" \) to the corresponding \( "LivedTime" \) field, which makes the coming agent jump to the new neighbors first to respond the changes. When nodes learn some disconnected neighbors, they simply remove those neighbors from the AMIT to be consistent with the NT table. Once the agent departs, the node starts the timer tracing the elapsed time and updates on the \( "LivedTime" \) field of each entry in the AMIT. Since the AMIT table is stored on each node, accessed (read) by the agents, and maintained (written) by the nodes, so it is a kind of critical resource between agent (agents) and the mobile node. A mutex mechanism needs to be applied here to protect the integrity and correction of the data.

The structure of the AMIT can be referred in table 4.1. The simulation based comparisons among the three strategies stated above are made, and will be explained in the next chapter.
Table 4.1 An example of AMIT table stored on nodes

<table>
<thead>
<tr>
<th>Neighbor node</th>
<th>LivedTime</th>
</tr>
</thead>
<tbody>
<tr>
<td>D</td>
<td>14</td>
</tr>
<tr>
<td>H</td>
<td>22</td>
</tr>
<tr>
<td>K</td>
<td>∞</td>
</tr>
<tr>
<td>C</td>
<td>5</td>
</tr>
</tbody>
</table>

(The entry with the neighbor node “K” will be selected.)

4.3.2 The Analysis Of The Improved ANT-AODV Routing Algorithm

The navigation algorithm in agents based routing protocol provides the function of the route discovery process. By replacing the “passing messages” based model with the “walking agents” based model, the routes are explored with the reduction of the bandwidth waste by control messages. Besides the route discovery part, a routing protocol for MANETs also includes the local connectivity management, the structure of routing table and the route maintenance as in other conventional algorithms.

4.3.2.1 The Local Connectivity Management In The Improved Ant-AODV

In the Ant-AODV routing algorithm, the local connectivity management is accomplished by a method of periodic broadcasts of “hello” (beacon) messages, the same as in AODV. The “hello” messages are frequent broadcasted one hop distance by each node to its neighbors only. It can be seemed as lower connectivity maintenance protocol working beneath the routing protocol. Each node uses the information from this lower protocol to update the NT table. Another attention is due to AODV can only be applied in symmetric networks, so the Ant-AODV algorithm has the same limitation.

4.3.2.2 The Structure Of Routing Table In The Improved Ant-AODV

An extended structure of routing table over the ants based protocols’ routing tables is exploited in Ant-AODV. For each destination node, multiple entries are associated with it.
instead of only one. Other than the hops count is used to evaluate the distance of the path to the specified destination, a timer attached to each entry is used to indicate the lifetime the route has been through, and to help decide whether the route is valid and should be removed. More than one route entry for each destination node brings multiple backups of the routing information for each destination node. In the situation of that the only explored route is invalid due to the neighbor of next hop moves outside the range of communication, the sending node has to initiate an AODV routing discovery procedure or wait for the coming agents to bring the routing information about the specific destination. Both ways may take a long delay to accomplish. The method of adding backup routes can avoid the delay like this, since in that case the node can find another available route even though it's not as good (short or fresh) as the previously used one. Another goodness of extending the routing table is the exclusion of the interference between the agents and AODV while updating the routing table on the same entry. Agents and AODV can work on the different entries for the same destination simultaneously without conflicts. Certainly the limit on the maximum number of the entries for one destination can be defined in different applications. In this thesis, we define this limit as two.

Besides, another important extension is the field of “active upstream node”, which presents the upstream neighbor node in the data flow route when this route is being used. This field has the same function as the “Active Neighbors in the route” in AODV, and it is used and updated in the same situation as in AODV (discussed in detail in next section). An example of the routing table is shown in table 4.2.
### Table 4.2 An example of routing table with two entries for each destination

<table>
<thead>
<tr>
<th>Destination</th>
<th>Next Hop</th>
<th>Next Hop</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>hops</td>
<td>Time</td>
</tr>
<tr>
<td>B</td>
<td>D</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P</td>
<td>O</td>
<td>12</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

4.3.2.3 The Route Maintenance In The Improved Ant-AODV

As in AODV, the route error messages (RERR) are still used in Ant-AODV, but they do not only work for those routes built by the AODV protocol. Due to the introduction of the “active upstream node” field, the concept of the “source initiated” request is also introduced in agents based routing protocol, in which it has never been discussed before. Each node which intends to send a packet can be deemed as the “source” node as in AODV, DSR or other source-initiated on-demand protocols. After it sends the packet to one of its neighbor following the routing table, the receiver node deems the initiator as the “active upstream node”. Because the “source address” and “destination address” are also included in the data packets, all intermediate nodes know where the packets are from and where they will go. Hence along the path the packet follows, each node on the path can record its predecessor respectively, and this information is stored in the “active upstream node” field in the routing table. In a similar fashion, each node is also able to record the downstream neighbor node: after sends the packet following some certain entry, flags this entry as an active downstream neighbor node, and reset the “time to expire” to 0 (the most fresh indication).

A route entry is considered active if it is in use by any active neighbors, and the path from a source to a destination, which is followed by packets along active route entries, is called an
active path [68]. The recorded predecessor information can be used to update the routing table when broken links on the active path happen. In this case, the “URERR” message is generated by the neighbor nodes next to the broken link; the node, which lost its downstream neighbor node, propagates the RERR message to each of its active upstream neighbors to inform them of the erasure of that part of the route. These nodes in return propagate this “link failure notification” to their upstream neighbors, and so on until the source node is reached. On the other side, the node, which lost its upstream neighbor node, generates the “DRERR” and sends it to all its downstream neighbors to inform them the erasure of the reverse route—from each downstream nodes to all the node’s (which initiates the “DRERR” alert) upstream nodes along the active route, until the destination node is reached. For the nodes next to the broken link themselves, while they spread the “URERR” or “DRERR”, they also need to remove the all routes associated with that lost neighbor as the “next hop” in the routing table. The process can be described as in Fig.4.5.

![Figure 4.5. An example of "Notification of Link Failures" propagation](image)

The introduction of “messages passing” model for the delivery of the “link failure notification” is able to serve well in the mobile agents based model (traditionally, they are used
separately). Since the “messages passing” model is not used in a “flood” (this method is used in AODV, DSR, etc. for route discovery), instead, the error message is only broadcasted along a path, which can only cost O(n) in time complexity or message complexity. Nevertheless, the speed of responding to the failure of the links is increased a lot, which in fact avoids the further wasteful delivery, and informs the source node (requests the packet transmission) selects another available route in its routing table. The result of this is that the standard of “packet delivery fraction” and “goodput” both get increased. “packet delivery fraction” is the ratio of number of data packets sent to the number of data packets received and “goodput” is total number of useful packets received at all destination nodes. In Marwaha’s simulation for Ant-AODV [71], the result of “packet delivery fraction” for ant-based routing is very low compared with AODV and Ant-AODV. Although Ant-AODV already has a satisfactory value on “packet delivery fraction” over ant-based routing, the comparison shows that the outcome is heavily relied on the efforts from AODV. We are trying to make the ant-based routing in the hybrid protocol exert a dominant influence on the whole performance, and to exploit the AODV part as a backup method to explore the route which is required in a rush but no agents happen to work for it (preparing for the applications in the QoS environment). A fully exertion to mobile agents can promote the hybrid protocol—Ant-AODV to work as a proactive algorithm with much less overhead on control messages at the same time. This is another main aim of this thesis.

Thus, mobile agents based routing protocols have the features between the proactive protocols and reactive protocols. The adoption of mobile agents as the messengers walking around in the network, instead of the enormous control messages, can substantially reduce the
amount of non-data packets' occupation on the limited transmission capability of the MANETs. While achieving the efficient usage of the network bandwidth, they also overcome the inherent delay shortcoming in the reactive protocols, by reacting the topology changes immediately, and even as quick as the proactive ones (the routing entries have been ready for the specific data's delivery).
CHAPTER 5

THE SIMULATION BASED COMPARISONS FOR THE

NAVIGATION STRATEGIES

The main aim of this research is to design and implement a novel navigation strategy for mobile agents based routing protocol in wireless ad-hoc networks that will provide the following benefits. Maximize network performance, scalability, provide end-to-end reliable communications and reduce possible delays. Thus a simulator based on object oriented was developed for the comparisons among the three navigations stated in last chapter. The simulator is composed of five parts, including an event scheduler, a network generator, multiple mobile agents, routing implementation, and a visual interface.

5.1 The Introduction Of The Simulator

Some parameters can be provided to the simulator, such as the number of mobile nodes, the number of agents, the moving speed of the mobile nodes, the range of the communication, the size of the simulated area and the navigation strategy employed to compare with. All the other factors are generated from several random number series, which can be kept as the same input for the comparison of different algorithms. Those factors can be the initial positions of the mobile nodes, the direction and the speed of each mobile node’s movement, and the home
nodes of the mobile agents where the mobile agents reside initially. For the reason of fairness, we provide all compared algorithms under the same environment. The event scheduler in this simulator compels each agent work in a time-tick mode synchronously. In each step, the agents do their jobs independently. They maintain their own history tables, update the routing tables on the mobile nodes, select the next hops from AMIT and jump to them carrying the history tables. Two mobile nodes are defined as source node and destination node respectively. When the simulator starts, all mobile agents start walking around following the specified navigation strategy for routes discovery and routing updates. As soon as the source node learns the routing information for the destination node, it sends out the data packet immediately to the retrieved neighbor from the routing table updated by the mobile agents. The neighbor node received the data packet will take charge of the delivery of the data packet in turn, and propagate it provided the routes information for the packet are ready. When the destination node receives the message, this run of the simulation terminates. And then the results interested can be outputted.

The interface of the simulator looks like fig. 5.1
Figure 5.1. The snapshot of the simulation

(In the snapshot, the white rectangles present the mobile nodes; the gray circles centered with the mobile nodes (white rectangles) are used to demonstrate the communication range of the mobile nodes; the green spots in the white rectangles depict the mobile agents moving around among the mobile nodes; and the red spot is describing the delivery of the data packet.)

5.2 The Comparisons Among The Three Navigation Algorithms

The comparisons simulated in this thesis are provided an environment with the parameters, which include (1) a closed area with 500x800 square meters; (2) 60 mobile nodes; (3) the range of the communication is 90 meters; (4) the speed of the mobile nodes is from 0 to 3 meters/time-step exclusively; (5) the size of the history memory each mobile agent holds is 15; (6) the discrete time-step is 50 milliseconds/time-tick; (7) the frequency of mobile agents movement 1hop/time-step.)
To evaluate the performance of the three navigation strategies, we take three associated gauges into account. The speed of the convergence on the routing information, is used to evaluate how fast the navigation strategy can explore the routes and spread them in the network; the number of total time steps when the run terminates, is considered as the whole efficiency of the navigation strategy; and the number of hops spent on delivering the data packet, is evaluated as the as the routing efficiency affected by the navigation algorithms. The simulated results are listed in the following tables table5.1, table5.2, and the comparisons among the three strategies are shown in the fig. 5.2, fig.5.3, fig.5.4.

The “RandomAgent” employs the random migration strategy; the “SmartAgent” adopts the same strategy as the Minar’s “conscientious agent” migration algorithm in [69], and the
"SmarterAgent" uses the navigation algorithm proposed in this thesis. The data are sampled in six groups, each one with different numbers of agents from 10 to 60.

Form the results in Fig.5.2, it is clear that "SmartAgent" is much faster than the "RandomAgent", since it employs the "oldest-node-first" and "no-return rule" in its migration strategy. It also obvious that "SmarterAgent" has a equivalent performance with the "SmartAgent" on the speed of the convergence, and even in the case of few agents (10 to 30), it outperforms a little over the "SmartAgent". That is because it first inherits the "SmartAgent" strategy to dispatch all agents widely, and base on this employs the agents more efficient than it through the indirect inter-agent communications (stigmergy). However, with the increase of the number of the mobile agents, the advantage goes down.

![Total Steps Comparison](image)

Figure 5.3. The comparison of the total steps taken to deliver the data packet

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
Figure 5.4. The comparison of the actual hops taken to deliver the data packet

Because all simulations made to compare adopt the same routing algorithm, the only factor affecting the whole routing efficiency is the navigation strategy. The result is very positive for the “Smarter Agent”, no matter the total steps or the actual hops taken to deliver the data packet. These results show that the “SmarterAgent” is much more efficient than the “SmartAgent” (37% better). With the application of the “SmarterAgent” in the hybrid routing algorithm, the whole performance can be improved a lot. Especially for the situations with few agents, the better performance is more apparent, which in return can save more MANETs’ scarce resource to achieve a better whole performance.

The results of the comparison can satisfy our expectation on improving the performance of the hybrid Ant-AODV routing algorithm through improving the performance of the ants based routing algorithm which is not fully exploited in Marwaha’s original Ant-AODV [71].
Table 5.1. Sample data for the comparisons (10-30) mobile agents

<table>
<thead>
<tr>
<th></th>
<th>Agent Number 10</th>
<th>Agent Number 20</th>
<th>Agent Number 30</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Random</td>
<td>Smart</td>
<td>Smarter</td>
</tr>
<tr>
<td>convergence</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>315</td>
<td>315</td>
<td>50</td>
<td></td>
</tr>
<tr>
<td>309</td>
<td>54</td>
<td>42</td>
<td></td>
</tr>
<tr>
<td>347</td>
<td>324</td>
<td>43</td>
<td></td>
</tr>
<tr>
<td>356</td>
<td>54</td>
<td>321</td>
<td></td>
</tr>
<tr>
<td>335</td>
<td>45</td>
<td>321</td>
<td></td>
</tr>
<tr>
<td>348</td>
<td>41</td>
<td>49</td>
<td></td>
</tr>
<tr>
<td>363</td>
<td>50</td>
<td>47</td>
<td></td>
</tr>
<tr>
<td>315</td>
<td>328</td>
<td>55</td>
<td></td>
</tr>
<tr>
<td>353</td>
<td>51</td>
<td>316</td>
<td></td>
</tr>
<tr>
<td>332</td>
<td>315</td>
<td>319</td>
<td></td>
</tr>
<tr>
<td>355</td>
<td>321</td>
<td>42</td>
<td></td>
</tr>
<tr>
<td>384</td>
<td>59</td>
<td>316</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total Steps</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>326</td>
<td>577</td>
<td>531</td>
<td></td>
</tr>
<tr>
<td>316</td>
<td>327</td>
<td>51</td>
<td></td>
</tr>
<tr>
<td>390</td>
<td>572</td>
<td>68</td>
<td></td>
</tr>
<tr>
<td>412</td>
<td>886</td>
<td>358</td>
<td></td>
</tr>
<tr>
<td>380</td>
<td>51</td>
<td>351</td>
<td></td>
</tr>
<tr>
<td>351</td>
<td>84</td>
<td>60</td>
<td></td>
</tr>
<tr>
<td>365</td>
<td>331</td>
<td>57</td>
<td></td>
</tr>
<tr>
<td>326</td>
<td>601</td>
<td>613</td>
<td></td>
</tr>
<tr>
<td>745</td>
<td>319</td>
<td>320</td>
<td></td>
</tr>
<tr>
<td>336</td>
<td>325</td>
<td>330</td>
<td></td>
</tr>
<tr>
<td>385</td>
<td>608</td>
<td>53</td>
<td></td>
</tr>
<tr>
<td>523</td>
<td>319</td>
<td>320</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Delivery Hops</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>21</td>
<td>15</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>3</td>
<td>9</td>
<td></td>
</tr>
<tr>
<td>42</td>
<td>8</td>
<td>21</td>
<td></td>
</tr>
<tr>
<td>51</td>
<td>41</td>
<td>11</td>
<td></td>
</tr>
<tr>
<td>45</td>
<td>10</td>
<td>24</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>13</td>
<td>39</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>35</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>33</td>
<td>4</td>
<td>45</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>3</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>36</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>139</td>
<td>4</td>
<td>11</td>
<td></td>
</tr>
<tr>
<td>31.5</td>
<td>15.0833</td>
<td>17</td>
<td></td>
</tr>
</tbody>
</table>
Table 5.2. Sample data for the comparisons (40-60) mobile agents

<table>
<thead>
<tr>
<th></th>
<th>Agent Number 40</th>
<th></th>
<th>Agent Number 50</th>
<th></th>
<th>Agent Number 60</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Random Smart</td>
<td>Smarter</td>
<td>Random Smart</td>
<td>Smarter</td>
<td>Random Smart</td>
<td>Smarter</td>
</tr>
<tr>
<td>convergence</td>
<td>331 49 44</td>
<td></td>
<td>316 42 42</td>
<td></td>
<td>314 51 41</td>
<td></td>
</tr>
<tr>
<td></td>
<td>344 40 44</td>
<td></td>
<td>325 48 40</td>
<td></td>
<td>324 41 50</td>
<td></td>
</tr>
<tr>
<td></td>
<td>325 41 41</td>
<td></td>
<td>343 39 40</td>
<td></td>
<td>326 42 43</td>
<td></td>
</tr>
<tr>
<td></td>
<td>312 39 318</td>
<td></td>
<td>314 50 41</td>
<td></td>
<td>311 39 39</td>
<td></td>
</tr>
<tr>
<td></td>
<td>333 40 39</td>
<td></td>
<td>322 45 44</td>
<td></td>
<td>340 43 47</td>
<td></td>
</tr>
<tr>
<td></td>
<td>327 57 39</td>
<td></td>
<td>348 40 44</td>
<td></td>
<td>330 39 38</td>
<td></td>
</tr>
<tr>
<td></td>
<td>333 38 54</td>
<td></td>
<td>314 49 40</td>
<td></td>
<td>328 44 39</td>
<td></td>
</tr>
<tr>
<td></td>
<td>319 40 40</td>
<td></td>
<td>315 42 41</td>
<td></td>
<td>329 41 37</td>
<td></td>
</tr>
<tr>
<td></td>
<td>340 315 44</td>
<td></td>
<td>336 44 55</td>
<td></td>
<td>316 40 41</td>
<td></td>
</tr>
<tr>
<td></td>
<td>318 39 48</td>
<td></td>
<td>329 40 52</td>
<td></td>
<td>318 42 37</td>
<td></td>
</tr>
<tr>
<td></td>
<td>354 47 55</td>
<td></td>
<td>319 39 55</td>
<td></td>
<td>314 40 39</td>
<td></td>
</tr>
<tr>
<td></td>
<td>320 39 40</td>
<td></td>
<td>319 48 41</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>convergence</td>
<td>329.67 65.333</td>
<td>67.17</td>
<td>325 43.833</td>
<td>44.583</td>
<td>322.727 42 41</td>
<td></td>
</tr>
<tr>
<td>Total Steps</td>
<td>341 538 69</td>
<td></td>
<td>354 54 51</td>
<td></td>
<td>324 66 52</td>
<td></td>
</tr>
<tr>
<td></td>
<td>352 59 53</td>
<td></td>
<td>345 73 60</td>
<td></td>
<td>346 327 573</td>
<td></td>
</tr>
<tr>
<td></td>
<td>348 773 57</td>
<td></td>
<td>370 48 59</td>
<td></td>
<td>356 57 54</td>
<td></td>
</tr>
<tr>
<td></td>
<td>334 76 337</td>
<td></td>
<td>317 680 56</td>
<td></td>
<td>319 54 49</td>
<td></td>
</tr>
<tr>
<td></td>
<td>345 566 48</td>
<td></td>
<td>347 54 319</td>
<td></td>
<td>343 51 59</td>
<td></td>
</tr>
<tr>
<td></td>
<td>348 581 52</td>
<td></td>
<td>368 51 74</td>
<td></td>
<td>350 49 49</td>
<td></td>
</tr>
<tr>
<td></td>
<td>336 54 322</td>
<td></td>
<td>327 76 57</td>
<td></td>
<td>355 47 66</td>
<td></td>
</tr>
<tr>
<td></td>
<td>322 52 557</td>
<td></td>
<td>317 53 53</td>
<td></td>
<td>332 408 334</td>
<td></td>
</tr>
<tr>
<td></td>
<td>350 319 52</td>
<td></td>
<td>339 55 577</td>
<td></td>
<td>370 58 52</td>
<td></td>
</tr>
<tr>
<td></td>
<td>327 50 54</td>
<td></td>
<td>339 337 75</td>
<td></td>
<td>321 490 50</td>
<td></td>
</tr>
<tr>
<td></td>
<td>368 48 483</td>
<td></td>
<td>333 410 55</td>
<td></td>
<td>355 66 57</td>
<td></td>
</tr>
<tr>
<td></td>
<td>348 53</td>
<td></td>
<td>326 49 50</td>
<td></td>
<td>67 57</td>
<td></td>
</tr>
<tr>
<td></td>
<td>343.25 283.27</td>
<td>178.1</td>
<td>340.17 161.67</td>
<td>123.83</td>
<td>342.818 145 121</td>
<td></td>
</tr>
<tr>
<td>Delivery Hops</td>
<td>34 25</td>
<td>12 9</td>
<td>17 11</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>18 9</td>
<td>25 15</td>
<td>43 22</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>80 15</td>
<td>9 15</td>
<td>10 11</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>17 3</td>
<td>40 15</td>
<td>13 10</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>34 9</td>
<td>9 4</td>
<td>10 11</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>20 13</td>
<td>12 23</td>
<td>23 9</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>16 4</td>
<td>22 17</td>
<td>12 18</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>12 75</td>
<td>46 12</td>
<td>9 11</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>4 12</td>
<td>42 4</td>
<td>27 11</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>11 10</td>
<td>11 50</td>
<td>24 9</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>89 26</td>
<td>11 23</td>
<td>23 12</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>19</td>
<td>9</td>
<td>11</td>
<td>12</td>
<td>12</td>
</tr>
<tr>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
</tr>
<tr>
<td></td>
<td>28.667</td>
<td>18.33</td>
<td>20.667</td>
<td>16.5</td>
<td>18.583</td>
<td>12.25</td>
</tr>
</tbody>
</table>
CHAPTER 6

CONCLUSION

An ad hoc mobile network is a collection of nodes, each of which is capable of moving, resulting in continual changes in the topology of the network. These nodes communicate through wireless transmission, and each of them serves as a router for the other network nodes. Ad hoc networks have many unique characteristics that make network communication challenging. Among all difficulties, it has been widely recognized that routing strategy is the most important research problem. [1]

6.1 The Thesis Summery

Although the routing problem for the MANET has existed for decades and miscellaneous algorithms have been proposed for it, those algorithms' inherent shortcomings are still the obstacles to prevent them from realistic applications. With the popularity of the mobile agents module employed as a powerful tool in the distributed computing, the ants based routing algorithms provide a more promising paradigm for MANETs' routing problems. However, both pure ants based algorithms and the hybrid algorithm (Ant-AODV) do not employ the ants' nature merits fully to make routing efficiently.

Through the thorough analysis on [73], we conclude that the migration strategy adopted by
the ants based routing algorithm plays a significant role on affecting the whole protocol’s efficiency. With the application of a new introduced table (AMIT), an innovative navigation strategy exploiting the collaborations among agents is proposed in this thesis. The simulation based comparisons between the new strategy and the other traditional two strategies show that the new proposed navigation strategy substantially improves the efficiency (37%) of the ants based routing protocol, while keeping the merit of fast convergence from the good traditional strategy. The simulated results also indicate the application of the new navigation strategy is able to improve the performance of the Ant-AODV protocol, which is considered as a more realistic routing protocol for MANETs in this thesis.

Based on Ant-AODV protocol, several important revisions are made to gain better performance. Through the structure extension of the routing table used in Ant-AODV, less end to end delay and more reliable routing can be achieved. Multiple entries in the routing table not only introduce the backup routes in the protocol, but also make the two constituent sub-protocols work more efficiently. The original introduction of the “message passing” module into the pure agents based paradigm makes the route maintenance more efficiently. The result of the “link failure notification” is the reduction of the further useless propagation on the non-existed routes, which increases the “goodput” and saves the scarce resource.

6.2 The Future Works

Ad hoc mobile networking is currently one of the most rapidly growing research areas. Ants based routing protocols or Ant-AODV like ant-hybrid protocols taking the advantages of the ant colony overcome some drawbacks of the traditional routing algorithms for MANETs
and inspire more researchers working on this paradigm. In the future works, we plan to continue strengthen the employment of the ants (especially the inter-agent communications) in the Ant-AODV protocol, and compare more criterions on the protocols based on the NS-2 simulator. To maintain a loop-free route in the agents based paradigm is still an open problem in this area. We will concentrate our research on this important characteristic in the traditional routing protocols.
REFERENCES


Routing(DSDV) for Mobile Computers”. Computer Communication Review, pages 234-244,


[10] C. E. Perkins and E. M. Royer. “Ad-Hoc On-Demand Distance Vector Routing”. In


ftp://iridia.ulb.ac.be/pub/dorigo/conferences/IC.22-HICSS31.ps.gz


Mobile Ad Hoc Networks”, *Symposium on Ad Hoc Wireless Network, National University of Singapore*


[26] C. Krugel, and T. Toth, “SPARTA a Mobile Agent based Intrusion Detection System”, In *6th International Conference on Telecommunications (ConTEL), Croatia, June 2001*


[33] A. Lingnau, O. Drobnik, and P. Domel, "An HTTP based infrastructure for mobile agents", *WWW J. (Fourth Int. WWW Conf.), December 1995*

[34] R. Gray, "Agent Tcl: a transportable agent system", in *Proc. CIKM Workshop Intell Inform Agents (CIKM '95), 1995*


General Magic White Paper, 1994


[46] D Bertsekas. "Dynamic Programming and Optimal Control". Athena Scientific, Belmont,


(WNMC) in conjunction with the International Conf. on Distributed Computing Systems (ICDCS). Pages 425-432, Apr. 2001


VITA

Graduate College
University Of Nevada, Las Vegas

Jun Wang

Home Address:
1600 E. University Ave, Apt. 225
Las Vegas, NV 89119

Degrees:
Bachelor of Science, Computer Science, 1997
Beijing Polytechnic University, Beijing, P.R. China

Thesis Title:
An improved MultiAnts-AODV routing protocol for Ad Hoc Wireless Networks

Thesis Examination Committee:
Committee Chair, Dr. Ajoy K. Datta, Ph.D.
Committee Chair, Dr. Maria Gradinariu, Ph.D.
Committee Member, Dr. Kazem Tagha, Ph.D.
Committee Member, Dr. Wolfgang Bein, Ph.D.
Committee Member, Dr. Z. Y. Wang, Ph.D.